Group Ring Module

Hidetsune Kobayashi, L. Chen, H. Murao

June 16, 2019

Abstract

The theory of groups, rings and modules is developed to a great depth. Group theory results include Zassenhaus's theorem and the Jordan-Hoelder theorem. The ring theory development includes ideals, quotient rings and the Chinese remainder theorem. The module development includes the Nakayama lemma, exact sequences and Tensor products.

Contents

1 Preliminaries **4**
1.1 Lemmas for logical manipulation 4
1.2 Natural numbers and Integers 4
1.2.1 Integers 6
1.3 Sets 9
1.3.1 A short notes for proof steps 9
1.3.2 Sets 9
1.4 Functions 12
1.5 Nsets 19
1.5.1 Lemmas for existence of reduced chain. 26
1.6 Lower bounded set of integers 26
1.7 Augmented integer: integer and $\infty - \infty$ 27
1.7.1 Ordering of integers and ordering nats 36
1.7.2 The $\leq$ Ordering 36
1.7.3 Aug ordering 39
1.8 Amin, amax 41
1.8.1 Maximum element of a set of ants 43
1.9 Cardinality of sets 45
1.9.1 Lemmas required in Algebra6.thy 50

2 Ordered Set **53**
2.1 Basic Concepts of Ordered Sets 53
2.1.1 Total ordering 56
2.1.2 Two ordered sets 57
2.1.3 Homomorphism of ordered sets 58
2.2 Pre elements 77
2.3 Transfinite induction 78
2.4 *Ordered-set2*. Lemmas to prove Zorn's lemma. 78
2.5 Zorn's lemma 79

3 Group Theory. Focused on Jordan Hoelder theorem **91**
3.1 Definition of a Group 91
3.2 Subgroups 93

3.3 Cosets . . . 100
3.4 Normal subgroups and Quotient groups . . . 103
3.5 Setproducts . . . 107
3.6 Preliminary lemmas for Zassenhaus . . . 109
3.7 Homomorphism . . . 112
3.8 Gkernel . . . 113
3.9 Image . . . 115
3.10 Induced homomorphisms . . . 116
3.10.1 Homomorphism therems . . . 117
3.11 Isomorphims . . . 120
3.11.1 An automorphism groups . . . 120
3.11.2 Complete system of representatives . . . 120
3.12 Zassenhaus . . . 122
3.13 Chain of groups I . . . 124
3.14 Existence of reduced chain . . . 127
3.15 Existence of reduced chain and composition series . . . 132
3.16 Chain of groups II . . . 132
3.17 Jordan Hoelder theorem . . . 137
3.17.1 *Rfn-tools.* Tools to treat refinement of a cmpser, rtos. 137
3.18 Abelian groups . . . 146
3.18.1 Homomorphism of abelian groups . . . 151
3.18.2 Quotient abelian group . . . 153
3.19 Direct product and direct sum of abelian groups, in general case . . . 155
3.19.1 Characterization of a direct product . . . 160

4 Ring theory 163
4.1 Definition of a ring and an ideal . . . 163
4.2 Calculation of elements . . . 167
4.2.1 nscale . . . 167
4.2.2 npow . . . 168
4.2.3 nsum and fSum . . . 169
4.3 Ring homomorphisms . . . 173
4.3.1 Ring of integers . . . 179
4.4 Quotient rings . . . 179
4.5 Primary ideals, Prime ideals . . . 184
4.6 Operation of ideals . . . 195
4.7 Direct product1, general case . . . 197
4.8 Chinese remainder theorem . . . 201
4.9 Addition of finite elements of a ring and *ideal-multiplication* . 204
4.10 Extension and contraction . . . 211
4.11 Complete system of representatives . . . 212
4.12 Polynomial ring . . . 213
4.13 Addition and multiplication of *polyn-exprs* . . . 214

4.13.1 Simple properties of a *polyn-ring* 214
4.13.2 Coefficients of a polynomial 215
4.13.3 Addition of *polyn-exprs* 215
4.13.4 Multiplication of *pol-exprs* 219
4.13.5 Multiplication 219
4.14 The degree of a polynomial 222
4.14.1 Multiplication of polynomials 229
4.14.2 Degree with value in *aug-minf* 230
4.15 Homomorphism of polynomial rings 231
4.16 Relatively prime polynomials 236
4.16.1 Polynomial, coeff mod P 237

5 Modules 248
5.1 Basic properties of Modules 248
5.2 Injective hom, surjective hom, bijective hom and inverse hom 254
5.3 nsum and Generators 271
5.3.1 Sum up coefficients 273
5.3.2 Free generators 275
5.4 nsum and Generators (continued) 278
5.5 Existence of homomorphism 279
5.6 Nakayama lemma 285
5.7 Direct sum and direct products of modules 286
5.8 Exact sequence 288
5.9 Tensor product 293

6 Construction of an abelian group 296
6.1 Free generated abelian group I, direct sum and direct product 2296
6.2 Abelian group generated by a singleton (constructive) 300
6.3 Abelian Group generated by one element II (nonconstructive) 306
6.4 Free Generated Modules (constructive) 309
6.5 A fgmodule and a free module 313
6.6 Direct sum, again 314
6.6.1 Existence of the tensor product 316

theory *Algebra1*
imports *Main HOL−Library.FuncSet*
begin

Chapter 1

Preliminaries

Some of the lemmas of this section are proved in src/HOL/Integ of Isabelle version 2003.

1.1 Lemmas for logical manipulation

lemma *True-then*:$True \longrightarrow P \Longrightarrow P$
⟨*proof*⟩

lemma *ex-conjI*:$[\![P\ c;\ Q\ c]\!] \Longrightarrow \exists\, c.\ P\ c \wedge Q\ c$
⟨*proof*⟩

lemma *forall-spec*:$[\![\ \forall\, b.\ P\ b \longrightarrow Q\ b;\ P\ a]\!] \Longrightarrow Q\ a$
⟨*proof*⟩

lemma *a-b-exchange*:$[\![a;\ a = b]\!] \Longrightarrow b$
⟨*proof*⟩

lemma *eq-prop*:$[\![\ P;\ P = Q]\!] \Longrightarrow Q$
⟨*proof*⟩

lemma *forball-contra*:$[\![\forall\, y{\in}A.\ P\ x\ y \longrightarrow \neg\ Q\ y;\ \forall\, y{\in}A.\ Q\ y \vee R\ y]\!] \Longrightarrow$
$\forall\, y{\in}A.\ (\neg\ P\ x\ y) \vee R\ y$
⟨*proof*⟩

lemma *forball-contra1*:$[\![\forall\, y{\in}A.\ P\ x\ y \longrightarrow Q\ y;\ \forall\, y{\in}A.\ \neg\ Q\ y]\!] \Longrightarrow \forall\, y{\in}A.\ \neg\ P\ x$
y
⟨*proof*⟩

1.2 Natural numbers and Integers

Elementary properties of natural numbers and integers

lemma *nat-nonzero-pos*:$(a{::}nat) \neq 0 \Longrightarrow 0 < a$

⟨*proof*⟩

lemma *add-both*:$(a::nat) = b \Longrightarrow a + c = b + c$
⟨*proof*⟩

lemma *add-bothl*:$a = b \Longrightarrow c + a = c + b$
⟨*proof*⟩

lemma *diff-Suc*:$(n::nat) \leq m \Longrightarrow m - n + Suc\ 0 = Suc\ m - n$
⟨*proof*⟩

lemma *le-convert*:$[\![a = b;\ a \leq c]\!] \Longrightarrow b \leq c$
⟨*proof*⟩

lemma *ge-convert*:$[\![a = b;\ c \leq a]\!] \Longrightarrow c \leq b$
⟨*proof*⟩

lemma *less-convert*:$[\![\ a = b;\ c < b\ ]\!] \Longrightarrow c < a$
⟨*proof*⟩

lemma *ineq-conv1*:$[\![a = b;\ a < c]\!] \Longrightarrow b < c$
⟨*proof*⟩

lemma *diff-Suc-pos*:$0 < a - Suc\ 0 \Longrightarrow\ 0 < a$
⟨*proof*⟩

lemma *minus-SucSuc*:$a - Suc\ (Suc\ 0) = a - Suc\ 0 - Suc\ 0$
⟨*proof*⟩

lemma *Suc-Suc-Tr*:$Suc\ (Suc\ 0) \leq n \Longrightarrow Suc\ (n - Suc\ (Suc\ 0)) = n - Suc\ 0$
⟨*proof*⟩

lemma *Suc-Suc-less*:$Suc\ 0 < a \Longrightarrow Suc\ (a - Suc\ (Suc\ 0)) < a$
⟨*proof*⟩

lemma *diff-zero-eq*:$n = (0::nat) \Longrightarrow m = m - n$
⟨*proof*⟩

lemma *Suc-less-le*:$x < Suc\ n \Longrightarrow x \leq n$
⟨*proof*⟩

lemma *less-le-diff*:$x < n \Longrightarrow x \leq n - Suc\ 0$
⟨*proof*⟩

lemma *le-pre-le*:$x \leq n - Suc\ 0 \Longrightarrow x \leq n$
⟨*proof*⟩

lemma *nat-not-less*:$\neg\ (m::nat) < n \Longrightarrow n \leq m$
⟨*proof*⟩

lemma *less-neq*:$n < (m::nat) \Longrightarrow n \neq m$
⟨*proof*⟩

lemma *less-le-diff1*:$n \neq 0 \Longrightarrow ((m::nat) < n) = (m \leq (n - Suc\ 0))$
⟨*proof*⟩

lemma *nat-not-less1*:$n \neq 0 \Longrightarrow (\neg (m::nat) < n) = (\neg m \leq (n - Suc\ 0))$
⟨*proof*⟩

lemma *nat-eq-le*:$m = (n::nat) \Longrightarrow m \leq n$
⟨*proof*⟩

1.2.1 Integers

lemma *non-zero-int*: $(n::int) \neq 0 \Longrightarrow 0 < n \vee n < 0$
⟨*proof*⟩

lemma *zgt-0-zge-1*:$(0::int) < z \Longrightarrow 1 \leq z$
⟨*proof*⟩

lemma *not-zle*:$(\neg (n::int) \leq m) = (m < n)$
⟨*proof*⟩

lemma *not-zless*:$(\neg (n::int) < m) = (m \leq n)$
⟨*proof*⟩

lemma *zle-imp-zless-or-eq*:$(n::int) \leq m \Longrightarrow n < m \vee n = m$
⟨*proof*⟩

lemma *zminus-zadd-cancel*: $- z + (z + w) = (w::int)$
⟨*proof*⟩

lemma *int-neq-iff*:$((w::int) \neq z) = (w < z) \vee (z < w)$
⟨*proof*⟩

lemma *zless-imp-zle*:$(z::int) < z' \Longrightarrow z \leq z'$
⟨*proof*⟩

lemma *zdiff*:$z - (w::int) = z + (- w)$
⟨*proof*⟩

lemma *zle-zless-trans*:$[\![(i::int) \leq j; j < k]\!] \Longrightarrow i < k$
⟨*proof*⟩

lemma *zless-zle-trans*:$[\![(i::int) < j; j \leq k]\!] \Longrightarrow i < k$
⟨*proof*⟩

lemma *zless-neq*:$(i::int) < j \Longrightarrow i \neq j$

⟨*proof*⟩

lemma *int-mult-mono*:$[\![\ i < j;\ (0::int) < k\ ]\!] \Longrightarrow k * i < k * j$
⟨*proof*⟩

lemma *int-mult-le*:$[\![i \leq j;\ (0::int) \leq k]\!] \Longrightarrow k * i \leq k * j$
⟨*proof*⟩

lemma *int-mult-le1*:$[\![i \leq j;\ (0::int) \leq k]\!] \Longrightarrow i * k \leq j * k$
⟨*proof*⟩

lemma *zmult-zminus-right*:$(w::int) * (-\ z) = -\ (w * z)$
⟨*proof*⟩

lemma *zmult-zle-mono1-neg*:$[\![(i::int) \leq j;\ k \leq 0]\!] \Longrightarrow j * k \leq i * k$
⟨*proof*⟩

lemma *zmult-zless-mono-neg*:$[\![(i::int) < j;\ k < 0]\!] \Longrightarrow j * k < i * k$
⟨*proof*⟩

lemma *zmult-neg-neg*:$[\![i < (0::int);\ j < 0\ ]\!] \Longrightarrow 0 < i * j$
⟨*proof*⟩

lemma *zmult-pos-pos*:$[\![(0::int) < i;\ 0 < j]\!] \Longrightarrow 0 < i * j$
⟨*proof*⟩

lemma *zmult-pos-neg*:$[\![(0::int) < i;\ j < 0]\!] \Longrightarrow i * j < 0$
⟨*proof*⟩

lemma *zmult-neg-pos*:$[\![i < (0::int);\ 0 < j]\!] \Longrightarrow i * j < 0$
⟨*proof*⟩

lemma *zle*:$((z::int) \leq w) = (\neg\ (w < z))$
⟨*proof*⟩

lemma *times-1-both*:$[\![(0::int) < z;\ z * z' = 1]\!] \Longrightarrow z = 1 \wedge z' = 1$
⟨*proof*⟩

lemma *zminus-minus*:$i - -\ (j::int) = i + j$
⟨*proof*⟩

lemma *zminus-minus-pos*:$(n::int) < 0 \Longrightarrow 0 < -\ n$
⟨*proof*⟩

lemma *zadd-zle-mono*:$[\![w' \leq w;\ z' \leq (z::int)]\!] \Longrightarrow w' + z' \leq w + z$
⟨*proof*⟩

lemma *zmult-zle-mono*:$[\![i \leq (j::int);\ 0 < k]\!] \Longrightarrow k * i \leq\ k * j$
⟨*proof*⟩

lemma *zmult-zle-mono-r*: $[\![i \le (j::int);\ 0 < k]\!] \Longrightarrow i * k \le j * k$
⟨*proof*⟩

lemma *pos-zmult-pos*: $[\![\ 0 \le (a::int);\ 0 < (b::int)]\!] \Longrightarrow a \le a * b$
⟨*proof*⟩

lemma *pos-mult-l-gt*: $[\![(0::int) < w;\ i \le j;\ 0 \le i]\!] \Longrightarrow i \le w * j$
⟨*proof*⟩

lemma *pos-mult-r-gt*: $[\![(0::int) < w;\ i \le j;\ 0 \le i]\!] \Longrightarrow i \le j * w$
⟨*proof*⟩

lemma *mult-pos-iff*: $[\![(0::int) < i;\ 0 \le i * j\ ]\!] \Longrightarrow 0 \le j$
⟨*proof*⟩

lemma *zmult-eq*: $[\![(0::int) < w;\ z = z']\!] \Longrightarrow w * z = w * z'$
⟨*proof*⟩

lemma *zmult-eq-r*: $[\![(0::int) < w;\ z = z']\!] \Longrightarrow z * w = z' * w$
⟨*proof*⟩

lemma *zdiv-eq-l*: $[\![(0::int) < w;\ z * w = z' * w\ ]\!] \Longrightarrow z = z'$
⟨*proof*⟩

lemma *zdiv-eq-r*: $[\![(0::int) < w;\ w * z = w * z'\ ]\!] \Longrightarrow z = z'$
⟨*proof*⟩

lemma *int-nat-minus*: $0 < (n::int) \Longrightarrow nat\ (n - 1) = (nat\ n) - 1$
⟨*proof*⟩

lemma *int-nat-add*: $[\![0 < (n::int);\ 0 < (m::int)]\!] \Longrightarrow (nat\ (n - 1)) + (nat\ (m - 1)) + (Suc\ 0) = nat\ (n + m - 1)$
⟨*proof*⟩

lemma *int-equation*: $(x::int) = y + z \Longrightarrow x - y = z$
⟨*proof*⟩

lemma *int-pos-mult-monor*: $[\![\ 0 < (n::int);\ 0 \le n * m\ ]\!] \Longrightarrow 0 \le m$
⟨*proof*⟩

lemma *int-pos-mult-monol*: $[\![\ 0 < (m::int);\ 0 \le n * m\ ]\!] \Longrightarrow 0 \le n$
⟨*proof*⟩

lemma *zdiv-positive*: $[\![(0::int) \le a;\ 0 < b]\!] \Longrightarrow 0 \le a\ div\ b$
⟨*proof*⟩

lemma *zdiv-pos-mono-r*: $[\![\ (0::int) < w;\ w * z \le w * z']\!] \Longrightarrow z \le z'$

$\langle proof \rangle$

lemma *zdiv-pos-mono-l*:$[\![(0::int) < w; z * w \leq z' * w]\!] \Longrightarrow z \leq z'$
$\langle proof \rangle$

lemma *zdiv-pos-pos-l*:$[\![(0::int) < w; 0 \leq z * w]\!] \Longrightarrow 0 \leq z$
$\langle proof \rangle$

1.3 Sets

1.3.1 A short notes for proof steps

1.3.2 Sets

lemma *inEx*:$x \in A \Longrightarrow \exists y \in A.\ y = x$
$\langle proof \rangle$

lemma *inEx-rev*: $\exists y \in A.\ y = x \Longrightarrow x \in A$
$\langle proof \rangle$

lemma *nonempty-ex*:$A \neq \{\} \Longrightarrow \exists x.\ x \in A$
$\langle proof \rangle$

lemma *ex-nonempty*:$\exists x.\ x \in A \Longrightarrow A \neq \{\}$
$\langle proof \rangle$

lemma *not-eq-outside*:$a \notin A \Longrightarrow \forall b \in A.\ b \neq a$
$\langle proof \rangle$

lemma *ex-nonempty-set*:$\exists a.\ P\ a \Longrightarrow \{x.\ P\ x\} \neq \{\}$
$\langle proof \rangle$

lemma *nonempty*: $x \in A \Longrightarrow A \neq \{\}$
$\langle proof \rangle$

lemma *subset-self*:$A \subseteq A$
$\langle proof \rangle$

lemma *conditional-subset*:$\{x \in A.\ P\ x\} \subseteq A$
$\langle proof \rangle$

lemma *bsubsetTr*:$\{x.\ x \in A \wedge P\ x\} \subseteq A$
$\langle proof \rangle$

lemma *sets-not-eq*:$[\![A \neq B; B \subseteq A]\!] \Longrightarrow \exists a \in A.\ a \notin B$
$\langle proof \rangle$

lemma *diff-nonempty*:$[\![A \neq B; B \subseteq A]\!] \Longrightarrow A - B \neq \{\}$
$\langle proof \rangle$

lemma *sub-which1*: $[\![A \subseteq B \vee B \subseteq A;\ x \in A;\ x \notin B]\!] \Longrightarrow B \subseteq A$
⟨*proof*⟩

lemma *sub-which2*: $[\![A \subseteq B \vee B \subseteq A;\ x \notin A;\ x \in B]\!] \Longrightarrow A \subseteq B$
⟨*proof*⟩

lemma *nonempty-int*: $A \cap B \neq \{\} \Longrightarrow \exists x.\ x \in A \cap B$
⟨*proof*⟩

lemma *no-meet1*: $A \cap B = \{\} \Longrightarrow \forall a \in A.\ a \notin B$
⟨*proof*⟩

lemma *no-meet2*: $A \cap B = \{\} \Longrightarrow \forall a \in B.\ a \notin A$
⟨*proof*⟩

lemma *elem-some*: $x \in A \Longrightarrow \exists y \in A.\ x = y$
⟨*proof*⟩

lemma *singleton-sub*: $a \in A \Longrightarrow \{a\} \subseteq A$
⟨*proof*⟩

lemma *eq-elem-in*: $[\![\ a \in A;\ a = b\ ]\!] \Longrightarrow b \in A$
⟨*proof*⟩

lemma *eq-set-inc*: $[\![\ a \in A;\ A = B\ ]\!] \Longrightarrow a \in B$
⟨*proof*⟩

lemma *eq-set-not-inc*: $[\![a \notin A;\ A = B\ ]\!] \Longrightarrow a \notin B$
⟨*proof*⟩

lemma *int-subsets*: $[\![\ A1 \subseteq A;\ B1 \subseteq B\ ]\!] \Longrightarrow A1 \cap B1 \subseteq A \cap B$
⟨*proof*⟩

lemma *inter-mono*: $A \subseteq B \Longrightarrow A \cap C \subseteq B \cap C$
⟨*proof*⟩

lemma *sub-Un1*: $B \subseteq B \cup C$
⟨*proof*⟩

lemma *sub-Un2*: $C \subseteq B \cup C$
⟨*proof*⟩

lemma *subset-contr*: $[\![\ A \subset B;\ B \subseteq A\ ]\!] \Longrightarrow$ *False*
⟨*proof*⟩

lemma *psubset-contr*: $[\![\ A \subset B;\ B \subset A\ ]\!] \Longrightarrow$ *False*
⟨*proof*⟩

lemma *eqsets-sub*: $A = B \Longrightarrow A \subseteq B$
⟨*proof*⟩

lemma *not-subseteq*: $\neg\ A \subseteq B \Longrightarrow \exists\, a \in A.\ a \notin B$
⟨*proof*⟩

lemma *in-un1*: $[\![\ x \in A \cup B;\ x \notin B\ ]\!] \Longrightarrow x \in A$
⟨*proof*⟩

lemma *proper-subset*: $[\![A \subseteq B;\ x \notin A;\ x \in B]\!] \Longrightarrow A \neq B$
⟨*proof*⟩

lemma *in-un2*: $[\![\ x \in A \cup B;\ x \notin A\ ]\!] \Longrightarrow x \in B$
⟨*proof*⟩

lemma *diff-disj*: $x \notin A \Longrightarrow A - \{x\} = A$
⟨*proof*⟩

lemma *in-diff*: $[\![x \neq a;\ x \in A]\!] \Longrightarrow x \in A - \{a\}$
⟨*proof*⟩

lemma *in-diff1*: $x \in A - \{a\} \Longrightarrow x \neq a$
⟨*proof*⟩

lemma *sub-inserted1*: $[\![Y \subseteq \mathit{insert}\ a\ X;\ \neg\ Y \subseteq X]\!] \Longrightarrow a \notin X \wedge a \in Y$
⟨*proof*⟩

lemma *sub-inserted2*: $[\![Y \subseteq \mathit{insert}\ a\ X;\ \neg\ Y \subseteq X]\!] \Longrightarrow Y = (Y - \{a\}) \cup \{a\}$
⟨*proof*⟩

lemma *insert-sub*: $[\![\ A \subseteq B;\ a \in B]\!] \Longrightarrow (\mathit{insert}\ a\ A) \subseteq B$
⟨*proof*⟩

lemma *insert-diff*: $A \subseteq (\mathit{insert}\ b\ B) \Longrightarrow A - \{b\} \subseteq B$
⟨*proof*⟩

lemma *insert-inc1*: $A \subseteq \mathit{insert}\ a\ A$
⟨*proof*⟩

lemma *insert-inc2*: $a \in \mathit{insert}\ a\ A$
⟨*proof*⟩

lemma *nonempty-some*: $A \neq \{\} \Longrightarrow (\mathit{SOME}\ x.\ x \in A) \in A$
⟨*proof*⟩

lemma *mem-family-sub-Un*: $A \in C \Longrightarrow A \subseteq \bigcup\ C$
⟨*proof*⟩

lemma *sub-Union*: $\exists\, X{\in}C.\ A \subseteq X \Longrightarrow A \subseteq \bigcup\ C$

⟨*proof*⟩

lemma *family-subset-Un-sub*:$\forall A{\in}C.\ A \subseteq B \Longrightarrow \bigcup\ C \subseteq B$
⟨*proof*⟩

lemma *in-set-with-P*:$P\ x \Longrightarrow x \in \{y.\ P\ y\}$
⟨*proof*⟩

lemma *sub-single*:$[\![A \neq \{\};\ A \subseteq \{a\}]\!] \Longrightarrow A = \{a\}$
⟨*proof*⟩

lemma *not-sub-single*:$[\![A \neq \{\};\ A \neq \{a\}]\!] \Longrightarrow \neg\ A \subseteq \{a\}$
⟨*proof*⟩

lemma *not-sub*:$\neg\ A \subseteq B \Longrightarrow \exists\, a.\ a{\in}A \wedge a \notin B$
⟨*proof*⟩

1.4 Functions

definition
cmp :: $['b \Rightarrow 'c,\ 'a \Rightarrow 'b] \Rightarrow ('a \Rightarrow 'c)$ **where**
$cmp\ g\ f = (\lambda x.\ g\ (f\ x))$

definition
idmap :: $'a\ set \Rightarrow ('a \Rightarrow 'a)$ **where**
$idmap\ A = (\lambda x{\in}A.\ x)$

definition
constmap :: $['a\ set,\ 'b\ set] \Rightarrow ('a \Rightarrow 'b)$ **where**
$constmap\ A\ B = (\lambda x{\in}A.\ SOME\ y.\ y \in B)$

definition
invfun :: $['a\ set,\ 'b\ set,\ 'a \Rightarrow 'b] \Rightarrow ('b \Rightarrow 'a)$ **where**
$invfun\ A\ B\ (f :: 'a \Rightarrow 'b) = (\lambda y{\in}B.(SOME\ x.\ (x \in A \wedge f\ x = y)))$

abbreviation
INVFUN :: $['a \Rightarrow 'b,\ 'b\ set,\ 'a\ set] \Rightarrow ('b \Rightarrow 'a)$ $((3\text{-}^{-1}\text{-},\text{-})\ [82,82,83]82)$ **where**
$f^{-1}{}_{B,A} == invfun\ A\ B\ f$

lemma *eq-fun*:$[\![\ f \in A \rightarrow B;\ f = g\ ]\!] \Longrightarrow g \in A \rightarrow B$
⟨*proof*⟩

lemma *eq-fun-eq-val*: $f = g \Longrightarrow f\ x = g\ x$
⟨*proof*⟩

lemma *eq-elems-eq-val*:$x = y \Longrightarrow f\ x = f\ y$
⟨*proof*⟩

lemma *cmp-fun*:$[\![f \in A \rightarrow B;\ g \in B \rightarrow C\ ]\!] \Longrightarrow cmp\ g\ f \in A \rightarrow C$

⟨*proof*⟩

lemma *cmp-fun-image*: $[\![f \in A \to B;\ g \in B \to C\ ]\!] \Longrightarrow$
$(cmp\ g\ f)\ `\ A = \ g\ `\ (f\ `\ A)$
⟨*proof*⟩

lemma *cmp-fun-sub-image*: $[\![f \in A \to B;\ g \in B \to C;\ A1 \subseteq A]\!] \Longrightarrow$
$(cmp\ g\ f)\ `\ A1 = \ g\ `\ (f\ `\ A1)$
⟨*proof*⟩

lemma *restrict-fun-eq*: $\forall\, x{\in}A.\ f\ x = g\ x \Longrightarrow (\lambda x{\in}A.\ f\ x) = (\lambda x{\in}A.\ g\ x)$
⟨*proof*⟩

lemma *funcset-mem*: $[\![f \in A \to B;\ x \in A]\!] \Longrightarrow f\ x \in B$
⟨*proof*⟩

lemma *img-subset*: $f \in A \to B \Longrightarrow f\ `\ A \subseteq B$
⟨*proof*⟩

lemma *funcset-mem1*: $[\![\forall\, l{\in}A.\ f\ l \in B;\ x \in A]\!] \Longrightarrow f\ x \in B$
⟨*proof*⟩

lemma *func-to-img*: $f \in A \to B \Longrightarrow f \in A \to f\ `\ A$
⟨*proof*⟩

lemma *restrict-in-funcset*: $\forall\, x{\in}\ A.\ f\ x \in B \Longrightarrow$
$(\lambda x{\in}A.\ f\ x){\in}\ A \to B$
⟨*proof*⟩

lemma *funcset-eq*: $[\![\ f \in extensional\ A;\ g \in extensional\ A;\ \forall\, x{\in}A.\ f\ x = g\ x\ ]\!] \Longrightarrow$
$f = g$
⟨*proof*⟩

lemma *eq-funcs*: $[\![f \in A \to B;\ g \in A \to B;\ f = g;\ x \in A]\!] \Longrightarrow f\ x = g\ x$
⟨*proof*⟩

lemma *restriction-of-domain*: $[\![\ f \in A \to B;\ A1 \subseteq A\ ]\!] \Longrightarrow$
$restrict\ f\ A1 \in A1 \to B$
⟨*proof*⟩

lemma *restrict-restrict*: $[\![\ restrict\ f\ A \in A \to B;\ A1 \subseteq A\ ]\!] \Longrightarrow$
$restrict\ (restrict\ f\ A)\ A1 = restrict\ f\ A1$
⟨*proof*⟩

lemma *restr-restr-eq*: $[\![\ restrict\ f\ A \in A \to B;\ restrict\ f\ A = restrict\ g\ A;$
$A1 \subseteq A\ ]\!] \Longrightarrow restrict\ f\ A1 = restrict\ g\ A1$
⟨*proof*⟩

lemma *funcTr*:$[\![$ $f \in A \to B$; $g \in A \to B$; $f = g$; $a \in A]\!] \Longrightarrow f\ a = g\ a$
⟨*proof*⟩

lemma *funcTr1*:$[\![f = g;\ a \in A]\!] \Longrightarrow f\ a = g\ a$
⟨*proof*⟩

lemma *restrictfun-im*:$[\![$ $(restrict\ f\ A) \in A \to B$; $A1 \subseteq A$ $]\!] \Longrightarrow$
$(restrict\ f\ A)$ ‘ $A1 = f$ ‘ $A1$
⟨*proof*⟩

lemma *mem-in-image*:$[\![$ $f \in A \to B$; $a \in A]\!] \Longrightarrow f\ a \in f$ ‘ A
⟨*proof*⟩

lemma *mem-in-image1*:$[\![$ $\forall\, l{\in}A.\ f\ l \in B$; $a \in A]\!] \Longrightarrow f\ a \in f$ ‘ A
⟨*proof*⟩

lemma *mem-in-image2*:$a \in A \Longrightarrow f\ a \in f$ ‘ A
⟨*proof*⟩

lemma *mem-in-image3*:$b \in f$ ‘ $A \Longrightarrow \exists\, a \in A.\ b = f\ a$
⟨*proof*⟩

lemma *elem-in-image2*: $[\![$ $f \in A \to B$; $A1 \subseteq A$; $x \in A1]\!] \Longrightarrow f\ x \in f$‘ $A1$
⟨*proof*⟩

lemma *funcs-nonempty*:$[\![$ $A \neq \{\}$; $B \neq \{\}$ $]\!] \Longrightarrow (A \to B) \neq \{\}$
⟨*proof*⟩

lemma *idmap-funcs*: $idmap\ A \in A \to A$
⟨*proof*⟩

lemma *l-idmap-comp*: $[\![f \in extensional\ A;\ f \in A \to B]\!] \Longrightarrow$
$compose\ A\ (idmap\ B)\ f = f$
⟨*proof*⟩

lemma *r-idmap-comp*:$[\![f \in extensional\ A;\ f \in A \to B]\!] \Longrightarrow$
$compose\ A\ f\ (idmap\ A) = f$
⟨*proof*⟩

lemma *extend-fun*: $[\![$ $f \in A \to B$; $B \subseteq B1$ $]\!] \Longrightarrow f \in A \to B1$
⟨*proof*⟩

lemma *restrict-fun*: $[\![$ $f \in A \to B$; $A1 \subseteq A$ $]\!] \Longrightarrow restrict\ f\ A1 \in A1 \to B$
⟨*proof*⟩

lemma *set-of-hom*: $\forall\, x \in A.\ f\ x \in B \Longrightarrow restrict\ f\ A \in A \to B$
⟨*proof*⟩

lemma *composition* : $[\![f \in A \to B; g \in B \to C]\!] \Longrightarrow (compose\ A\ g\ f) \in A \to C$
⟨*proof*⟩

lemma *comp-assoc*: $[\![f \in A \to B; g \in B \to C; h \in C \to D]\!] \Longrightarrow$
$compose\ A\ h\ (compose\ A\ g\ f) = compose\ A\ (compose\ B\ h\ g)\ f$
⟨*proof*⟩

lemma *restrictfun-inj*: $[\![inj\text{-}on\ f\ A; A1 \subseteq A]\!] \Longrightarrow inj\text{-}on\ (restrict\ f\ A1)\ A1$
⟨*proof*⟩

lemma *restrict-inj*: $[\![inj\text{-}on\ f\ A; A1 \subseteq A]\!] \Longrightarrow inj\text{-}on\ f\ A1$
⟨*proof*⟩

lemma *injective*: $[\![inj\text{-}on\ f\ A; x \in A; y \in A; x \neq y]\!] \Longrightarrow f\ x \neq f\ y$
⟨*proof*⟩

lemma *injective-iff*: $[\![inj\text{-}on\ f\ A; x \in A; y \in A]\!] \Longrightarrow$
$(x = y) = (f\ x = f\ y)$
⟨*proof*⟩

lemma *injfun-elim-image*: $[\![f \in A \to B; inj\text{-}on\ f\ A; x \in A]\!] \Longrightarrow$
$f\ `\ (A - \{x\}) = (f\ `\ A) - \{f\ x\}$
⟨*proof*⟩

lemma *cmp-inj*: $[\![f \in A \to B; g \in B \to C; inj\text{-}on\ f\ A; inj\text{-}on\ g\ B]\!] \Longrightarrow$
$inj\text{-}on\ (cmp\ g\ f)\ A$
⟨*proof*⟩

lemma *cmp-assoc*: $[\![f \in A \to B; g \in B \to C; h \in C \to D; x \in A]\!] \Longrightarrow$
$(cmp\ h\ (cmp\ g\ f))\ x = (cmp\ (cmp\ h\ g)\ f)\ x$
⟨*proof*⟩

lemma *bivar-fun*: $[\![f \in A \to (B \to C); a \in A]\!] \Longrightarrow f\ a \in B \to C$
⟨*proof*⟩

lemma *bivar-fun-mem*: $[\![f \in A \to (B \to C); a \in A; b \in B]\!] \Longrightarrow f\ a\ b \in C$
⟨*proof*⟩

lemma *bivar-func-eq*: $[\![\forall a{\in}A.\ \forall b{\in}B.\ f\ a\ b = g\ a\ b]\!] \Longrightarrow$
$(\lambda x{\in}A.\ \lambda y{\in}B.\ f\ x\ y) = (\lambda x{\in}A.\ \lambda y{\in}B.\ g\ x\ y)$
⟨*proof*⟩

lemma *set-image*: $[\![f \in A \to B; A1 \subseteq A; A2 \subseteq A]\!] \Longrightarrow$
$f`(A1 \cap A2) \subseteq (f`\ A1) \cap (f`\ A2)$
⟨*proof*⟩

lemma *image-sub*: $[\![f \in A \to B; A1 \subseteq A]\!] \Longrightarrow (f`A1) \subseteq B$
⟨*proof*⟩

lemma *image-sub0*: $f \in A \to B \Longrightarrow (f`A) \subseteq B$
⟨*proof*⟩

lemma *image-nonempty*: $[\![f \in A \to B;\ A1 \subseteq A;\ A1 \neq \{\}\ ]\!] \Longrightarrow f\ `\ A1 \neq \{\}$
⟨*proof*⟩

lemma *im-set-mono*: $[\![f \in A \to B;\ A1 \subseteq A2;\ A2 \subseteq A\ ]\!] \Longrightarrow (f\ `\ A1) \subseteq (f\ `\ A2)$
⟨*proof*⟩

lemma *im-set-un*: $[\![\ f \in A \to B;\ A1 \subseteq A;\ A2 \subseteq A\ ]\!] \Longrightarrow$
$f`(A1 \cup A2) = (f`A1) \cup (f`A2)$
⟨*proof*⟩

lemma *im-set-un1*: $[\![\forall l \in A.\ f\ l \in B;\ A = A1 \cup A2]\!] \Longrightarrow$
$f\ `(A1 \cup A2) = f\ `(A1) \cup f\ `(A2)$
⟨*proof*⟩

lemma *im-set-un2*: $A = A1 \cup A2 \Longrightarrow\ f\ `A = f\ `(A1) \cup f\ `(A2)$
⟨*proof*⟩

definition
invim::$['a \Rightarrow 'b,\ 'a\ set,\ 'b\ set] \Rightarrow 'a\ set$ **where**
$invim\ f\ A\ B = \{x.\ x \in A \wedge f\ x \in B\}$

lemma *invim*: $[\![\ f{:}A \to B;\ B1 \subseteq B\ ]\!] \Longrightarrow invim\ f\ A\ B1 \subseteq A$
⟨*proof*⟩

lemma *setim-cmpfn*: $[\![\ f{:}A \to B;\ g{:}B \to C;\ A1 \subseteq A\ ]\!] \Longrightarrow$
$(compose\ A\ g\ f)`\ A1 = g`(f`\ A1)$
⟨*proof*⟩

definition
surj-to :: $['a \Rightarrow 'b,\ 'a\ set,\ 'b\ set] \Rightarrow bool$ **where**
$surj\text{-}to\ f\ A\ B \longleftrightarrow f`A = B$

lemma *surj-to-test*: $[\![\ f \in A \to B;\ \forall b \in B.\ \exists a \in A.\ f\ a = b\ ]\!] \Longrightarrow$
$surj\text{-}to\ f\ A\ B$
⟨*proof*⟩

lemma *surj-to-image*: $f \in A \to B \Longrightarrow surj\text{-}to\ f\ A\ (f\ `\ A)$
⟨*proof*⟩

lemma *surj-to-el*: $[\![\ f \in A \to B;\ surj\text{-}to\ f\ A\ B\ ]\!] \Longrightarrow \forall b \in B.\ \exists a \in A.\ f\ a = b$
⟨*proof*⟩

lemma *surj-to-el1*: $[\![\ f \in A \to B;\ surj\text{-}to\ f\ A\ B;\ b \in B]\!] \Longrightarrow \exists a \in A.\ f\ a = b$
⟨*proof*⟩

lemma *surj-to-el2*:$[\![surj\text{-}to\ f\ A\ B;\ b \in B]\!] \Longrightarrow \exists\, a{\in}A.\ f\ a = b$
⟨*proof*⟩

lemma *compose-surj*: $[\![f{:}A \to B;\ surj\text{-}to\ f\ A\ B;\ g : B \to C;\ surj\text{-}to\ g\ B\ C\ ]\!]$
$\Longrightarrow surj\text{-}to\ (compose\ A\ g\ f)\ A\ C$
⟨*proof*⟩

lemma *cmp-surj*: $[\![f{:}A \to B;\ surj\text{-}to\ f\ A\ B;\ g : B \to C;\ surj\text{-}to\ g\ B\ C\ ]\!]$
$\Longrightarrow surj\text{-}to\ (cmp\ g\ f)\ A\ C$
⟨*proof*⟩

lemma *inj-onTr0*:$[\![\ f \in A \to B;\ x \in A;\ y \in A;\ inj\text{-}on\ f\ A;\ f\ x = f\ y]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *inj-onTr1*:$[\![inj\text{-}on\ f\ A;\ x \in A;\ y \in A;\ f\ x = f\ y]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *inj-onTr2*:$[\![inj\text{-}on\ f\ A;\ x \in A;\ y \in A;\ f\ x \neq f\ y]\!] \Longrightarrow x \neq y$
⟨*proof*⟩

lemma *comp-inj*: $[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ g \in B \to C;\ inj\text{-}on\ g\ B\ ]\!]$
$\Longrightarrow inj\text{-}on\ (compose\ A\ g\ f)\ A$
⟨*proof*⟩

lemma *cmp-inj-1*: $[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ g \in B \to C;\ inj\text{-}on\ g\ B\ ]\!]$
$\Longrightarrow inj\text{-}on\ (cmp\ g\ f)\ A$
⟨*proof*⟩

lemma *cmp-inj-2*: $[\![\forall\, l{\in}A.\ f\ l \in B;\ inj\text{-}on\ f\ A;\ \forall\, k{\in}B.\ g\ k \in C;\ inj\text{-}on\ g\ B\ ]\!]$
$\Longrightarrow inj\text{-}on\ (cmp\ g\ f)\ A$
⟨*proof*⟩

lemma *invfun-mem*:$[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B;\ b \in B\ ]\!]$
$\Longrightarrow\ (invfun\ A\ B\ f)\ b \in A$
⟨*proof*⟩

lemma *inv-func*:$[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B]\!]$
$\Longrightarrow\ (invfun\ A\ B\ f) \in B \to A$
⟨*proof*⟩

lemma *invfun-r*:$[\![\ f{\in}A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B;\ b \in B\ ]\!]$
$\Longrightarrow f\ ((invfun\ A\ B\ f)\ b) = b$
⟨*proof*⟩

lemma *invfun-l*:$[\![f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B;\ a \in A]\!]$
$\Longrightarrow (invfun\ A\ B\ f)\ (f\ a) = a$
⟨*proof*⟩

lemma *invfun-inj*: $[\![f \in A \to B; \textit{inj-on } f\ A; \textit{surj-to } f\ A\ B]\!]$
$\Longrightarrow \textit{inj-on } (\textit{invfun } A\ B\ f)\ B$
⟨*proof*⟩

lemma *invfun-surj*: $[\![f \in A \to B; \textit{inj-on } f\ A; \textit{surj-to } f\ A\ B]\!]$
$\Longrightarrow \textit{surj-to } (\textit{invfun } A\ B\ f)\ B\ A$
⟨*proof*⟩

definition
bij-to :: $['a \Rightarrow 'b, 'a\ set, 'b\ set] \Rightarrow bool$ **where**
$\textit{bij-to } f\ A\ B \longleftrightarrow \textit{surj-to } f\ A\ B \wedge \textit{inj-on } f\ A$

lemma *idmap-bij*: $\textit{bij-to } (\textit{idmap } A)\ A\ A$
⟨*proof*⟩

lemma *bij-invfun*: $[\![f \in A \to B; \textit{bij-to } f\ A\ B]\!] \Longrightarrow$
$\textit{bij-to } (\textit{invfun } A\ B\ f)\ B\ A$
⟨*proof*⟩

lemma *l-inv-invfun*: $[\![f \in A \to B; \textit{inj-on } f\ A; \textit{surj-to } f\ A\ B]\!]$
$\Longrightarrow \textit{compose } A\ (\textit{invfun } A\ B\ f)\ f = \textit{idmap } A$
⟨*proof*⟩

lemma *invfun-mem1*: $[\![f \in A \to B; \textit{bij-to } f\ A\ B; b \in B]\!] \Longrightarrow$
$(\textit{invfun } A\ B\ f)\ b \in A$
⟨*proof*⟩

lemma *invfun-r1*: $[\![f{\in}A \to B; \textit{bij-to } f\ A\ B; b \in B\ ]\!]$
$\Longrightarrow f\ ((\textit{invfun } A\ B\ f)\ b) = b$
⟨*proof*⟩

lemma *invfun-l1*: $[\![f \in A \to B; \textit{bij-to } f\ A\ B; a \in A]\!]$
$\Longrightarrow (\textit{invfun } A\ B\ f)\ (f\ a) = a$
⟨*proof*⟩

lemma *compos-invfun-r*: $[\![f \in A \to B; \textit{bij-to } f\ A\ B; g \in A \to C; h \in B \to C;$
$g \in \textit{extensional } A; \textit{compose } B\ g\ (\textit{invfun } A\ B\ f) = h]\!] \Longrightarrow$
$g = \textit{compose } A\ h\ f$
⟨*proof*⟩

lemma *compos-invfun-l*: $[\![f \in A \to B; \textit{bij-to } f\ A\ B; g \in C \to B; h \in C \to A;$
$\textit{compose } C\ (\textit{invfun } A\ B\ f)\ g = h; g \in \textit{extensional } C\ ]\!] \Longrightarrow$
$g = \textit{compose } C\ f\ h$
⟨*proof*⟩

lemma *invfun-set*: $[\![f \in A \to B; \textit{bij-to } f\ A\ B; C \subseteq B]\!] \Longrightarrow$
$f\ `\ ((\textit{invfun } A\ B\ f)`\ C) = C$
⟨*proof*⟩

lemma *compos-bij*:$[\![f \in A \to B;\ \textit{bij-to}\ f\ A\ B;\ g \in B \to C;\ \textit{bij-to}\ g\ B\ C]\!] \Longrightarrow$
$\textit{bij-to}\ (\textit{compose}\ A\ g\ f)\ A\ C$
⟨*proof*⟩

1.5 Nsets

definition
$\textit{nset} :: [\textit{nat}, \textit{nat}] \Rightarrow (\textit{nat})\ \textit{set}$ **where**
$\textit{nset}\ i\ j = \{k.\ i \le k \land k \le j\}$

definition
$\textit{slide} :: \textit{nat} \Rightarrow \textit{nat} \Rightarrow \textit{nat}$ **where**
$\textit{slide}\ i\ j == i + j$

definition
$\textit{sliden} :: \textit{nat} \Rightarrow \textit{nat} \Rightarrow \textit{nat}$ **where**
$\textit{sliden}\ i\ j == j - i$

definition
$\textit{jointfun} :: [\textit{nat}, \textit{nat} \Rightarrow 'a, \textit{nat}, \textit{nat} \Rightarrow 'a] \Rightarrow (\textit{nat} \Rightarrow 'a)$ **where**
$(\textit{jointfun}\ n\ f\ m\ g) = (\lambda i.\ \textit{if}\ i \le n\ \textit{then}\ f\ i\ \textit{else}\ \ g\ ((\textit{sliden}\ (\textit{Suc}\ n))\ i))$

definition
$\textit{skip} :: \textit{nat} \Rightarrow (\textit{nat} \Rightarrow \textit{nat})$ **where**
$\textit{skip}\ i = (\lambda x.\ (\textit{if}\ i = 0\ \textit{then}\ \textit{Suc}\ x\ \textit{else}$
$(\textit{if}\ x \in \{j.\ j \le (i - \textit{Suc}\ 0)\}\ \textit{then}\ x\ \ \textit{else}\ \textit{Suc}\ x)))$

lemma *nat-pos*:$0 \le (l::\textit{nat})$
⟨*proof*⟩

lemma *Suc-pos*:$\textit{Suc}\ k \le r \Longrightarrow 0 < r$
⟨*proof*⟩

lemma *nat-pos2*:$(k::\textit{nat}) < r \Longrightarrow 0 < r$
⟨*proof*⟩

lemma *eq-le-not*:$[\![(a::\textit{nat}) \le b;\ \neg\ a < b\ ]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma *im-of-constmap*:$(\textit{constmap}\ \{0\}\ \{a\})\ `\ \{0\} = \{a\}$
⟨*proof*⟩

lemma *noteq-le-less*:$[\![\ m \le (n::\textit{nat});\ m \ne n\ ]\!] \Longrightarrow m < n$
⟨*proof*⟩

lemma *nat-not-le-less*:$(\neg\ (n::\textit{nat}) \le m) = (m < n)$
⟨*proof*⟩

lemma *self-le*:$(n::nat) \leq n$
⟨*proof*⟩

lemma *n-less-Suc*:$(n::nat) < Suc\ n$
⟨*proof*⟩

lemma *less-diff-pos*:$i < (n::nat) \Longrightarrow 0 < n - i$
⟨*proof*⟩

lemma *less-diff-Suc*:$i < (n::nat) \Longrightarrow n - (Suc\ i) = (n - i) - (Suc\ 0)$
⟨*proof*⟩

lemma *less-pre-n*:$0 < n \Longrightarrow n - Suc\ 0 < n$
⟨*proof*⟩

lemma *Nset-inc-0*:$(0::nat) \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-1*:$\{i.\ i \leq Suc\ 0\} = \{0,\ Suc\ 0\}$
⟨*proof*⟩

lemma *Nset-1-1*:$(k \leq Suc\ 0) = (k = 0 \vee k = Suc\ 0)$
⟨*proof*⟩

lemma *Nset-2*:$\{i,\ j\} = \{j,\ i\}$
⟨*proof*⟩

lemma *Nset-nonempty*:$\{i.\ i \leq (n::nat)\} \neq \{\}$
⟨*proof*⟩

lemma *Nset-le*:$x \in \{i.\ i \leq n\} \Longrightarrow x \leq n$
⟨*proof*⟩

lemma *n-in-Nsetn*:$(n::nat) \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-pre*:$[\![\ (x::nat) \in \{i.\ i \leq (Suc\ n)\};\ x \neq Suc\ n\ ]\!] \Longrightarrow x \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-pre1*:$\{i.\ i \leq (Suc\ n)\} - \{Suc\ n\} = \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *le-Suc-mem-Nsetn*:$x \leq Suc\ n \Longrightarrow x - Suc\ 0 \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *le-Suc-diff-le*:$x \leq Suc\ n \Longrightarrow x - Suc\ 0 \leq n$
⟨*proof*⟩

lemma *Nset-not-pre*:$[\![\ x \notin \{i.\ i \leq n\};\ x \in \{i.\ i \leq (Suc\ n)\}]\!] \Longrightarrow x = Suc\ n$

⟨proof⟩

lemma *mem-of-Nset*: $x \leq (n::nat) \Longrightarrow x \in \{i.\ i \leq n\}$
⟨proof⟩

lemma *less-mem-of-Nset*: $x < (n::nat) \Longrightarrow x \in \{i.\ i \leq n\}$
⟨proof⟩

lemma *Nset-nset*: $\{i.\ i \leq (Suc\ (n + m))\} = \{i.\ i \leq n\} \cup$
$nset\ (Suc\ n)\ (Suc\ (n + m))$
⟨proof⟩

lemma *Nset-nset-1*: $[\![0 < n;\ i < n]\!] \Longrightarrow \{j.\ j \leq n\} = \{j.\ j \leq i\} \cup$
$nset\ (Suc\ i)\ n$
⟨proof⟩

lemma *Nset-img0*: $[\![f \in \{j.\ j \leq Suc\ n\} \to B;\ (f\ (Suc\ n)) \in f\ `\ \{j.\ j \leq n\}]\!] \Longrightarrow$
$f\ `\ \{j.\ j \leq Suc\ n\} = f\ `\ \{j.\ j \leq n\}$
⟨proof⟩

lemma *Nset-img*: $f \in \{j.\ j \leq Suc\ n\} \to B \Longrightarrow$
$insert\ (f\ (Suc\ n))\ (f\ `\ \{j.\ j \leq n\}) = f\ `\ \{j.\ j \leq Suc\ n\}$
⟨proof⟩

primrec *nasc-seq* :: $[nat\ set,\ nat,\ nat] \Rightarrow nat$
where
dec-seq-0: $nasc\text{-}seq\ A\ a\ 0 = a$
| *dec-seq-Suc*: $nasc\text{-}seq\ A\ a\ (Suc\ n) =$
$(SOME\ b.\ ((b \in A) \wedge (nasc\text{-}seq\ A\ a\ n) < b))$

lemma *nasc-seq-mem*: $[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m \in A \wedge (\forall\, x \in A.\ x \leq m))]\!] \Longrightarrow$
$(nasc\text{-}seq\ A\ a\ n) \in A$
⟨proof⟩

lemma *nasc-seqn*: $[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m \in A \wedge (\forall\, x \in A.\ x \leq m))]\!] \Longrightarrow$
$(nasc\text{-}seq\ A\ a\ n) < (nasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨proof⟩

lemma *nasc-seqn1*: $[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m \in A \wedge (\forall\, x \in A.\ x \leq m))]\!] \Longrightarrow$
$Suc\ (nasc\text{-}seq\ A\ a\ n) \leq (nasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨proof⟩

lemma *ubs-ex-n-maxTr*: $[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m \in A \wedge (\forall\, x \in A.\ x \leq m))]\!]$
$\Longrightarrow\ (a + n) \leq (nasc\text{-}seq\ A\ a\ n)$
⟨proof⟩

lemma *ubs-ex-n-max*: $[\![A \neq \{\};\ A \subseteq \{i.\ i \leq (n::nat)\}]\!] \Longrightarrow$
$\exists!m.\ m \in A \wedge (\forall\, x \in A.\ x \leq m)$
⟨proof⟩

definition
  n-max :: *nat set* ⇒ *nat* **where**
  $n\text{-}max\ A = (THE\ m.\ m \in A \wedge (\forall x{\in}A.\ x \leq m))$

lemma *n-max*: $[\![A \subseteq \{i.\ i \leq (n{::}nat)\};\ A \neq \{\}]\!] \Longrightarrow$
$(n\text{-}max\ A) \in A \wedge (\forall x{\in}A.\ x \leq (n\text{-}max\ A))$
⟨*proof*⟩

lemma *n-max-eq-sets*: $[\![A = B;\ A \neq \{\};\ \exists n.\ A \subseteq \{j.\ j \leq n\}]\!] \Longrightarrow$
$n\text{-}max\ A = n\text{-}max\ B$
⟨*proof*⟩

lemma *skip-mem*: $l \in \{i.\ i \leq n\} \Longrightarrow (skip\ i\ l) \in \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *skip-fun*: $(skip\ i) \in \{i.\ i \leq n\} \rightarrow \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *skip-im-Tr0*: $x \in \{i.\ i \leq n\} \Longrightarrow skip\ 0\ x = Suc\ x$
⟨*proof*⟩

lemma *skip-im-Tr0-1*: $0 < y \Longrightarrow skip\ 0\ (y - Suc\ 0) = y$
⟨*proof*⟩

lemma *skip-im-Tr1*: $[\![\ i \in \{i.\ i \leq (Suc\ n)\};\ 0 < i;\ x \leq i - Suc\ 0\ ]\!] \Longrightarrow$
$skip\ i\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr1-1*: $[\![\ i \in \{i.\ i \leq (Suc\ n)\};\ 0 < i;\ x < i]\!] \Longrightarrow$
$skip\ i\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr1-2*: $[\![\ i \leq (Suc\ n);\ x < i]\!] \Longrightarrow\ skip\ i\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr2*: $[\![\ 0 < i;\ i \in \{i.\ i \leq (Suc\ n)\};\ i \leq x]\!] \Longrightarrow$
$skip\ i\ x = Suc\ x$
⟨*proof*⟩

lemma *skip-im-Tr2-1*: $[\![i \in \{i.\ i \leq (Suc\ n)\};\ i \leq x]\!] \Longrightarrow$
$skip\ i\ x = Suc\ x$
⟨*proof*⟩

lemma *skip-im-Tr3*: $x \in \{i.\ i \leq n\} \Longrightarrow skip\ (Suc\ n)\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr4*: $[\![x \leq Suc\ n;\ 0 < x]\!] \Longrightarrow x - Suc\ 0 \leq n$

⟨*proof*⟩

lemma *skip-fun-im*:$i \in \{j.\ j \le (Suc\ n)\} \Longrightarrow$
$(skip\ i)\ `\ \{j.\ j \le n\} = (\{j.\ j \le (Suc\ n)\} - \{i\})$
⟨*proof*⟩

lemma *skip-fun-im1*:$[\![i \in \{j.\ j \le (Suc\ n)\};\ x \in \{j.\ j \le n\}]\!] \Longrightarrow$
$(skip\ i)\ x \in (\{j.\ j \le (Suc\ n)\} - \{i\})$
⟨*proof*⟩

lemma *skip-id*:$l < i \Longrightarrow skip\ i\ l = l$
⟨*proof*⟩

lemma *Suc-neq*:$[\![0 < i;\ i - Suc\ 0 < l]\!] \Longrightarrow i \ne Suc\ l$
⟨*proof*⟩

lemma *skip-il-neq-i*:$skip\ i\ l \ne i$
⟨*proof*⟩

lemma *skip-inj*:$[\![i \in \{k.\ k \le n\};\ j \in \{k.\ k \le n\};\ i \ne j]\!] \Longrightarrow$
$skip\ k\ i \ne skip\ k\ j$
⟨*proof*⟩

lemma *le-imp-add-int*: $i \le (j::nat) \Longrightarrow \exists k.\ j = i + k$
⟨*proof*⟩

lemma *jointfun-hom0*:$[\![\ f \in \{j.\ j \le n\} \to A;\ g \in \{k.\ k \le m\} \to B\ ]\!] \Longrightarrow$
$(jointfun\ n\ f\ m\ g) \in \{l.\ l \le (Suc\ (n + m))\} \to\ (A \cup B)$
⟨*proof*⟩

lemma *jointfun-mem*:$[\![\forall j \le (n::nat).\ f\ j \in A;\ \forall j \le m.\ g\ j \in B;$
$l \le (Suc\ (n + m))]\!] \Longrightarrow (jointfun\ n\ f\ m\ g)\ l \in (A \cup B)$
⟨*proof*⟩

lemma *jointfun-inj*:$[\![f \in \{j.\ j \le n\} \to B;\ inj\text{-}on\ f\ \{j.\ j \le n\};$
$b \notin f\ `\ \{j.\ j \le n\}]\!] \Longrightarrow$
$inj\text{-}on\ (jointfun\ n\ f\ 0\ (\lambda k \in \{0::nat\}.\ b))\ \{j.\ j \le Suc\ n\}$
⟨*proof*⟩

lemma *slide-hom*:$i \le j \Longrightarrow (slide\ i) \in \{l.\ l \le (j - i)\} \to nset\ i\ j$
⟨*proof*⟩

lemma *slide-mem*:$[\![\ i \le j;\ l \in \{k.\ k \le (j - i)\}]\!] \Longrightarrow slide\ i\ l \in nset\ i\ j$
⟨*proof*⟩

lemma *slide-iM*:$(slide\ i)\ `\ \{l.\ 0 \le l\} = \{k.\ i \le k\}$
⟨*proof*⟩

lemma *jointfun-hom*:$[\![\ f \in \{i.\ i \le n\} \to A;\ g \in \{j.\ j \le m\} \to B\ ]\!] \Longrightarrow$

$(jointfun\ n\ f\ m\ g) \in \{j.\ j \leq (Suc\ (n + m))\} \to A \cup B$
⟨proof⟩

lemma *im-jointfunTr1*:$(jointfun\ n\ f\ m\ g)$ ‘ $\{i.\ i \leq n\} = f$ ‘ $\{i.\ i \leq n\}$
⟨proof⟩

lemma *im-jointfunTr2*:$(jointfun\ n\ f\ m\ g)$ ‘ $(nset\ (Suc\ n)\ (Suc\ (n + m))) =$
g ‘ $(\{j.\ j \leq m\})$
⟨proof⟩

lemma *im-jointfun*:$[\![f \in \{j.\ j \leq n\} \to A;\ g \in \{j.\ j \leq m\} \to B]\!] \Longrightarrow$
$(jointfun\ n\ f\ m\ g)$ ‘$(\{j.\ j \leq (Suc\ (n + m))\}) =$
f ‘$\{j.\ j \leq n\} \cup g$ ‘$\{j.\ j \leq m\}$
⟨proof⟩

lemma *im-jointfun1*:$(jointfun\ n\ f\ m\ g)$ ‘$(\{j.\ j \leq (Suc\ (n + m))\}) =$
f ‘$\{j.\ j \leq n\} \cup g$ ‘ $\{j.\ j \leq m\}$
⟨proof⟩

lemma *jointfun-surj*:$[\![f \in \{j.\ j \leq n\} \to A;\ surj\text{-}to\ f\ \{j.\ j \leq (n::nat)\}\ A;$
$g \in \{j.\ j \leq (m::nat)\} \to B;\ surj\text{-}to\ g\ \{j.\ j \leq m\}\ B]\!] \Longrightarrow$
$surj\text{-}to\ (jointfun\ n\ f\ m\ g)\ \{j.\ j \leq Suc\ (n + m)\}\ (A \cup B)$
⟨proof⟩

lemma *Nset-un*:$\{j.\ j \leq (Suc\ n)\} = \{j.\ j \leq n\} \cup \{Suc\ n\}$
⟨proof⟩

lemma *Nsetn-sub*: $\{j.\ j \leq n\} \subseteq \{j.\ j \leq (Suc\ n)\}$
⟨proof⟩

lemma *Nset-pre-sub*:$(0::nat) < k \Longrightarrow \{j.\ j \leq (k - Suc\ 0)\} \subseteq \{j.\ j \leq k\}$
⟨proof⟩

lemma *Nset-pre-un*:$(0::nat) < k \Longrightarrow \{j.\ j \leq k\} = \{j.\ j \leq (k - Suc\ 0)\} \cup \{k\}$
⟨proof⟩

lemma *Nsetn-sub-mem*: $l \in \{j.\ j \leq n\} \Longrightarrow l \in \{j.\ j \leq (Suc\ n)\}$
⟨proof⟩

lemma *Nsetn-sub-mem1*:$\forall j.\ j \in \{j.\ j \leq n\} \longrightarrow j \in \{j.\ j \leq (Suc\ n)\}$
⟨proof⟩

lemma *Nset-Suc*:$\{j.\ j \leq (Suc\ n)\} = insert\ (Suc\ n)\ \{j.\ j \leq n\}$
⟨proof⟩

lemma *nsetnm-sub-mem*:$\forall j.\ j \in nset\ n\ (n + m) \longrightarrow j \in nset\ n\ (Suc\ (n + m))$
⟨proof⟩

lemma *Nset-0*:$\{j.\ j \leq (0::nat)\} = \{0\}$

⟨proof⟩

lemma *Nset-Suc0*:$\{i.\ i \le (Suc\ 0)\} = \{0,\ Suc\ 0\}$
⟨proof⟩

lemma *Nset-Suc-Suc*:$Suc\ (Suc\ 0) \le n \Longrightarrow$
$\{j.\ j \le (n - Suc\ (Suc\ 0))\} = \{j.\ j \le n\} - \{n - Suc\ 0,\ n\}$
⟨proof⟩

lemma *func-pre*:$f \in \{j.\ j \le (Suc\ n)\} \to A \Longrightarrow f \in \{j.\ j \le n\} \to A$
⟨proof⟩

lemma *image-Nset-Suc*:$f\ `\ (\{j.\ j \le (Suc\ n)\}) =$
$insert\ (f\ (Suc\ n))\ (f\ `\ \{j.\ j \le n\})$
⟨proof⟩

definition
Nleast :: *nat set* ⇒ *nat* **where**
$Nleast\ A = (THE\ a.\ (a \in A \wedge (\forall x \in A.\ a \le x)))$

definition
Nlb :: [*nat set*, *nat*] ⇒ *bool* **where**
$Nlb\ A\ n \longleftrightarrow (\forall a \in A.\ n \le a)$

primrec *ndec-seq* :: [*nat set*, *nat*, *nat*] ⇒ *nat* **where**
ndec-seq-0 :$ndec\text{-}seq\ A\ a\ 0 = a$
| *ndec-seq-Suc*:$ndec\text{-}seq\ A\ a\ (Suc\ n) =$
$(SOME\ b.\ ((b \in A) \wedge b < (ndec\text{-}seq\ A\ a\ n)))$

lemma *ndec-seq-mem*:$[\![a \in (A::nat\ set);\ \neg\ (\exists m.\ m \in A \wedge (\forall x \in A.\ m \le x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ n) \in A$
⟨proof⟩

lemma *ndec-seqn*:$[\![a \in (A::nat\ set);\neg\ (\exists m.\ m \in A \wedge (\forall x \in A.\ m \le x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ (Suc\ n)) < (ndec\text{-}seq\ A\ a\ n)$
⟨proof⟩

lemma *ndec-seqn1*:$[\![a \in (A::nat\ set);\ \neg\ (\exists m.\ m \in A \wedge (\forall x \in A.\ m \le x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ (Suc\ n)) \le (ndec\text{-}seq\ A\ a\ n) - 1$
⟨proof⟩

lemma *ex-NleastTr*:$[\![a \in (A::nat\ set);\ \neg\ (\exists m.\ m \in A \wedge (\forall x \in A.\ m \le x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ n) \le (a - n)$
⟨proof⟩

lemma *nat-le*:$((a::nat) - (a + 1)) \le 0$
⟨proof⟩

lemma *ex-Nleast*:$(A::nat\ set) \neq \{\} \Longrightarrow \exists!m.\ m \in A \wedge (\forall x \in A.\ m \leq x)$
⟨*proof*⟩

lemma *Nleast*:$(A::nat\ set) \neq \{\} \Longrightarrow Nleast\ A \in A \wedge (\forall x \in A.\ (Nleast\ A) \leq x)$
⟨*proof*⟩

1.5.1 Lemmas for existence of reduced chain.

lemma *jointgd-tool1*: $0 < i \Longrightarrow 0 \leq i - Suc\ 0$
⟨*proof*⟩

lemma *jointgd-tool2*: $0 < i \Longrightarrow i = Suc\ (i - Suc\ 0)$
⟨*proof*⟩

lemma *jointgd-tool3*:$[\![0 < i;\ \ i \leq m]\!] \Longrightarrow i - Suc\ 0 \leq (m - Suc\ 0)$
⟨*proof*⟩

lemma *jointgd-tool4*:$n < i \Longrightarrow i - n = Suc(\ i - Suc\ n)$
⟨*proof*⟩

lemma *pos-prec-less*:$0 < i \Longrightarrow i - Suc\ 0 < i$
⟨*proof*⟩

lemma *Un-less-Un*:$[\![f \in \{j.\ j \leq (Suc\ n)\} \rightarrow (X::'a\ set\ set);$
$A \subseteq \bigcup (f\ `\ \{j.\ j \leq (Suc\ n)\});$
$i \in \{j.\ j \leq (Suc\ n)\};\ j \in \{l.\ l \leq (Suc\ n)\};\ i \neq j \wedge f\ i \subseteq f\ j]\!]$
$\Longrightarrow A \subseteq \bigcup (compose\ \{j.\ j \leq n\}\ f\ (skip\ i)\ `\ \{j.\ j \leq n\})$
⟨*proof*⟩

1.6 Lower bounded set of integers

definition $Zset = \{x.\ \exists (n::int).\ x = n\}$

definition
$Zleast :: int\ set \Rightarrow int$ **where**
$Zleast\ A = (THE\ a.\ (a \in A \wedge (\forall x \in A.\ a \leq x)))$

definition
$LB :: [int\ set,\ int] \Rightarrow bool$ **where**
$LB\ A\ n = (\forall a \in A.\ n \leq a)$

lemma *linorder-linear1*:$(m::int) < n \vee n \leq m$
⟨*proof*⟩

primrec *dec-seq* $:: [int\ set,\ int,\ nat] \Rightarrow int$
where
dec-seq-0: $dec\text{-}seq\ A\ a\ 0 = a$
| *dec-seq-Suc*: $dec\text{-}seq\ A\ a\ (Suc\ n) = (SOME\ b.\ ((b \in A) \wedge b < (dec\text{-}seq\ A\ a\ n)))$

lemma *dec-seq-mem*:$[\![a \in A;\ A \subseteq Zset; \neg (\exists m.\ m \in A \wedge (\forall x \in A.\ m \leq x))]\!] \Longrightarrow$
$(dec\text{-}seq\ A\ a\ n) \in A$
⟨*proof*⟩

lemma *dec-seqn*:$[\![a \in A;\ A \subseteq Zset; \neg (\exists m.\ m \in A \wedge (\forall x \in A.\ m \leq x))]\!] \Longrightarrow$
$(dec\text{-}seq\ A\ a\ (Suc\ n)) < (dec\text{-}seq\ A\ a\ n)$
⟨*proof*⟩

lemma *dec-seqn1*:$[\![a \in A;\ A \subseteq Zset; \neg (\exists m.\ m \in A \wedge (\forall x \in A.\ m \leq x))]\!] \Longrightarrow$
$(dec\text{-}seq\ A\ a\ (Suc\ n)) \leq (dec\text{-}seq\ A\ a\ n) - 1$
⟨*proof*⟩

lemma *lbs-ex-ZleastTr*:$[\![a \in A;\ A \subseteq Zset; \neg (\exists m.\ m \in A \wedge (\forall x \in A.\ m \leq x))]\!] \Longrightarrow$
$(dec\text{-}seq\ A\ a\ n) \leq (a - int(n))$
⟨*proof*⟩

lemma *big-int-less*:$a - int(nat(abs(a) + abs(N) + 1)) < N$
⟨*proof*⟩

lemma *lbs-ex-Zleast*:$[\![A \neq \{\};\ A \subseteq Zset;\ LB\ A\ n]\!] \Longrightarrow \exists !m.\ m \in A \wedge (\forall x \in A.\ m \leq x)$
⟨*proof*⟩

lemma *Zleast*:$[\![A \neq \{\};\ A \subseteq Zset;\ LB\ A\ n]\!] \Longrightarrow Zleast\ A \in A \wedge$
$(\forall x \in A.\ (Zleast\ A) \leq x)$
⟨*proof*⟩

lemma *less-convert1*:$[\![\ a = c;\ a < b\ ]\!] \Longrightarrow c < b$
⟨*proof*⟩

lemma *less-convert2*:$[\![a = b;\ b < c]\!] \Longrightarrow a < c$
⟨*proof*⟩

1.7 Augmented integer: integer and $\infty - \infty$

definition
zag :: $(int * int)$ *set* **where**
$zag = \{(x,y) \mid x\ y.\ x * y = (0::int) \wedge (y = -1 \vee y = 0 \vee y = 1)\}$

definition
zag-pl::$[(int * int), (int * int)] \Rightarrow (int * int)$ **where**
$zag\text{-}pl\ x\ y == if\ (snd\ x + snd\ y) = 2\ then\ (0,\ 1)$
$else\ if\ (snd\ x + snd\ y) = 1\ then\ (0,\ 1)$
$else\ if\ (snd\ x + snd\ y) = 0\ then\ (fst\ x + fst\ y,\ 0)$
$else\ if\ (snd\ x + snd\ y) = -1\ then\ (0,\ -1)$
$else\ if\ (snd\ x + snd\ y) = -2\ then\ (0,\ -1)\ else\ undefined$

definition
zag-t :: $[(int * int), (int * int)] \Rightarrow (int * int)$ **where**

zag-t x y = (*if* (*snd x*)∗(*snd y*) = *0* *then*
(*if 0* < (*fst x*)∗(*snd y*) + (*snd x*)∗(*fst y*) *then* (*0*,*1*)
else (*if* (*fst x*)∗(*snd y*) + (*snd x*)∗(*fst y*) = *0*
then ((*fst x*)∗(*fst y*), *0*) *else* (*0*, −*1*)))
else (*if 0* < (*snd x*)∗(*snd y*) *then* (*0*, *1*) *else* (*0*, −*1*)))

definition *Ainteg* = *zag*

typedef *ant* = *Ainteg*
morphisms *Rep-Ainteg Abs-Ainteg*
⟨*proof*⟩

definition
ant :: *int* ⇒ *ant* **where**
ant m = *Abs-Ainteg*((*m*, *0*))

definition
tna :: *ant* ⇒ *int* **where**
tna z = (*if Rep-Ainteg*(*z*) ≠ (*0*,*1*) ∧ *Rep-Ainteg*(*z*) ≠ (*0*,−*1*) *then*
fst (*Rep-Ainteg*(*z*)) *else undefined*)

instantiation *ant* :: {*zero*, *one*, *plus*, *uminus*, *minus*, *times*, *ord*}
begin

definition
Zero-ant-def : *0* == *ant 0*

definition
One-ant-def : *1* == *ant 1*

definition
add-ant-def:
z + *w* ==
Abs-Ainteg (*zag-pl* (*Rep-Ainteg z*) (*Rep-Ainteg w*))

definition
minus-ant-def : − *z* ==
Abs-Ainteg((− (*fst* (*Rep-Ainteg z*)), − (*snd* (*Rep-Ainteg z*))))

definition
diff-ant-def: *z* − (*w*::*ant*) == *z* + (−*w*)

definition
mult-ant-def:
z ∗ *w* ==
Abs-Ainteg (*zag-t* (*Rep-Ainteg z*) (*Rep-Ainteg w*))

definition
le-ant-def:

$(z::ant) \leq w == if\ (snd\ (Rep\text{-}Ainteg\ w)) = 1\ then\ True$
$else\ (if\ (snd\ (Rep\text{-}Ainteg\ w)) = 0\ then\ (if\ (snd\ (Rep\text{-}Ainteg\ z)) = 1$
$then\ False\ else\ (if\ (snd\ (Rep\text{-}Ainteg\ z)) = 0\ then$
$(fst\ (Rep\text{-}Ainteg\ z)) \leq (fst\ (Rep\text{-}Ainteg\ w))\ \ else\ True))$
$else\ (if\ snd\ (Rep\text{-}Ainteg\ z) = -1\ then\ True\ else\ False))$

definition
less-ant-def: $((z::ant) < (w::ant)) == (z \leq w \wedge z \neq w)$

instance ⟨*proof*⟩

end

definition
inf-ant :: *ant* (∞) **where**
$\infty = Abs\text{-}Ainteg((0,1))$

definition
an :: *nat* ⇒ *ant* **where**
an m = *ant* (*int m*)

definition
na :: *ant* ⇒ *nat* **where**
$na\ x = (if\ (x < 0)\ then\ 0\ else$
$if\ x \neq \infty\ then\ (nat\ (tna\ x))\ else\ undefined)$

definition
UBset :: *ant* ⇒ *ant set* **where**
$UBset\ z = \{(x::ant).\ \ x \leq z\}$

definition
LBset :: *ant* ⇒ *ant set* **where**
$LBset\ z = \{(x::ant).\ z \leq x\}$

lemma *ant-z-in-Ainteg*:$(z::int,\ 0) \in Ainteg$
⟨*proof*⟩

lemma *ant-inf-in-Ainteg*:$((0::int),\ 1) \in Ainteg$
⟨*proof*⟩

lemma *ant-minf-in-Ainteg*:$((0::int),\ -1) \in Ainteg$
⟨*proof*⟩

lemma *ant-0-in-Ainteg*:$((0::int),\ 0) \in Ainteg$
⟨*proof*⟩

lemma *an-0*[*simp*]:$an\ 0 = 0$
⟨*proof*⟩

lemma *an-1*[*simp*]:$an\ 1 = 1$
⟨*proof*⟩

lemma *mem-ant*:$(a::ant) = -\infty \lor (\exists (z::int).\ a = ant\ z) \lor a = \infty$
⟨*proof*⟩

lemma *minf*:$-\infty = Abs\text{-}Ainteg((0,-1))$
⟨*proof*⟩

lemma *z-neq-inf*[*simp*]:$(ant\ z) \neq \infty$
⟨*proof*⟩

lemma *z-neq-minf*[*simp*]:$(ant\ z) \neq -\infty$
⟨*proof*⟩

lemma *minf-neq-inf*[*simp*]:$-\infty \neq \infty$
⟨*proof*⟩

lemma *a-ipi*[*simp*]:$\infty + \infty = \infty$
⟨*proof*⟩

lemma *a-zpi*[*simp*]:$(ant\ z) + \infty\ = \infty$
⟨*proof*⟩

lemma *a-ipz*[*simp*]: $\infty + (ant\ z) = \infty$
⟨*proof*⟩

lemma *a-zpz*:$(ant\ m) + (ant\ n) = ant\ (m + n)$
⟨*proof*⟩

lemma *a-mpi*[*simp*]:$-\infty + \infty\ = 0$
⟨*proof*⟩

lemma *a-ipm*[*simp*]:$\infty + (-\infty) = 0$
⟨*proof*⟩

lemma *a-mpm*[*simp*]:$-\infty + (-\infty) = -\infty$
⟨*proof*⟩

lemma *a-mpz*[*simp*]:$-\infty + (ant\ m) = -\infty$
⟨*proof*⟩

lemma *a-zpm*[*simp*]:$(ant\ m) + (-\infty) = -\infty$
⟨*proof*⟩

lemma *a-mdi*[*simp*]:$-\infty - \infty\ = -\ \infty$
⟨*proof*⟩

lemma *a-zdz*:$(ant\ m) - (ant\ n) = ant\ (m - n)$
⟨*proof*⟩

lemma *a-i-i*[*simp*]:$\infty * \infty = \infty$
⟨*proof*⟩

lemma *a-0-i*[*simp*]:$0 * \infty = 0$
⟨*proof*⟩

lemma *a-i-0*[*simp*]:$\infty * 0 = 0$
⟨*proof*⟩

lemma *a-0-m*[*simp*]:$0 * (-\infty) = 0$
⟨*proof*⟩

lemma *a-m-0*[*simp*]:$(-\infty) * 0 = 0$
⟨*proof*⟩

lemma *a-m-i*[*simp*]:$(-\infty) * \infty = -\infty$
⟨*proof*⟩

lemma *a-i-m*[*simp*]:$\infty * (-\infty) = - \infty$
⟨*proof*⟩

lemma *a-pos-i*[*simp*]:$0 < m \Longrightarrow (ant\ m) * \infty = \infty$
⟨*proof*⟩

lemma *a-i-pos*[*simp*]:$0 < m \Longrightarrow \infty * (ant\ m) = \infty$
⟨*proof*⟩

lemma *a-neg-i*[*simp*]:$m < 0 \Longrightarrow (ant\ m) * \infty = -\infty$
⟨*proof*⟩

lemma *a-i-neg*[*simp*]:$m < 0 \Longrightarrow \infty * (ant\ m) = -\infty$
⟨*proof*⟩

lemma *a-z-z*:$(ant\ m) * (ant\ n) = ant\ (m*n)$
⟨*proof*⟩

lemma *a-pos-m*[*simp*]:$0 < m \Longrightarrow (ant\ m) * (-\infty) = -\infty$
⟨*proof*⟩

lemma *a-m-pos*[*simp*]:$0 < m \Longrightarrow (-\infty) * (ant\ m) = -\infty$
⟨*proof*⟩

lemma *a-neg-m*[*simp*]:$m < 0 \Longrightarrow (ant\ m) * (-\infty) = \infty$
⟨*proof*⟩

lemma *neg-a-m*[*simp*]:$m < 0 \Longrightarrow (-\infty) * (ant\ m) = \infty$
⟨*proof*⟩

lemma *a-m-m*[*simp*]:$(-\infty) * (-\infty) = \infty$
⟨*proof*⟩

lemma *inj-on-Abs-Ainteg*:*inj-on Abs-Ainteg Ainteg*
⟨*proof*⟩

lemma *an-Suc*:$an\ (Suc\ n) = an\ n + 1$
⟨*proof*⟩

lemma *aeq-zeq* [*iff*]: $(ant\ m = ant\ n) = (m = n)$
⟨*proof*⟩

lemma *aminus*:$-\ ant\ m = ant\ (-m)$
⟨*proof*⟩

lemma *aminusZero*:$-\ ant\ 0 = ant\ 0$
⟨*proof*⟩

lemma *ant-0*: $ant\ 0 = (0::ant)$
⟨*proof*⟩

lemma *inf-neq-0*[*simp*]:$\infty \neq 0$
⟨*proof*⟩

lemma *zero-neq-inf*[*simp*]:$0 \neq \infty$
⟨*proof*⟩

lemma *minf-neq-0*[*simp*]:$-\infty \neq 0$
⟨*proof*⟩

lemma *zero-neq-minf*[*simp*]:$0 \neq -\infty$
⟨*proof*⟩

lemma *a-minus-zero*[*simp*]:$-(0::ant) = 0$
⟨*proof*⟩

lemma *a-minus-minus*: $-\ (-\ z) = (z::ant)$
⟨*proof*⟩

lemma *aminus-0*: $-\ (-\ 0) = (0::ant)$
⟨*proof*⟩

lemma *a-a-z-0*:$\llbracket\ 0 < z;\ a * ant\ z = 0\rrbracket \Longrightarrow a = 0$
⟨*proof*⟩

lemma *adiv-eq*:⟦ $z \neq 0$; $a * (ant\ z) = b * (ant\ z)$⟧ $\Longrightarrow a = b$
⟨*proof*⟩

lemma *aminus-add-distrib*: $-(z + w) = (-z) + (-w{::}ant)$
⟨*proof*⟩

lemma *aadd-commute*:$(x{::}ant) + y = y + x$
⟨*proof*⟩

definition
aug-inf :: *ant set* (Z_∞) **where**
$Z_\infty = \{(z{::}ant).\ z \neq -\infty\}$

definition
aug-minf :: *ant set* $(Z_{-\infty})$ **where**
$Z_{-\infty} = \{(z{::}ant).\ z \neq \infty\}$

lemma *z-in-aug-inf*:$ant\ z \in Z_\infty$
⟨*proof*⟩

lemma *Zero-in-aug-inf*:$0 \in Z_\infty$
⟨*proof*⟩

lemma *z-in-aug-minf*:$ant\ z \in Z_{-\infty}$
⟨*proof*⟩

lemma *mem-aug-minf*:$a \in Z_{-\infty} \Longrightarrow a = -\infty \vee (\exists z.\ a = ant\ z)$
⟨*proof*⟩

lemma *minus-an-in-aug-minf*: $-\ an\ n \in Z_{-\infty}$
⟨*proof*⟩

lemma *Zero-in-aug-minf*:$0 \in Z_{-\infty}$
⟨*proof*⟩

lemma *aadd-assoc-i*: ⟦$x \in Z_\infty$; $y \in Z_\infty$; $z \in Z_\infty$⟧ $\Longrightarrow (x + y) + z = x + (y + z)$
⟨*proof*⟩

lemma *aadd-assoc-m*: ⟦$x \in Z_{-\infty}$; $y \in Z_{-\infty}$; $z \in Z_{-\infty}$⟧ $\Longrightarrow$
$(x + y) + z = x + (y + z)$
⟨*proof*⟩

lemma *aadd-0-r*: $x + (0{::}ant) = x$
⟨*proof*⟩

lemma *aadd-0-l*: $(0{::}ant) + x = x$
⟨*proof*⟩

lemma *aadd-minus-inv*: $(-\ x) + x = (0::ant)$
⟨*proof*⟩

lemma *aadd-minus-r*: $x + (-\ x) = (0::ant)$
⟨*proof*⟩

lemma *ant-minus-inj*:$ant\ z \neq ant\ w \Longrightarrow -\ ant\ z \neq -\ ant\ w$
⟨*proof*⟩

lemma *aminus-mult-minus*: $(-\ (ant\ z)) * (ant\ w) = -\ ((ant\ z) * (ant\ w))$
⟨*proof*⟩

lemma *amult-commute*: $(x::ant) * y = y * x$
⟨*proof*⟩

lemma *z-le-i*[*simp*]:$(ant\ x) \leq \infty$
⟨*proof*⟩

lemma *z-less-i*[*simp*]:$(ant\ x) < \infty$
⟨*proof*⟩

lemma *m-le-z*:$-\infty \leq (ant\ x)$
⟨*proof*⟩

lemma *m-less-z*[*simp*]:$-\infty < (ant\ x)$
⟨*proof*⟩

lemma *noninf-mem-Z*:$[\![x \in Z_\infty;\ x \neq \infty]\!] \Longrightarrow \exists (z::int).\ x = ant\ z$
⟨*proof*⟩

lemma *z-mem-Z*:$ant\ z \in Z_\infty$
⟨*proof*⟩

lemma *inf-ge-any*[*simp*]:$x \leq \infty$
⟨*proof*⟩

lemma *zero-lt-inf*:$0 < \infty$
⟨*proof*⟩

lemma *minf-le-any*[*simp*]:$-\infty \leq x$
⟨*proof*⟩

lemma *minf-less-0*:$-\infty < 0$
⟨*proof*⟩

lemma *ale-antisym*[*simp*]:$[\![(x::ant) \leq y;\ y \leq x\ ]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *x-gt-inf*[*simp*]:$\infty \leq x \Longrightarrow x = \infty$

⟨*proof*⟩

lemma *Zinf-pOp-closed*: $[\![x \in Z_\infty;\ y \in Z_\infty]\!] \Longrightarrow x + y \in Z_\infty$
⟨*proof*⟩

lemma *Zminf-pOp-closed*: $[\![x \in Z_{-\infty};\ y \in Z_{-\infty}]\!] \Longrightarrow x + y \in Z_{-\infty}$
⟨*proof*⟩

lemma *amult-distrib1*: $(ant\ z) \neq 0 \Longrightarrow$
$(a + b) * (ant\ z) = a * (ant\ z) + b * (ant\ z)$
⟨*proof*⟩

lemma *amult-0-r*: $(ant\ z) * 0 = 0$
⟨*proof*⟩

lemma *amult-0-l*: $0 * (ant\ z) = 0$
⟨*proof*⟩

definition
asprod :: $[int, ant] \Rightarrow ant$ (**infixl** $*_a$ *200*) **where**
$m *_a x ==$
if $x = \infty$ *then* (*if* $0 < m$ *then* ∞ *else* (*if* $m < 0$ *then* $-\infty$ *else*
if $m = 0$ *then* 0 *else undefined*))
else (*if* $x = -\infty$ *then*
(*if* $0 < m$ *then* $-\infty$ *else* (*if* $m < 0$ *then* ∞ *else*
if $m = 0$ *then* 0 *else undefined*))
else $(ant\ m) * x$)

lemma *asprod-pos-inf*[*simp*]: $0 < m \Longrightarrow m *_a \infty = \infty$
⟨*proof*⟩

lemma *asprod-neg-inf*[*simp*]: $m < 0 \Longrightarrow m *_a \infty = -\infty$
⟨*proof*⟩

lemma *asprod-pos-minf*[*simp*]: $0 < m \Longrightarrow m *_a (-\infty) = (-\infty)$
⟨*proof*⟩

lemma *asprod-neg-minf*[*simp*]: $m < 0 \Longrightarrow m *_a (-\infty) = \infty$
⟨*proof*⟩

lemma *asprod-mult*: $m *_a (ant\ n) = ant(m * n)$
⟨*proof*⟩

lemma *asprod-1*: $1 *_a x = x$
⟨*proof*⟩

lemma *agsprod-assoc-a*: $m *_a (n *_a (ant\ x)) = (m * n) *_a (ant\ x)$

⟨*proof*⟩

lemma *agsprod-assoc*:$[\![m \neq 0;\ n \neq 0]\!] \Longrightarrow m *_a (n *_a x) = (m * n) *_a x$
⟨*proof*⟩

lemma *asprod-distrib1*:$m \neq 0 \Longrightarrow m *_a (x + y) = (m *_a x) + (m *_a y)$
⟨*proof*⟩

lemma *asprod-0-x*[*simp*]:$0 *_a x = 0$
⟨*proof*⟩

lemma *asprod-n-0*:$n *_a 0 = 0$
⟨*proof*⟩

lemma *asprod-distrib2*:$[\![0 < i;\ 0 < j]\!] \Longrightarrow (i + j) *_a x = (i *_a x) + (j *_a x)$
⟨*proof*⟩

lemma *asprod-minus*:$x \neq -\infty \wedge x \neq \infty \Longrightarrow -z *_a x = z *_a (-x)$
⟨*proof*⟩

lemma *asprod-div-eq*:$[\![n \neq 0;\ n *_a x = n *_a y]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *asprod-0*:$[\![z \neq 0;\ z *_a x = 0\ ]\!] \Longrightarrow x = 0$
⟨*proof*⟩

lemma *asp-z-Z*:$z *_a ant\ x \in Z_\infty$
⟨*proof*⟩

lemma *tna-ant*: $tna\ (ant\ z) = z$
⟨*proof*⟩

lemma *ant-tna*:$x \neq \infty \wedge x \neq -\infty \Longrightarrow ant\ (tna\ x) = x$
⟨*proof*⟩

lemma *ant-sol*:$[\![a \in Z_\infty;\ b \in Z_\infty;\ c \in Z_\infty;\ b \neq \infty;\ a = b + c]\!] \Longrightarrow a - b = c$
⟨*proof*⟩

1.7.1 Ordering of integers and ordering nats

1.7.2 The ≤ Ordering

lemma *zneq-aneq*:$(n \neq m) = ((ant\ n) \neq (ant\ m))$
⟨*proof*⟩

lemma *ale*:$(n \leq m) = ((ant\ n) \leq (ant\ m))$
⟨*proof*⟩

lemma *aless*:$(n < m) = ((ant\ n) < (ant\ m))$
⟨*proof*⟩

lemma *ale-refl*: $w \leq (w::ant)$
⟨*proof*⟩

lemma *aeq-ale*:$(a::ant) = b \Longrightarrow a \leq b$
⟨*proof*⟩

lemma *ale-trans*: $[\![(i::ant) \leq j; j \leq k]\!] \Longrightarrow i \leq k$
⟨*proof*⟩

lemma *aless-le-not-le*: $((w::ant) < z) = (w \leq z \wedge \neg z \leq w)$
⟨*proof*⟩

instance *ant* :: *order*
⟨*proof*⟩

lemma *ale-linear*: $(z::ant) \leq w \vee w \leq z$
⟨*proof*⟩

instance *ant* :: *linorder*
⟨*proof*⟩

lemmas *aless-linear* = *less-linear* [**where** $'a = ant$]

lemma *ant-eq-0-conv* [*simp*]: $(ant\ n = 0) = (n = 0)$
⟨*proof*⟩

lemma *aless-zless*: $(ant\ m < ant\ n) = (m<n)$
⟨*proof*⟩

lemma *a0-less-int-conv* [*simp*]: $(0 < ant\ n) = (0 < n)$
⟨*proof*⟩

lemma *a0-less-1*: $0 < (1::ant)$
⟨*proof*⟩

lemma *a0-neq-1* [*simp*]: $0 \neq (1::ant)$
⟨*proof*⟩

lemma *ale-zle* [*simp*]: $((ant\ i) \leq (ant\ j)) = (i \leq j)$
⟨*proof*⟩

lemma *ant-1* [*simp*]: $ant\ 1 = 1$
⟨*proof*⟩

lemma *zpos-apos*:$(0 \leq n) = (0 \leq (ant\ n))$

⟨proof⟩

lemma *zposs-aposss*:$(0 < n) = (0 < (ant\ n))$
⟨proof⟩

lemma *an-nat-pos*[*simp*]:$0 \leq an\ n$
⟨proof⟩

lemma *amult-one-l*: $1 * (x::ant) = x$
⟨proof⟩

lemma *amult-one-r*:$(x::ant)* 1 = x$
⟨proof⟩

lemma *amult-eq-eq-r*:$[\![z \neq 0;\ \ a * ant\ z = b * ant\ z]\!] \Longrightarrow a = b$
⟨proof⟩

lemma *amult-eq-eq-l*:$[\![z \neq 0;\ \ (ant\ z) * a = (ant\ z) * b]\!] \Longrightarrow a = b$
⟨proof⟩

lemma *amult-pos*:$[\![0 < b;\ 0 \leq x]\!]\ \Longrightarrow x \leq (b *_a x)$
⟨proof⟩

lemma *asprod-amult*:$0 < z \Longrightarrow z *_a x = (ant\ z) * x$
⟨proof⟩

lemma *amult-pos1*:$[\![0 < b;\ 0 \leq x]\!]\ \Longrightarrow x \leq ((ant\ b) * x)$
⟨proof⟩

lemma *amult-pos-mono-l*:$0 < w \Longrightarrow (((ant\ w) * x) \leq ((ant\ w) * y)) = \ (x \leq y)$
⟨proof⟩

lemma *amult-pos-mono-r*:$0 < w \Longrightarrow ((x * (ant\ w)) \leq (y * (ant\ w))) = \ (x \leq y)$
⟨proof⟩

lemma *apos-neq-minf*:$0 \leq a \Longrightarrow a \neq -\infty$
⟨proof⟩

lemma *asprod-pos-mono*:$0 < w \Longrightarrow ((w *_a x) \leq (w *_a y)) = \ (x \leq y)$
⟨proof⟩

lemma *a-inv*:$(a::ant) + b = 0 \Longrightarrow a = -b$
⟨proof⟩

lemma *asprod-pos-pos*:$0 \leq x \Longrightarrow 0 \leq int\ n *_a x$
⟨proof⟩

lemma *asprod-1-x*[*simp*]:$1 *_a x = x$
⟨proof⟩

lemma *asprod-n-1*[*simp*]:$n *_a 1 = ant\ n$
⟨*proof*⟩

1.7.3 Aug ordering

lemma *aless-imp-le*: $x < (y::ant) \Longrightarrow x \le y$
⟨*proof*⟩

lemma *gt-a0-ge-1*:$(0::ant) < x \Longrightarrow 1 \le x$
⟨*proof*⟩

lemma *gt-a0-ge-aN*:$[\![0 < x;\ N \neq 0]\!] \Longrightarrow (ant\ (int\ N)) \le (int\ N) *_a x$
⟨*proof*⟩

lemma *aless-le-trans*:$[\![(x::ant) < y;\ y \le z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ale-less-trans*:$[\![(x::ant) \le y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *aless-trans*:$[\![(x::ant) < y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ale-neq-less*:$[\![\ (x::ant) \le y;\ x \neq y]\!] \Longrightarrow x < y$
⟨*proof*⟩

lemma *aneg-le*:$(\neg\ (x::ant) \le y) = (y < x)$
⟨*proof*⟩

lemma *aneg-less*:$(\neg\ x < (y::ant)) = (y \le x)$
⟨*proof*⟩

lemma *aadd-le-mono*:$x \le (y::ant) \Longrightarrow x + z \le y + z$
⟨*proof*⟩

lemma *aadd-less-mono-z*:$(x::ant) < y \Longrightarrow (x + (ant\ z)) < (y + (ant\ z))$
⟨*proof*⟩

lemma *aless-le-suc*[*simp*]:$(a::ant) < b \Longrightarrow a + 1 \le b$
⟨*proof*⟩

lemma *aposs-le-1*:$(0::ant) < x \Longrightarrow 1 \le x$
⟨*proof*⟩

lemma *pos-in-aug-inf*:$(0::ant) \le x \Longrightarrow x \in Z_\infty$
⟨*proof*⟩

lemma *aug-inf-noninf-is-z*:$[\![x \in Z_\infty;\ x \neq \infty]\!] \Longrightarrow \exists z.\ x = ant\ z$

⟨proof⟩

lemma *aadd-two-pos*:$[\![0 \leq (x::ant); 0 \leq y]\!] \Longrightarrow 0 \leq x + y$
⟨proof⟩

lemma *aadd-pos-poss*:$[\![(0::ant) \leq x; 0 < y]\!] \Longrightarrow 0 < (x + y)$
⟨proof⟩

lemma *aadd-poss-pos*:$[\![(0::ant) < x; 0 \leq y]\!] \Longrightarrow 0 < (x + y)$
⟨proof⟩

lemma *aadd-pos-le*:$0 \leq (a::ant) \Longrightarrow b \leq a + b$
⟨proof⟩

lemma *aadd-poss-less*:$[\![b \neq \infty; b \neq -\infty; 0 < a]\!] \Longrightarrow b < a + b$
⟨proof⟩

lemma *ale-neg*:$(0::ant) \leq x \Longrightarrow (- x) \leq 0$
⟨proof⟩

lemma *ale-diff-pos*:$(x::ant) \leq y \Longrightarrow 0 \leq (y - x)$
⟨proof⟩

lemma *aless-diff-poss*:$(x::ant) < y \Longrightarrow 0 < (y - x)$
⟨proof⟩

lemma *ale-minus*: $(x::ant) \leq y \Longrightarrow - y \leq - x$
⟨proof⟩

lemma *aless-minus*:$(x::ant) < y \Longrightarrow - y < - x$
⟨proof⟩

lemma *aadd-minus-le*:$(a::ant) \leq 0 \Longrightarrow a + b \leq b$
⟨proof⟩

lemma *aadd-minus-less*:$[\![b \neq -\infty \wedge b \neq \infty; (a::ant) < 0]\!] \Longrightarrow a + b < b$
⟨proof⟩

lemma *an-inj*:$an\ n = an\ m \Longrightarrow n = m$
⟨proof⟩

lemma *nat-eq-an-eq*:$n = m \Longrightarrow an\ n = an\ m$
⟨proof⟩

lemma *aneq-natneq*:$(an\ n \neq an\ m) = (n \neq m)$
⟨proof⟩

lemma *ale-natle*: $(an\ n \leq an\ m) = (n \leq m)$
⟨proof⟩

lemma *aless-natless*:$(an\ n < an\ m) = (n < m)$
⟨*proof*⟩

lemma *na-an*:$na\ (an\ n) = n$
⟨*proof*⟩

lemma *asprod-ge*:
$0 < b \Longrightarrow N \neq 0 \Longrightarrow an\ N \leq int\ N *_a b$
⟨*proof*⟩

lemma *an-npn*:$an\ (n + m) = an\ n + an\ m$
⟨*proof*⟩

lemma *an-ndn*:$n \leq m \Longrightarrow an\ (m - n) = an\ m - an\ n$
⟨*proof*⟩

1.8 Amin, amax

definition
amin :: [*ant*, *ant*] ⇒ *ant* **where**
$amin\ x\ y = (if\ (x \leq y)\ then\ x\ else\ y)$

definition
amax :: [*ant*, *ant*] ⇒ *ant* **where**
$amax\ x\ y = (if\ (x \leq y)\ then\ y\ else\ x)$

primrec *Amin* :: [*nat*, *nat* ⇒ *ant*] ⇒ *ant*
where
Amin-0 : $Amin\ 0\ f = (f\ 0)$
| *Amin-Suc* :$Amin\ (Suc\ n)\ f = amin\ (Amin\ n\ f)\ (f\ (Suc\ n))$

primrec *Amax* :: [*nat*, *nat* ⇒ *ant*] ⇒ *ant*
where
Amax-0 : $Amax\ 0\ f = f\ 0$
| *Amax-Suc* :$Amax\ (Suc\ n)\ f = amax\ (Amax\ n\ f)\ (f\ (Suc\ n))$

lemma *amin-ge*:$x \leq amin\ x\ y \vee y \leq amin\ x\ y$
⟨*proof*⟩

lemma *amin-le-l*:$amin\ x\ y \leq x$
⟨*proof*⟩

lemma *amin-le-r*:$amin\ x\ y \leq y$
⟨*proof*⟩

lemma *amax-le*:$amax\ x\ y \leq x \vee amax\ x\ y \leq y$
⟨*proof*⟩

lemma *amax-le-n*:$[\![x \le n; \; y \le n]\!] \Longrightarrow amax\ x\ y \le n$
⟨*proof*⟩

lemma *amax-ge-l*:$x \le amax\ x\ y$
⟨*proof*⟩

lemma *amax-ge-r*:$y \le amax\ x\ y$
⟨*proof*⟩

lemma *amin-mem-i*:$[\![x \in Z_\infty; \; y \in Z_\infty]\!] \Longrightarrow amin\ x\ y \in Z_\infty$
⟨*proof*⟩

lemma *amax-mem-m*:$[\![x \in Z_{-\infty}; \; y \in Z_{-\infty}]\!] \Longrightarrow amax\ x\ y \in Z_{-\infty}$
⟨*proof*⟩

lemma *amin-commute*:$amin\ x\ y = amin\ y\ x$
⟨*proof*⟩

lemma *amin-mult-pos*:$0 < z \Longrightarrow amin\ (z *_a x)\ (z *_a y) = z *_a amin\ x\ y$
⟨*proof*⟩

lemma *amin-amult-pos*:$0 < z \Longrightarrow$
$amin\ ((ant\ z) * x)\ ((ant\ z) * y) = (ant\ z) * amin\ x\ y$
⟨*proof*⟩

lemma *times-amin*:$[\![0 < a; \; amin\ (x * (ant\ a))\ (y * (ant\ a)) \le z * (ant\ a)]\!] \Longrightarrow$
$amin\ x\ y \le z$
⟨*proof*⟩

lemma *Amin-memTr*:$f \in \{i.\ i \le n\} \to Z_\infty \longrightarrow Amin\ n\ f \in Z_\infty$
⟨*proof*⟩

lemma *Amin-mem*:$f \in \{i.\ i \le n\} \to Z_\infty \Longrightarrow Amin\ n\ f \in Z_\infty$
⟨*proof*⟩

lemma *Amax-memTr*:$f \in \{i.\ i \le n\} \to Z_{-\infty} \longrightarrow Amax\ n\ f \in Z_{-\infty}$
⟨*proof*⟩

lemma *Amax-mem*:$f \in \{i.\ i \le n\} \to Z_{-\infty} \Longrightarrow Amax\ n\ f \in Z_{-\infty}$
⟨*proof*⟩

lemma *Amin-mem-mem*:$\forall j \le n.\ f\ j \in Z_\infty \Longrightarrow Amin\ n\ f \in Z_\infty$
⟨*proof*⟩

lemma *Amax-mem-mem*:$\forall j \le n.\ f\ j \in Z_{-\infty} \Longrightarrow Amax\ n\ f \in Z_{-\infty}$
⟨*proof*⟩

lemma *Amin-leTr*:$f \in \{i.\ i \le n\} \to Z_\infty \longrightarrow (\forall j \in \{i.\ i \le n\}.\ Amin\ n\ f \le (f\ j))$
⟨*proof*⟩

lemma *Amin-le*:$[\![f \in \{j.\ j \leq n\} \rightarrow\ Z_\infty;\ j \in \{k.\ k \leq n\}]\!] \Longrightarrow Amin\ n\ f \leq (f\ j)$
⟨*proof*⟩

lemma *Amax-geTr*:$f \in \{j.\ j \leq n\} \rightarrow Z_{-\infty} \longrightarrow (\forall j{\in}\{j.\ j \leq n\}.\ (f\ j) \leq Amax\ n\ f)$
⟨*proof*⟩

lemma *Amax-ge*:$[\![f \in \{j.\ j \leq n\} \rightarrow Z_{-\infty};\ j \in \{j.\ j \leq n\}]\!] \Longrightarrow$
$(f\ j) \leq (Amax\ n\ f)$
⟨*proof*⟩

lemma *Amin-mem-le*:$[\![\forall j \leq n.\ (f\ j) \in\ Z_\infty;\ j \in \{j.\ j \leq n\}]\!] \Longrightarrow$
$(Amin\ n\ f) \leq (f\ j)$
⟨*proof*⟩

lemma *Amax-mem-le*:$[\![\forall j \leq n.\ (f\ j) \in\ Z_{-\infty};\ j \in \{j.\ j \leq n\}]\!] \Longrightarrow$
$(f\ j) \leq (Amax\ n\ f)$
⟨*proof*⟩

lemma *amin-ge1*:$[\![(z{::}ant) \leq x;\ z \leq y\ ]\!] \Longrightarrow z \leq amin\ x\ y$
⟨*proof*⟩

lemma *amin-gt*:$[\![(z{::}ant) < x;\ z < y]\!] \Longrightarrow z < amin\ x\ y$
⟨*proof*⟩

lemma *Amin-ge1Tr*:$(\forall j{\leq}(Suc\ n).\ (f\ j) \in Z_\infty \wedge z \leq (f\ j)) \longrightarrow$
$z \leq (Amin\ (Suc\ n)\ f)$
⟨*proof*⟩

lemma *Amin-ge1*:$[\![\ \forall j \leq (Suc\ n).\ f\ j \in Z_\infty;\ \forall j \leq (Suc\ n).\ z \leq (f\ j)]\!] \Longrightarrow$
$z \leq (Amin\ (Suc\ n)\ f)$
⟨*proof*⟩

lemma *amin-trans1*:$[\![x \in Z_\infty;\ y \in Z_\infty;\ z \in Z_\infty;\ z \leq x\ ]\!] \Longrightarrow$
$amin\ z\ y \leq amin\ x\ y$
⟨*proof*⟩

lemma *inf-in-aug-inf*:$\infty\ \in Z_\infty$
⟨*proof*⟩

1.8.1 Maximum element of a set of ants

primrec *aasc-seq* :: [*ant set*, *ant*, *nat*] ⇒ *ant*
where
aasc-seq-0 : $aasc\text{-}seq\ A\ a\ 0 = a$
| *aasc-seq-Suc* : $aasc\text{-}seq\ A\ a\ (Suc\ n) =$
$(SOME\ b.\ ((b \in A) \wedge (aasc\text{-}seq\ A\ a\ n) < b))$

lemma *aasc-seq-mem*:$[\![a \in A; \neg (\exists m.\ m\in A \wedge (\forall x\in A.\ x \le m))]\!] \Longrightarrow$
$(aasc\text{-}seq\ A\ a\ n) \in A$
⟨*proof*⟩

lemma *aasc-seqn*:$[\![a \in A; \neg (\exists m.\ m\in A \wedge (\forall x\in A.\ x \le m))]\!] \Longrightarrow$
$(aasc\text{-}seq\ A\ a\ n) < (aasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨*proof*⟩

lemma *aasc-seqn1*:$[\![a \in A; \neg (\exists m.\ m\in A \wedge (\forall x\in A.\ x \le m))]\!] \Longrightarrow$
$(aasc\text{-}seq\ A\ a\ n) + 1 \le (aasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨*proof*⟩

lemma *aubs-ex-n-maxTr*:$[\![a \in A; \neg (\exists m.\ m\in A \wedge (\forall x\in A.\ x \le m))]\!] \Longrightarrow$
$(a + an\ n) \le (aasc\text{-}seq\ A\ a\ n)$
⟨*proof*⟩

lemma *aubs-ex-AMax*:$[\![A \subseteq UBset\ (ant\ z); A \neq \{\}]\!] \Longrightarrow \exists!m.\ m\in A \wedge (\forall x\in A.\ x$
$\le m)$
⟨*proof*⟩

definition
$AMax :: ant\ set \Rightarrow ant$ **where**
$AMax\ A = (THE\ m.\ m \in A \wedge (\forall x\in A.\ x \le m))$

definition
$AMin::ant\ set \Rightarrow ant$ **where**
$AMin\ A = (THE\ m.\ m \in A \wedge (\forall x\in A.\ m \le x))$

definition
$rev\text{-}o :: ant \Rightarrow ant$ **where**
$rev\text{-}o\ x = -\ x$

lemma *AMax*:$[\![A \subseteq UBset\ (ant\ z); A \neq \{\}]\!] \Longrightarrow$
$(AMax\ A) \in A \wedge (\forall x\in A.\ x \le (AMax\ A))$
⟨*proof*⟩

lemma *AMax-mem*:$[\![A \subseteq UBset\ (ant\ z); A \neq \{\}]\!] \Longrightarrow (AMax\ A) \in A$
⟨*proof*⟩

lemma *rev-map-nonempty*:$A \neq \{\} \Longrightarrow rev\text{-}o\ `\ A \neq \{\}$
⟨*proof*⟩

lemma *rev-map*:$rev\text{-}o \in LBset\ (ant\ (-z)) \rightarrow UBset\ (ant\ z)$
⟨*proof*⟩

lemma *albs-ex-AMin*:$[\![A \subseteq LBset\ (ant\ z); A \neq \{\}]\!] \Longrightarrow \exists!m.\ m\in A \wedge (\forall x\in A.\ m$
$\le x)$
⟨*proof*⟩

lemma *AMin*:$[\![A \subseteq LBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow$
$(AMin\ A) \in A \wedge (\forall x{\in}A.\ (AMin\ A) \leq x)$
⟨*proof*⟩

lemma *AMin-mem*:$[\![A \subseteq LBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow (AMin\ A) \in A$
⟨*proof*⟩

primrec *ASum* :: $(nat \Rightarrow ant) \Rightarrow nat \Rightarrow ant$
where
ASum-0: $ASum\ f\ 0 = f\ 0$
| *ASum-Suc*: $ASum\ f\ (Suc\ n) = (ASum\ f\ n) + (f\ (Suc\ n))$

lemma *age-plus*:$[\![0 \leq (a{::}ant);\ 0 \leq b;\ a + b \leq c]\!] \Longrightarrow a \leq c$
⟨*proof*⟩

lemma *age-diff-le*:$[\![(a{::}ant) \leq c;\ 0 \leq b]\!] \Longrightarrow a - b \leq c$
⟨*proof*⟩

lemma *adiff-le-adiff*:$a \leq (a'{::}ant) \Longrightarrow a - b \leq a' - b$
⟨*proof*⟩

lemma *aplus-le-aminus*:$[\![\ a \in\ Z_{-\infty};\ b \in\ Z_{-\infty};\ c \in\ Z_{-\infty};\ -b \in\ Z_{-\infty}]\!] \Longrightarrow$
$((a + b) \leq (c{::}ant)) = (a \leq c - b)$
⟨*proof*⟩

1.9 Cardinality of sets

cardinality is defined for the finite sets only

lemma *card-eq*:$A = B \Longrightarrow card\ A = card\ B$
⟨*proof*⟩

lemma *card0*:$card\ \{\} = 0$
⟨*proof*⟩

lemma *card-nonzero*:$[\![finite\ A;\ card\ A \neq 0]\!] \Longrightarrow A \neq \{\}$
⟨*proof*⟩

lemma *finite1*:$finite\ \{a\}$
⟨*proof*⟩

lemma *card1*:$card\ \{a\} = 1$
⟨*proof*⟩

lemma *nonempty-card-pos*:$[\![finite\ A;\ A \neq \{\}]\!] \Longrightarrow 0 < card\ A$
⟨*proof*⟩

lemma *nonempty-card-pos1*:$[\![finite\ A;\ A \neq \{\}]\!] \Longrightarrow Suc\ 0 \leq card\ A$
⟨*proof*⟩

lemma *card1-tr0*:$[\![$ *finite A*; *card A* = *Suc 0*; $a \in A$ $]\!] \Longrightarrow \{a\} = A$
⟨*proof*⟩

lemma *card1-tr1*:(*constmap* {*0*::*nat*} {*x*}) ∈ {*0*} → {*x*} ∧
surj-to (*constmap* {*0*::*nat*} {*x*}) {*0*} {*x*}
⟨*proof*⟩

lemma *card1-Tr2*:$[\![$*finite A*; *card A* = *Suc 0*$]\!] \Longrightarrow$
∃ *f*. *f* ∈ {*0*::*nat*} → *A* ∧ *surj-to f* {*0*} *A*
⟨*proof*⟩

lemma *card2*:$[\![$ *finite A*; $a \in A$; $b \in A$; $a \neq b$ $]\!] \Longrightarrow$ *Suc* (*Suc 0*) ≤ *card A*
⟨*proof*⟩

lemma *card2-inc-two*:$[\![$*0* < (*n*::*nat*); $x \in \{j.\ j \leq n\}$$]\!] \Longrightarrow$
$\exists\, y \in \{j.\ j \leq n\}.\ x \neq y$
⟨*proof*⟩

lemma *Nset2-prep1*:$[\![$*finite A*; *card A* = *Suc* (*Suc n*) $]\!] \Longrightarrow \exists\, x.\ x{\in}A$
⟨*proof*⟩

lemma *ex-least-set*:$[\![$*A* = {*H*. *finite H* ∧ *P H*}; *H* ∈ *A*$]\!] \Longrightarrow$
∃ *K* ∈ *A*. (*LEAST j*. *j* ∈ (*card* ‘ *A*)) = *card K*

⟨*proof*⟩

lemma *Nset2-prep2*:$x \in A \Longrightarrow A - \{x\} \cup \{x\} = A$
⟨*proof*⟩

lemma *Nset2-finiteTr*:∀ *A*. (*finite A* ∧(*card A* = *Suc n*) ⟶
(∃ *f*. *f* ∈ {*i*. *i* ≤ *n*} → *A* ∧ *surj-to f* {*i*. *i* ≤ *n*} *A*))
⟨*proof*⟩

lemma *Nset2-finite*:$[\![$ *finite A*; *card A* = *Suc n*$]\!] \Longrightarrow$
∃ *f*. *f* ∈ {*i*. *i* ≤ *n*} → *A* ∧ *surj-to f* {*i*. *i* ≤ *n*} *A*
⟨*proof*⟩

lemma *Nset2finite-inj-tr0*:*j* ∈ {*i*. *i* ≤ (*n*::*nat*)} ⟹
card ({*i*. *i* ≤ *n*} − {*j*}) = *n*
⟨*proof*⟩

lemma *Nset2finite-inj-tr1*:$[\![$ *i* ≤ (*n*::*nat*); *j* ≤ *n*; *f i* = *f j*; *i* ≠ *j* $]\!] \Longrightarrow$
f ‘ ({*i*. *i* ≤ *n*} − {*j*}) = *f* ‘ {*i*. *i* ≤ *n*}
⟨*proof*⟩

lemma *Nset2finite-inj*:$[\![$*finite A*; *card A* = *Suc n*; *surj-to f* {*i*. *i* ≤ *n*} *A* $]\!] \Longrightarrow$

inj-on f $\{i.\ i \le n\}$
⟨*proof*⟩

definition
zmax :: [*int*, *int*] ⇒ *int* **where**
zmax x y = (*if* $(x \le y)$ *then y else x*)

primrec *Zmax* :: [*nat*, *nat* ⇒ *int*] ⇒ *int*
where
Zmax-0 : *Zmax 0 f* = *f 0*
| *Zmax-Suc* :*Zmax* (*Suc n*) *f* = *zmax* (*Zmax n f*) (*f* (*Suc n*))

lemma *Zmax-memTr*:$f \in \{i.\ i \le (n::nat)\} \to (UNIV::int\ set) \longrightarrow$
$Zmax\ n\ f \in f\ `\ \{i.\ i \le n\}$
⟨*proof*⟩

lemma *zmax-ge-r*:$y \le zmax\ x\ y$
⟨*proof*⟩

lemma *zmax-ge-l*:$x \le zmax\ x\ y$
⟨*proof*⟩

lemma *Zmax-geTr*:$f \in \{j.\ j \le (n::nat)\} \to (UNIV::int\ set) \longrightarrow$
$(\forall j \in \{j.\ j \le n\}.\ (f\ j) \le Zmax\ n\ f)$
⟨*proof*⟩

lemma *Zmax-plus1*:$f \in \{j.\ j \le (n::nat)\} \to (UNIV::int\ set) \Longrightarrow$
$((Zmax\ n\ f) + 1) \notin f\ `\ \{j.\ j \le n\}$
⟨*proof*⟩

lemma *image-Nsetn-card-pos*: $0 < card\ (f\ `\ \{i.\ i \le (n::nat)\})$
⟨*proof*⟩

lemma *card-image-Nsetn-Suc*
:$[\![f \in \{j.\ j \le Suc\ n\} \to B;$
$f\ (Suc\ n) \notin f\ `\ \{j.\ j \le n\}]\!] \Longrightarrow$
$card\ (f\ `\ \{j.\ j \le Suc\ n\}) - Suc\ 0 =$
$Suc\ (card\ (f\ `\ \{j.\ j \le n\}) - Suc\ 0)$
⟨*proof*⟩

lemma *slide-surj*:$i < (j::nat) \Longrightarrow$
$surj\text{-}to\ (slide\ i)\ \{l.\ l \le (j - i)\}\ (nset\ i\ j)$
⟨*proof*⟩

lemma *slide-inj*:$i < j \Longrightarrow inj\text{-}on\ (slide\ i)\ \{k.\ k \le (j - i)\}$
⟨*proof*⟩

lemma *card-nset*:$i < (j :: nat) \Longrightarrow card\ (nset\ i\ j) = Suc\ (j - i)$
⟨*proof*⟩

lemma *sliden-hom*:$i < j \Longrightarrow (sliden\ i) \in nset\ i\ j \rightarrow \{k.\ k \leq (j - i)\}$
⟨*proof*⟩

lemma *slide-sliden*:$(sliden\ i)\ (slide\ i\ k) = k$
⟨*proof*⟩

lemma *sliden-surj*:$i < j \Longrightarrow surj\text{-}to\ (sliden\ i)\ (nset\ i\ j)\ \{k.\ k \leq (j - i)\}$
⟨*proof*⟩

lemma *sliden-inj*: $i < j \Longrightarrow inj\text{-}on\ (sliden\ i)\ (nset\ i\ j)$
⟨*proof*⟩

definition
transpos :: $[nat, nat] \Rightarrow (nat \Rightarrow nat)$ **where**
$transpos\ i\ j = (\lambda k.\ if\ k = i\ then\ j\ else\ if\ k = j\ then\ i\ else\ k)$

lemma *transpos-id*:$\llbracket\ i \leq n;\ j \leq n;\ i \neq j\ ;\ x \in \{k.\ k \leq n\} - \{i, j\}\ \rrbracket$
$\Longrightarrow transpos\ i\ j\ x = x$
⟨*proof*⟩

lemma *transpos-id-1*:$\llbracket i \leq n;\ j \leq n;\ i \neq j;\ x \leq n;\ x \neq i;\ x \neq j\rrbracket \Longrightarrow$
$transpos\ i\ j\ x = x$
⟨*proof*⟩

lemma *transpos-id-2*:$i \leq n \Longrightarrow transpos\ i\ n\ (Suc\ n) = Suc\ n$
⟨*proof*⟩

lemma *transpos-ij-1*:$\llbracket i \leq n;\ j \leq n;\ i \neq j\ \rrbracket \Longrightarrow$
$transpos\ i\ j\ i = j$
⟨*proof*⟩

lemma *transpos-ij-2*:$\llbracket i \leq n;\ j \leq n;\ i \neq j\rrbracket \Longrightarrow transpos\ i\ j\ j = i$
⟨*proof*⟩

lemma *transpos-hom*:$\llbracket i \leq n;\ j \leq n;\ i \neq j\rrbracket \Longrightarrow$
$(transpos\ i\ j) \in \{i.\ i \leq n\} \rightarrow \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *transpos-mem*:$\llbracket i \leq n;\ j \leq n;\ i \neq j;\ l \leq n\rrbracket \Longrightarrow$
$(transpos\ i\ j\ l) \leq n$
⟨*proof*⟩

lemma *transpos-inj*:$\llbracket i \leq n;\ j \leq n;\ i \neq j\rrbracket$
$\Longrightarrow inj\text{-}on\ (transpos\ i\ j)\ \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *transpos-surjec*:$\llbracket i \leq n;\ j \leq n;\ i \neq j\rrbracket$

$\Longrightarrow$ *surj-to* (*transpos i j*) $\{i.\ i \le n\}$ $\{i.\ i \le n\}$
⟨*proof*⟩

lemma *comp-transpos*:$[\![i \le n;\ j \le n;\ i \ne j]\!] \Longrightarrow$
$\forall k \le n.$ (*compose* $\{i.\ i \le n\}$ (*transpos i j*) (*transpos i j*)) $k = k$
⟨*proof*⟩

lemma *comp-transpos-1*:$[\![i \le n;\ j \le n;\ i \ne j;\ k \le n]\!] \Longrightarrow$
(*transpos i j*) ((*transpos i j*) k) $= k$
⟨*proof*⟩

lemma *cmp-transpos1*:$[\![i \le n;\ j \le n;\ i \ne j;\ k \le n]\!] \Longrightarrow$
(*cmp* (*transpos i j*) (*transpos i j*)) $k = k$
⟨*proof*⟩

lemma *cmp-transpos*:$[\![i \le n;\ i \ne n;\ a \le (Suc\ n)]\!] \Longrightarrow$
(*cmp* (*transpos i n*) (*cmp* (*transpos n* (*Suc n*)) (*transpos i n*))) a =
transpos i (*Suc n*) a
⟨*proof*⟩

lemma *im-Nset-Suc*:*insert* (f (*Suc n*)) (f ‘ $\{i.\ i \le n\}$) = f ‘ $\{i.\ i \le (Suc\ n)\}$
⟨*proof*⟩

lemma *Nset-injTr0*:$[\![f \in \{i.\ i \le (Suc\ n)\} \to \{i.\ i \le (Suc\ n)\};$
inj-on f $\{i.\ i \le (Suc\ n)\}$; f (*Suc n*) = *Suc n*$]\!] \Longrightarrow$
$f \in \{i.\ i \le n\} \to \{i.\ i \le n\} \wedge$ *inj-on* f $\{i.\ i \le n\}$
⟨*proof*⟩

lemma *inj-surj*:$[\![f \in \{i.\ i \le (n::nat)\} \to \{i.\ i \le n\};$
inj-on f $\{i.\ i \le (n::nat)\}]\!] \Longrightarrow f$ ‘ $\{i.\ i \le n\} = \{i.\ i \le n\}$
⟨*proof*⟩

lemma *Nset-pre-mem*:$[\![f{:}\{i.\ i{\le}(Suc\ n)\} \to \{i.\ i{\le}(Suc\ n)\};$
inj-on f $\{i.\ i{\le}(Suc\ n)\}$; f (*Suc n*) = *Suc n*; $k \le n]\!] \Longrightarrow f\ k \in \{i.\ i{\le}n\}$
⟨*proof*⟩

lemma *Nset-injTr1*:$[\![\ \forall l \le (Suc\ n).\ f\ l \le (Suc\ n);$ *inj-on* f $\{i.\ i \le (Suc\ n)\};$
f (*Suc n*) = *Suc n* $]\!] \Longrightarrow$ *inj-on* f $\{i.\ i \le n\}$
⟨*proof*⟩

lemma *Nset-injTr2*:$[\![\forall l{\le}\ (Suc\ n).\ f\ l \le (Suc\ n);$ *inj-on* f $\{i.\ i \le (Suc\ n)\};$
f (*Suc n*) = *Suc n*$]\!] \Longrightarrow \forall l \le n.\ f\ l \le n$
⟨*proof*⟩

lemma *TR-inj-inj*:$[\![\forall l{\le}\ (Suc\ n).\ f\ l \le (Suc\ n);$ *inj-on* f $\{i.\ i \le (Suc\ n)\};$
$i \le (Suc\ n);\ j \le (Suc\ n);\ i < j\ ]\!] \Longrightarrow$
inj-on (*compose* $\{i.\ i \le (Suc\ n)\}$ (*transpos i j*) f) $\{i.\ i \le (Suc\ n)\}$
⟨*proof*⟩

lemma *enumeration*:$[\![f \in \{i.\ i \le (n::nat)\} \to \{i.\ i \le m\};\ inj\text{-}on\ f\ \{i.\ i \le n\}]\!] \Longrightarrow n \le m$
⟨*proof*⟩

lemma *enumerate-1*:$[\![\forall j \le (n::nat).\ f\ j \in A;\ \forall j \le (m::nat).\ g\ j \in A;\ inj\text{-}on\ f\ \{i.\ i \le n\};\ inj\text{-}on\ g\ \{j.\ j \le m\};\ f\ `\{j.\ j \le n\} = A;\ g\ `\ \{j.\ j \le m\} = A\ ]\!] \Longrightarrow n = m$
⟨*proof*⟩

definition
$ninv :: [nat, (nat \Rightarrow nat)] \Rightarrow (nat \Rightarrow nat)$ **where**
$ninv\ n\ f = (\lambda y{\in}\{i.\ i \le n\}.\ (SOME\ x.\ (x \le n \wedge y = f\ x)))$

lemma *ninv-hom*:$[\![f \in \{i.\ i \le n\} \to \{i.\ i \le n\};\ inj\text{-}on\ f\ \{i.\ i \le n\}]\!] \Longrightarrow ninv\ n\ f \in \{i.\ i \le n\} \to \{i.\ i \le n\}$
⟨*proof*⟩

lemma *ninv-r-inv*:$[\![f \in \{i.\ i \le (n::nat)\} \to \{i.\ i \le n\};\ inj\text{-}on\ f\ \{i.\ i \le n\};\ b \le n]\!] \Longrightarrow f\ (ninv\ n\ f\ b) = b$
⟨*proof*⟩

lemma *ninv-inj*:$[\![f \in \{i.\ i \le n\} \to \{i.\ i \le n\};\ inj\text{-}on\ f\ \{i.\ i \le n\}]\!] \Longrightarrow inj\text{-}on\ (ninv\ n\ f)\ \{i.\ i \le n\}$
⟨*proof*⟩

1.9.1 Lemmas required in Algebra6.thy

lemma *ge2-zmult-pos*:
$2 \le m \Longrightarrow 0 < z \Longrightarrow 1 < int\ m * z$
⟨*proof*⟩

lemma *zmult-pos-mono*:$[\![\ (0::int) < w;\ w * z \le w * z']\!] \Longrightarrow z \le z'$
⟨*proof*⟩

lemma *zmult-pos-mono-r*:
$[\![(0::int) < w;\ z * w \le z' * w]\!] \Longrightarrow z \le z'$
⟨*proof*⟩

lemma *an-neq-inf*:$an\ n \ne \infty$
⟨*proof*⟩

lemma *an-neq-minf*:$an\ n \ne -\infty$
⟨*proof*⟩

lemma *aeq-mult*:$[\![z \ne 0;\ a = b]\!] \Longrightarrow a * ant\ z = b * ant\ z$
⟨*proof*⟩

lemma *tna-0*[*simp*]:$tna\ 0 = 0$
⟨*proof*⟩

lemma *ale-nat-le*:$(an\ n \leq an\ m) = (n \leq m)$
⟨*proof*⟩

lemma *aless-nat-less*:$(an\ n < an\ m) = (n < m)$
⟨*proof*⟩

lemma *apos-natpos*:$[\![a \neq \infty;\ 0 \leq a]\!] \Longrightarrow 0 \leq na\ a$
⟨*proof*⟩

lemma *apos-tna-pos*:$[\![n \neq \infty;\ 0 \leq n]\!] \Longrightarrow 0 \leq tna\ n$
⟨*proof*⟩

lemma *apos-na-pos*:$[\![n \neq \infty;\ 0 \leq n]\!] \Longrightarrow 0 \leq na\ n$
⟨*proof*⟩

lemma *aposs-tna-poss*:$[\![n \neq \infty;\ 0 < n]\!] \Longrightarrow 0 < tna\ n$
⟨*proof*⟩

lemma *aposs-na-poss*:$[\![n \neq \infty;\ 0 < n]\!] \Longrightarrow 0 < na\ n$
⟨*proof*⟩

lemma *nat-0-le*: $0 \leq z ==> int\ (nat\ z) = z$
⟨*proof*⟩

lemma *int-eq*:$m = n \Longrightarrow int\ m = int\ n$
⟨*proof*⟩

lemma *box-equation*:$[\![a = b;\ a = c]\!] \Longrightarrow b = c$
⟨*proof*⟩

lemma *aeq-nat-eq*:$[\![n \neq \infty;\ 0 \leq n;\ m \neq \infty;\ 0 \leq m]\!] \Longrightarrow$
$(n = m) = (na\ n = na\ m)$
⟨*proof*⟩

lemma *na-minf*:$na\ (-\infty) = 0$
⟨*proof*⟩

lemma *an-na*:$[\![a \neq \infty;\ 0 \leq a]\!] \Longrightarrow an\ (na\ a) = a$
⟨*proof*⟩

lemma *not-na-le-minf*:$\neg\ (an\ n \leq -\infty\ )$
⟨*proof*⟩

lemma *not-na-less-minf*:$\neg\ (an\ n < -\infty)$
⟨*proof*⟩

lemma *not-na-ge-inf*:$\neg\ \infty \leq (an\ n)$

⟨*proof*⟩

lemma *an-na-le*:$j \leq an\ n \Longrightarrow na\ j \leq n$
⟨*proof*⟩

lemma *aless-neq* :$(x::ant) < y \Longrightarrow x \neq y$
⟨*proof*⟩

Chapter 2

Ordered Set

2.1 Basic Concepts of Ordered Sets

record *'a carrier* =
  carrier :: *'a set*

record *'a Order* = *'a carrier* +
  rel :: (*'a* × *'a*) *set*

locale *Order* =
  fixes *D* (**structure**)
  assumes *closed*: $rel\ D \subseteq carrier\ D \times carrier\ D$
    and *refl*: $a \in carrier\ D \Longrightarrow (a,\ a) \in rel\ D$
    and *antisym*: $[\![a \in carrier\ D;\ b \in carrier\ D;\ (a,\ b) \in rel\ D;$
      $(b,\ a) \in rel\ D]\!] \Longrightarrow a = b$
    and *trans*: $[\![a \in carrier\ D;\ b \in carrier\ D;\ c \in carrier\ D;$
      $(a,\ b) \in rel\ D;\ (b,\ c) \in rel\ D]\!] \Longrightarrow (a,\ c) \in rel\ D$

definition
  ole :: - ⇒ *'a* ⇒ *'a* ⇒ *bool* (**infix** $\preceq\imath$ *60*) **where**
  $a \preceq_D b \longleftrightarrow (a,\ b) \in rel\ D$

definition
  oless :: - ⇒ *'a* ⇒ *'a* ⇒ *bool* (**infix** $\prec\imath$ *60*) **where**
  $a \prec_D b \equiv a \preceq_D b \wedge a \neq b$

lemma *Order-component*:(*E*::*'a Order*) = (| *carrier* = *carrier E*, *rel* = *rel E* |)
⟨*proof*⟩

lemma *Order-comp-eq*:$[\![carrier\ (E::'a\ Order) = carrier\ (F::'a\ Order);$
  $rel\ E = rel\ F]\!] \Longrightarrow E = F$
⟨*proof*⟩

lemma (**in** *Order*) *le-rel*: $[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \preceq b) = ((a,\ b) \in rel\ D)$
⟨*proof*⟩

lemma (**in** *Order*) *less-imp-le*:
$[\![a \in carrier\ D;\ b \in carrier\ D;\ a \prec b\ ]\!] \Longrightarrow a \preceq b$
⟨*proof*⟩

lemma (**in** *Order*) *le-refl*: $a \in carrier\ D \Longrightarrow a \preceq a$
⟨*proof*⟩

lemma (**in** *Order*) *le-antisym*: $[\![a \in carrier\ D;\ b \in carrier\ D;$
$a \preceq b;\ b \preceq a\ ]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *Order*) *le-trans*: $[\![a \in carrier\ D;\ b \in carrier\ D;\ c \in carrier\ D;$
$a \preceq b;\ b \preceq c\ ]\!] \Longrightarrow a \preceq c$
⟨*proof*⟩

lemma (**in** *Order*) *less-trans*: $[\![a \in carrier\ D;\ b \in carrier\ D;\ c \in carrier\ D;$
$a \prec b;\ b \prec c\ ]\!] \Longrightarrow a \prec c$
⟨*proof*⟩

lemma (**in** *Order*) *le-less-trans*: $[\![a \in carrier\ D;\ b \in carrier\ D;\ c \in carrier\ D;$
$a \preceq b;\ b \prec c\ ]\!] \Longrightarrow a \prec c$
⟨*proof*⟩

lemma (**in** *Order*) *less-le-trans*: $[\![a \in carrier\ D;\ b \in carrier\ D;\ c \in carrier\ D;$
$a \prec b;\ b \preceq c\ ]\!] \Longrightarrow a \prec c$
⟨*proof*⟩

lemma (**in** *Order*) *le-imp-less-or-eq*:
$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow (a \preceq b) = (a \prec b \vee a = b)$
⟨*proof*⟩

lemma (**in** *Order*) *less-neq*: $a \prec b \Longrightarrow a \neq b$
⟨*proof*⟩

lemma (**in** *Order*) *le-neq-less*: $[\![a \preceq b;\ a \neq b]\!] \Longrightarrow a \prec b$
⟨*proof*⟩

lemma (**in** *Order*) *less-irrefl*: $[\![a \in carrier\ D;\ a \prec a]\!] \Longrightarrow C$
⟨*proof*⟩

lemma (**in** *Order*) *less-irrefl'*: $a \in carrier\ D \Longrightarrow \neg\ a \prec a$
⟨*proof*⟩

lemma (**in** *Order*) *less-asym*:

$a \in carrier\ D \Longrightarrow b \in carrier\ D \Longrightarrow a \prec b \Longrightarrow b \prec a \Longrightarrow C$
⟨proof⟩

lemma (**in** *Order*) *less-asym'*:
$a \in carrier\ D \Longrightarrow b \in carrier\ D \Longrightarrow a \prec b \Longrightarrow \neg\ b \prec a$
⟨proof⟩

lemma (**in** *Order*) *gt-than-any-outside*:$\llbracket A \subseteq carrier\ D;\ b \in carrier\ D;$
$\forall x{\in}A.\ x \prec b\rrbracket \Longrightarrow b \notin A$
⟨proof⟩

definition
Iod :: - ⇒ *'a set* ⇒ - **where**
$Iod\ D\ T =$
$D\ (\!|carrier := T,\ rel := \{(a,\ b).\ (a,\ b) \in rel\ D \wedge a \in T \wedge b \in T\}|\!)$

definition
SIod :: *'a Order* ⇒ *'a set* ⇒ *'a Order* **where**
$SIod\ D\ T = (\!|carrier = T,\ rel = \{(a,\ b).\ (a,\ b) \in rel\ D \wedge a \in T \wedge b \in T\}|\!)$

lemma (**in** *Order*) *Iod-self*: $D = Iod\ D\ (carrier\ D)$
⟨proof⟩

lemma *SIod-self*:$Order\ D \Longrightarrow D = SIod\ D\ (carrier\ D)$
⟨proof⟩

lemma (**in** *Order*) *Od-carrier*:$carrier\ (D(\!|carrier := S,\ rel := R|\!)) = S$
⟨proof⟩

lemma (**in** *Order*) *Od-rel*:$rel\ (D(\!|carrier := S,\ rel := R|\!)) = R$
⟨proof⟩

lemma (**in** *Order*) *Iod-carrier*:
$T \subseteq carrier\ D \Longrightarrow carrier\ (Iod\ D\ T) = T$
⟨proof⟩

lemma *SIod-carrier*:$\llbracket Order\ D;\ T \subseteq carrier\ D\rrbracket \Longrightarrow carrier\ (SIod\ D\ T) = T$
⟨proof⟩

lemma (**in** *Order*) *Od-compare*:$(S = S' \wedge R = R') = (D(\!|carrier := S,\ rel := R|\!)$
$= D(\!|carrier := S',\ rel := R'|\!))$
⟨proof⟩

lemma (**in** *Order*) *Iod-le*:
$\llbracket T \subseteq carrier\ D;\ a \in T;\ b \in T\rrbracket \Longrightarrow (a \preceq_{Iod\ D\ T} b) = (a \preceq b)$
⟨proof⟩

lemma *SIod-le*:$\llbracket T \subseteq carrier\ D;\ a \in T;\ b \in T\rrbracket \Longrightarrow$
$(a \preceq_{SIod\ D\ T} b) = (a \preceq_D b)$

$\langle proof \rangle$

lemma (**in** *Order*) *Iod-less*:
$[\![T \subseteq carrier\ D;\ a \in T;\ b \in T]\!] \Longrightarrow (a \prec_{Iod\ D\ T}\ b) = (a \prec b)$
$\langle proof \rangle$

lemma *SIod-less*:$[\![T \subseteq carrier\ D;\ a \in T;\ b \in T]\!] \Longrightarrow$
$(a \prec_{SIod\ D\ T}\ b) = (a \prec_D\ b)$
$\langle proof \rangle$

lemma (**in** *Order*) *Iod-Order*:
$T \subseteq carrier\ D \Longrightarrow Order\ (Iod\ D\ T)$
$\langle proof \rangle$

lemma *SIod-Order*:$[\![\ Order\ D;\ T \subseteq carrier\ D]\!] \Longrightarrow Order\ (SIod\ D\ T)$
$\langle proof \rangle$

lemma (**in** *Order*) *emptyset-Iod*:$Order\ (Iod\ D\ \{\})$
$\langle proof \rangle$

lemma (**in** *Order*) *Iod-sub-sub*:
$[\![S \subseteq T;\ T \subseteq carrier\ D]\!] \Longrightarrow Iod\ (Iod\ D\ T)\ S = Iod\ D\ S$
$\langle proof \rangle$

lemma *SIod-sub-sub*:
$[\![S \subseteq T;\ T \subseteq carrier\ D]\!] \Longrightarrow SIod\ (SIod\ D\ T)\ S = SIod\ D\ S$
$\langle proof \rangle$

lemma *rel-SIod*:$[\![Order\ D;\ Order\ E;\ carrier\ E \subseteq carrier\ D;$
$\forall a \in carrier\ E.\ \forall b \in carrier\ E.\ (a \preceq_E\ b) = (a \preceq_D\ b)]\!] \Longrightarrow$
$rel\ E = rel\ (SIod\ D\ (carrier\ E))$
$\langle proof \rangle$

lemma *SIod-self-le*:$[\![Order\ D;\ Order\ E;$
$carrier\ E \subseteq carrier\ D;$
$\forall a \in carrier\ E.\ \forall b \in carrier\ E.\ (a \preceq_E\ b) = (a \preceq_D\ b)\ ]\!] \Longrightarrow$
$E = SIod\ D\ (carrier\ E)$
$\langle proof \rangle$

2.1.1 Total ordering

locale *Torder* = *Order* +
assumes *le-linear*: $[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$a \preceq b \lor b \preceq a$

lemma (**in** *Order*) *Iod-empty-Torder*:$Torder\ (Iod\ D\ \{\})$
$\langle proof \rangle$

lemma (**in** *Torder*) *le-cases*:
$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;\ (a \preceq b \Longrightarrow C);\ (b \preceq a \Longrightarrow C)]\!] \Longrightarrow C$
⟨*proof*⟩

lemma (**in** *Torder*) *Order*:*Order D*
⟨*proof*⟩

lemma (**in** *Torder*) *less-linear*:
$a \in \mathit{carrier}\ D \Longrightarrow b \in \mathit{carrier}\ D \Longrightarrow a \prec b \lor a = b \lor b \prec a$
⟨*proof*⟩

lemma (**in** *Torder*) *not-le-less*:
$a \in \mathit{carrier}\ D \Longrightarrow b \in \mathit{carrier}\ D \Longrightarrow$
$(\neg\ a \preceq b) = (b \prec a)$
⟨*proof*⟩

lemma (**in** *Torder*) *not-less-le*:
$a \in \mathit{carrier}\ D \Longrightarrow b \in \mathit{carrier}\ D \Longrightarrow$
$(\neg\ a \prec b) = (b \preceq a)$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-not-le-less*:$[\![T \subseteq \mathit{carrier}\ D;\ a \in T;\ b \in T;$
$\mathit{Torder}\ (\mathit{Iod}\ D\ T)]\!] \Longrightarrow (\neg\ a \preceq_{(\mathit{Iod}\ D\ T)} b) = b \prec_{(\mathit{Iod}\ D\ T)} a$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-not-less-le*:$[\![T \subseteq \mathit{carrier}\ D;\ a \in T;\ b \in T;$
$\mathit{Torder}\ (\mathit{Iod}\ D\ T)]\!] \Longrightarrow (\neg\ a \prec_{(\mathit{Iod}\ D\ T)} b) = b \preceq_{(\mathit{Iod}\ D\ T)} a$
⟨*proof*⟩

2.1.2 Two ordered sets

definition
Order-Pow :: *'a set* ⇒ *'a set Order* ((*po* -) [*999*] *1000*) **where**
po A =
(|*carrier* = *Pow A*,
rel = $\{(X,\ Y).\ X \in \mathit{Pow}\ A \land Y \in \mathit{Pow}\ A \land X \subseteq Y\}$|)

interpretation *order-Pow*: *Order po A*
⟨*proof*⟩

definition
Order-fs :: *'a set* ⇒ *'b set* ⇒ (*'a set* ∗ (*'a* ⇒ *'b*)) *Order* **where**
Order-fs A B =
(|*carrier* = $\{Z.\ \exists A1\ f.\ A1 \in \mathit{Pow}\ A \land f \in A1 \to B \land$
$f \in \mathit{extensional}\ A1 \land Z = (A1, f)\}$,
rel = $\{Y.\ Y \in (\{Z.\ \exists A1\ f.\ A1 \in \mathit{Pow}\ A \land f \in A1 \to B \land f \in \mathit{extensional}\ A1$
$\land Z = (A1, f)\}) \times (\{Z.\ \exists A1\ f.\ A1 \in \mathit{Pow}\ A \land f \in A1 \to B \land f \in \mathit{extensional}$
$A1$
$\land Z = (A1, f)\}) \land \mathit{fst}\ (\mathit{fst}\ Y) \subseteq \mathit{fst}\ (\mathit{snd}\ Y) \land$

$(\forall a\in (fst\ (fst\ Y)).\ (snd\ (fst\ Y))\ a = (snd\ (snd\ Y))\ a)\}|)$

lemma *Order-fs*:*Order* (*Order-fs A B*)
⟨*proof*⟩

2.1.3 Homomorphism of ordered sets

definition
ord-inj :: [('a, 'm0) *Order-scheme*, ('b, 'm1) *Order-scheme*,
'a ⇒ 'b] ⇒ *bool* **where**
ord-inj D E f ⟷ *f* ∈ *extensional* (*carrier D*) ∧
f ∈ (*carrier D*) → (*carrier E*) ∧
(*inj-on f* (*carrier D*)) ∧
$(\forall a\in carrier\ D.\ \forall b\in carrier\ D.\ (a \prec_D b) = ((f\ a) \prec_E (f\ b)))$

definition
ord-isom :: [('a, 'm0) *Order-scheme*, ('b, 'm1) *Order-scheme*,
'a ⇒ 'b] ⇒ *bool* **where**
ord-isom D E f ⟷ *ord-inj D E f* ∧
(*surj-to f* (*carrier D*) (*carrier E*))

lemma (**in** *Order*) *ord-inj-func*:⟦*Order E*; *ord-inj D E f*⟧ ⟹
f ∈ *carrier D* → *carrier E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-func*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
f ∈ *carrier D* → *carrier E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-inj-restrict-isom*:⟦*Order E*; *ord-inj D E f*; *T* ⊆ *carrier D*⟧
⟹ *ord-isom* (*Iod D T*) (*Iod E* (*f* ‘ *T*)) (*restrict f T*)
⟨*proof*⟩

lemma *ord-inj-Srestrict-isom*:⟦*Order D*; *Order E*; *ord-inj D E f*; *T* ⊆ *carrier D*⟧
⟹ *ord-isom* (*SIod D T*) (*SIod E* (*f* ‘ *T*)) (*restrict f T*)
⟨*proof*⟩

lemma (**in** *Order*) *id-ord-isom*:*ord-isom D D* (*idmap* (*carrier D*))
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-bij-to*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
bij-to f (*carrier D*) (*carrier E*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-inj-mem*:⟦*Order E*; *ord-inj D E f*; *a* ∈ *carrier D*⟧ ⟹
(*f a*) ∈ *carrier E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-mem*:⟦*Order E*; *ord-isom D E f*; *a* ∈ *carrier D*⟧ ⟹

$(f\ a) \in carrier\ E$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-surj*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f;\ b \in carrier\ E]\!] \Longrightarrow$
$\exists\, a{\in}carrier\ D.\ b = f\ a$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-surj-forall*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f]\!] \Longrightarrow$
$\forall\, b \in carrier\ E.\ \exists\, a{\in}carrier\ D.\ b = f\ a$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-onto*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f]\!] \Longrightarrow$
$f\ `\ (carrier\ D) = carrier\ E$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-inj-on*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f]\!] \Longrightarrow$
$inj\text{-}on\ f\ (carrier\ D)$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-inj*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f;$
$a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow (a = b) = ((f\ a) = (f\ b))$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-surj-to*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f]\!] \Longrightarrow$
$surj\text{-}to\ f\ (carrier\ D)\ (carrier\ E)$
⟨proof⟩

lemma (**in** *Order*) *ord-inj-less*:$[\![Order\ E;\ ord\text{-}inj\ D\ E\ f;\ a \in carrier\ D;$
$b \in carrier\ D]\!] \Longrightarrow (a \prec_D b) = ((f\ a) \prec_E (f\ b))$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-less*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f;$
$a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow (a \prec_D b) = ((f\ a) \prec_E (f\ b))$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-less-forall*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f]\!] \Longrightarrow$
$\forall\, a \in carrier\ D.\ \forall\ b \in carrier\ D.\ (a \prec_D b) = ((f\ a) \prec_E (f\ b))$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-le*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f;$
$a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow (a \preceq_D b) = ((f\ a) \preceq_E (f\ b))$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-le-forall*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f]\!] \Longrightarrow$
$\forall\, a \in carrier\ D.\ \forall\ b \in carrier\ D.\ (a \preceq b) = ((f\ a) \preceq_E (f\ b))$
⟨proof⟩

lemma (**in** *Order*) *ord-isom-convert*:$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f;$
$x \in carrier\ D;\ a \in carrier\ D]\!] \Longrightarrow (\forall\, y{\in}carrier\ D.\ (x \prec y \longrightarrow \neg\ y \prec a)) =$

($\forall z \in$ *carrier E*. $((f\ x) \prec_E z \longrightarrow \neg\ z \prec_E (f\ a)))$
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-sym*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
ord-isom E D (*invfun* (*carrier D*) (*carrier E*) *f*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-trans*:⟦*Order E*; *Order F*; *ord-isom D E f*;
ord-isom E F g ⟧ ⟹ *ord-isom D F* (*compose* (*carrier D*) *g f*)
⟨*proof*⟩

definition
ord-equiv :: [-, ('*b*, '*m1*) *Order-scheme*] ⇒ *bool* **where**
ord-equiv D E ⟷ (∃*f*. *ord-isom D E f*)

lemma (**in** *Order*) *ord-equiv*:⟦*Order E*; *ord-isom D E f*⟧ ⟹ *ord-equiv D E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-isom*:⟦*Order E*; *ord-equiv D E*⟧ ⟹
∃*f*. *ord-isom D E f*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-reflex*:*ord-equiv D D*
⟨*proof*⟩

lemma (**in** *Order*) *eq-ord-equiv*:⟦*Order E*; *D* = *E*⟧ ⟹ *ord-equiv D E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-sym*:⟦*Order E*; *ord-equiv D E* ⟧ ⟹ *ord-equiv E D*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-trans*:⟦*Order E*; *Order F*; *ord-equiv D E*;
ord-equiv E F⟧ ⟹ *ord-equiv D F*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-box*:⟦*Order E*; *Order F*; *ord-equiv D E*;
ord-equiv D F⟧ ⟹ *ord-equiv E F*
⟨*proof*⟩

lemma *SIod-isom-Iod*:⟦*Order D*; *T* ⊆ *carrier D* ⟧ ⟹
ord-isom (*SIod D T*) (*Iod D T*) ($\lambda x \in T.\ x$)
⟨*proof*⟩

definition
minimum-elem :: [- , '*a set*, '*a*] ⇒ *bool* **where**
minimum-elem = ($\lambda D\ X\ a.\ a \in X \wedge (\forall x \in X.\ a \preceq_D x)$)

locale *Worder* = *Torder* +
assumes *ex-minimum*: ∀ *X*. *X* ⊆ (*carrier D*) ∧ *X* ≠ {} ⟶

($\exists x.$ *minimum-elem D X x*)

lemma (**in** *Worder*) *Order*: *Order D*
⟨*proof*⟩

lemma (**in** *Worder*) *Torder*: *Torder D*
⟨*proof*⟩

lemma (**in** *Worder*) *Worder*: *Worder D*
⟨*proof*⟩

lemma (**in** *Worder*) *equiv-isom*: $[\![$*Worder E*; *ord-equiv D E*$]\!] \Longrightarrow$
$\exists f.$ *ord-isom D E f*
⟨*proof*⟩

lemma (**in** *Order*) *minimum-elem-mem*: $[\![X \subseteq$ *carrier D*; *minimum-elem D X a*$]\!]$
$\Longrightarrow a \in X$
⟨*proof*⟩

lemma (**in** *Order*) *minimum-elem-unique*: $[\![X \subseteq$ *carrier D*; *minimum-elem D X a1*;
minimum-elem D X a2$]\!] \Longrightarrow a1 = a2$
⟨*proof*⟩

lemma (**in** *Order*) *compare-minimum-elements*: $[\![S \subseteq$ *carrier D*; $T \subseteq$ *carrier D*;
$S \subseteq T$; *minimum-elem D S s*; *minimum-elem D T t* $]\!] \Longrightarrow t \preceq s$
⟨*proof*⟩

lemma (**in** *Order*) *minimum-elem-sub*: $[\![T \subseteq$ *carrier D*; $X \subseteq T]\!]$
$\Longrightarrow$ *minimum-elem D X a* = *minimum-elem* (*Iod D T*) *X a*
⟨*proof*⟩

lemma *minimum-elem-Ssub*: $[\![$*Order D*; $T \subseteq$ *carrier D*; $X \subseteq T]\!]$
$\Longrightarrow$ *minimum-elem D X a* = *minimum-elem* (*SIod D T*) *X a*
⟨*proof*⟩

lemma (**in** *Order*) *augmented-set-minimum*: $[\![X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$Y - \{a\} \subseteq X$; $y - \{a\} \neq \{\}$; *minimum-elem* (*Iod D X*) ($Y - \{a\}$) *x*;
$\forall x \in X.\ x \preceq a]\!] \Longrightarrow$ *minimum-elem* (*Iod D* (*insert a X*)) *Y x*
⟨*proof*⟩

lemma *augmented-Sset-minimum*: $[\![$*Order D*; $X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$Y - \{a\} \subseteq X$; $y - \{a\} \neq \{\}$; *minimum-elem* (*SIod D X*) ($Y - \{a\}$) *x*;
$\forall x \in X.\ x \preceq_D a]\!] \Longrightarrow$ *minimum-elem* (*SIod D* (*insert a X*)) *Y x*
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-minimum*: $[\![$*Order E*; *ord-isom D E f*;
$S \subseteq$ *carrier D*; $a \in$ *carrier D*; *minimum-elem D S a*$]\!] \Longrightarrow$
minimum-elem E (*f`S*) (*f a*)

⟨proof⟩

lemma (**in** *Worder*) *pre-minimum*: $[\![T \subseteq carrier\ D;\ minimum\text{-}elem\ D\ T\ t;$
$s \in carrier\ D;\ s \prec_D t\ ]\!] \Longrightarrow \neg\ s \in T$
⟨proof⟩

lemma *bex-nonempty-subset*: $\exists\, a.\ a \in A \wedge P\ a \Longrightarrow$
$\{x.\ x \in A \wedge P\ x\} \subseteq A \wedge \{x.\ x \in A \wedge P\ x\} \neq \{\}$
⟨proof⟩

lemma (**in** *Worder*) *to-subset*: $[\![T \subseteq carrier\ D;\ ord\text{-}isom\ D\ (Iod\ D\ T)\ f]\!] \Longrightarrow$
$\forall\, a.\ a \in carrier\ D \longrightarrow a \preceq (f\ a)$
⟨proof⟩

lemma *to-subsetS*: $[\![Worder\ D;\ T \subseteq carrier\ D;\ ord\text{-}isom\ D\ (SIod\ D\ T)\ f]\!] \Longrightarrow$
$\forall\, a.\ a \in carrier\ D \longrightarrow a \preceq_D (f\ a)$
⟨proof⟩

lemma (**in** *Worder*) *isom-Worder*: $[\![Order\ T;\ ord\text{-}isom\ D\ T\ f]\!] \Longrightarrow Worder\ T$
⟨proof⟩

lemma (**in** *Worder*) *equiv-Worder*: $[\![Order\ T;\ ord\text{-}equiv\ D\ T]\!] \Longrightarrow Worder\ T$
⟨proof⟩

lemma (**in** *Worder*) *equiv-Worder1*: $[\![Order\ T;\ ord\text{-}equiv\ T\ D]\!] \Longrightarrow Worder\ T$
⟨proof⟩

lemma (**in** *Worder*) *ord-isom-self-id*: $ord\text{-}isom\ D\ D\ f \Longrightarrow f = idmap\ (carrier\ D)$
⟨proof⟩

lemma (**in** *Worder*) *isom-unique*: $[\![Worder\ E;\ ord\text{-}isom\ D\ E\ f;\ ord\text{-}isom\ D\ E\ g]\!]$
$\Longrightarrow f = g$
⟨proof⟩

definition
segment :: - $\Rightarrow$ 'a $\Rightarrow$ 'a *set* **where**
segment $D\ a = (if\ a \notin carrier\ D\ then\ carrier\ D\ else$
$\{x.\ x \prec_D a \wedge x \in carrier\ D\})$

definition
Ssegment :: 'a *Order* $\Rightarrow$ 'a $\Rightarrow$ 'a *set* **where**
Ssegment $D\ a = (if\ a \notin carrier\ D\ then\ carrier\ D\ else$
$\{x.\ x \prec_D a \wedge x \in carrier\ D\})$

lemma (**in** *Order*) *segment-sub*: $segment\ D\ a \subseteq carrier\ D$
⟨proof⟩

lemma *Ssegment-sub*: $Ssegment\ D\ a \subseteq carrier\ D$
⟨proof⟩

lemma (**in** *Order*) *segment-free*:$a \notin carrier\ D \Longrightarrow$
$segment\ D\ a = carrier\ D$
⟨*proof*⟩

lemma *Ssegment-free*:$a \notin carrier\ D \Longrightarrow$
$Ssegment\ D\ a = carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *segment-sub-sub*:$[\![S \subseteq carrier\ D;\ d \in S]\!] \Longrightarrow$
$segment\ (Iod\ D\ S)\ d \subseteq segment\ D\ d$
⟨*proof*⟩

lemma *Ssegment-sub-sub*:$[\![Order\ D;\ S \subseteq carrier\ D;\ d \in S]\!] \Longrightarrow$
$Ssegment\ (SIod\ D\ S)\ d \subseteq Ssegment\ D\ d$
⟨*proof*⟩

lemma (**in** *Order*) *a-notin-segment*:$a \notin segment\ D\ a$
⟨*proof*⟩

lemma *a-notin-Ssegment*:$a \notin Ssegment\ D\ a$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-carr-segment*:
$carrier\ (Iod\ D\ (segment\ D\ a)) = segment\ D\ a$
⟨*proof*⟩

lemma *SIod-carr-Ssegment*:$Order\ D \Longrightarrow$
$carrier\ (SIod\ D\ (Ssegment\ D\ a)) = Ssegment\ D\ a$
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \prec b) = (a \in segment\ D\ b)$
⟨*proof*⟩

lemma *Ssegment-inc*:$[\![Order\ D;\ a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \prec_D b) = (a \in Ssegment\ D\ b)$
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc1*:$b \in carrier\ D \Longrightarrow$
$(a \prec b \wedge a \in carrier\ D) = (a \in segment\ D\ b)$
⟨*proof*⟩

lemma *Ssegment-inc1*:$[\![Order\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \prec_D b \wedge a \in carrier\ D) = (a \in Ssegment\ D\ b)$
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc-if*:$[\![b \in carrier\ D; a \in segment\ D\ b]\!] \Longrightarrow$
$a \prec b$

⟨*proof*⟩

lemma *Ssegment-inc-if*:$[\![Order\ D;\ b \in carrier\ D;\ a \in Ssegment\ D\ b]\!] \Longrightarrow a \prec_D b$
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc-less*:$[\![W \subseteq carrier\ D;\ a \in carrier\ D;\ y \in W;\ x \in segment\ (Iod\ D\ W)\ a;\ y \prec x]\!] \Longrightarrow y \in segment\ (Iod\ D\ W)\ a$
⟨*proof*⟩

lemma (**in** *Order*) *segment-order-less*:$\forall b\in carrier\ D.\ \forall x\in segment\ D\ b.\ \forall y\in segment\ D\ b.\ (x \prec y) = (x \prec_{(Iod\ D\ (segment\ D\ b))} y)$
⟨*proof*⟩

lemma *Ssegment-order-less*:$Order\ D \Longrightarrow \forall b\in carrier\ D.\ \forall x\in Ssegment\ D\ b.\ \forall y\in Ssegment\ D\ b.\ (x \prec_D y) = (x \prec_{(SIod\ D\ (Ssegment\ D\ b))} y)$
⟨*proof*⟩

lemma (**in** *Order*) *segment-order-le*:$\forall b\in carrier\ D.\ \forall x\in segment\ D\ b.\ \forall y\in segment\ D\ b.\ (x \preceq y) = (x \preceq_{(Iod\ D\ (segment\ D\ b))} y)$
⟨*proof*⟩

lemma *Ssegment-order-le*:$\forall b\in carrier\ D.\ \forall x\in Ssegment\ D\ b.\ \forall y\in Ssegment\ D\ b.\ (x \preceq_D y) = (x \preceq_{(SIod\ D\ (Ssegment\ D\ b))} y)$
⟨*proof*⟩

lemma (**in** *Torder*) *Iod-Torder*:$X \subseteq carrier\ D \Longrightarrow Torder\ (Iod\ D\ X)$
⟨*proof*⟩

lemma *SIod-Torder*:$[\![Torder\ D;\ X \subseteq carrier\ D]\!] \Longrightarrow Torder\ (SIod\ D\ X)$
⟨*proof*⟩

lemma (**in** *Order*) *segment-not-inc*:$[\![a \in carrier\ D;\ b \in carrier\ D;\ a \prec b]\!] \Longrightarrow b \notin segment\ D\ a$
⟨*proof*⟩

lemma *Ssegment-not-inc*:$[\![Order\ D;\ a \in carrier\ D;\ b \in carrier\ D;\ a \prec_D b]\!] \Longrightarrow b \notin Ssegment\ D\ a$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-not-inc-iff*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow (a \preceq b) = (b \notin segment\ D\ a)$
⟨*proof*⟩

lemma *Ssegment-not-inc-iff*:$[\![Torder\ D;\ a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow (a \preceq_D b) = (b \notin Ssegment\ D\ a)$
⟨*proof*⟩

lemma (**in** *Torder*) *minimum-segment-of-sub*:$[\![X \subseteq carrier\ D;$
$minimum\text{-}elem\ D\ (segment\ (Iod\ D\ X)\ d)\ m\ ]\!] \Longrightarrow minimum\text{-}elem\ D\ X\ m$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-out*:$[\![a \in carrier\ D;\ b \in carrier\ D;$
$a \prec b]\!] \Longrightarrow segment\ (Iod\ D\ (segment\ D\ a))\ b = segment\ D\ a$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-minimum-minimum*:$[\![X \subseteq carrier\ D;\ d \in X;$
$minimum\text{-}elem\ (Iod\ D\ (segment\ D\ d))\ (X \cap (segment\ D\ d))\ m]\!] \Longrightarrow$
$minimum\text{-}elem\ D\ X\ m$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-mono*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \prec b) = (segment\ D\ a \subset segment\ D\ b)$
⟨*proof*⟩

lemma *Ssegment-mono*:$[\![Torder\ D;\ a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \prec_D b) = (Ssegment\ D\ a \subset Ssegment\ D\ b)$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-le-mono*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \preceq b) = (segment\ D\ a \subseteq segment\ D\ b)$
⟨*proof*⟩

lemma *Ssegment-le-mono*:$[\![Torder\ D;\ a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \preceq_D b) = (Ssegment\ D\ a \subseteq Ssegment\ D\ b)$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-inj*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a = b) = (segment\ D\ a = segment\ D\ b)$
⟨*proof*⟩

lemma *Ssegment-inj*:$[\![Torder\ D;\ a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a = b) = (Ssegment\ D\ a = Ssegment\ D\ b)$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-inj-neq*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \neq b) = (segment\ D\ a \neq segment\ D\ b)$
⟨*proof*⟩

lemma *Ssegment-inj-neq*:$[\![Torder\ D;\ a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \neq b) = (Ssegment\ D\ a \neq Ssegment\ D\ b)$
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc-psub*:$[\![x \in segment\ D\ a]\!] \Longrightarrow$
$segment\ D\ x \subset segment\ D\ a$
⟨*proof*⟩

lemma *Ssegment-inc-psub*:$[\![$*Order D*; $x \in$ *Ssegment D a*$]\!] \Longrightarrow$
Ssegment D x $\subset$ *Ssegment D a*
⟨*proof*⟩

lemma (**in** *Order*) *segment-segment*:$[\![b \in$ *carrier D*; $a \in$ *segment D b*$]\!] \Longrightarrow$
segment (*Iod D* (*segment D b*)) *a* = *segment D a*
⟨*proof*⟩

lemma *Ssegment-Ssegment*:$[\![$*Order D*; $b \in$ *carrier D*; $a \in$ *Ssegment D b*$]\!] \Longrightarrow$
Ssegment (*SIod D* (*Ssegment D b*)) *a* = *Ssegment D a*
⟨*proof*⟩

lemma (**in** *Order*) *Iod-segment-segment*:$a \in$ *carrier* (*Iod D* (*segment D b*)) $\Longrightarrow$
Iod (*Iod D* (*segment D b*)) (*segment* (*Iod D* (*segment D b*)) *a*) =
Iod D (*segment D a*)
⟨*proof*⟩

lemma *SIod-Ssegment-Ssegment*:$[\![$*Order D*; $a \in$ *carrier* (*SIod D* (*Ssegment D b*))$]\!]$

$\Longrightarrow$
SIod (*SIod D* (*Ssegment D b*)) (*Ssegment* (*SIod D* (*Ssegment D b*)) *a*) =
SIod D (*Ssegment D a*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-segment-mem*:$[\![$*Order E*;
ord-isom D E f; $a \in$ *carrier D*; $x \in$ *segment D a* $]\!] \Longrightarrow$
(*f x*) $\in$ *segment E* (*f a*)

⟨*proof*⟩

lemma *ord-isom-Ssegment-mem*:$[\![$*Order D*; *Order E*;
ord-isom D E f; $a \in$ *carrier D*; $x \in$ *Ssegment D a*$]\!] \Longrightarrow$
(*f x*) $\in$ *Ssegment E* (*f a*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-segment-segment*:$[\![$*Order E*;
ord-isom D E f; $a \in$ *carrier D* $]\!] \Longrightarrow$
ord-isom (*Iod D* (*segment D a*)) (*Iod E* (*segment E* (*f a*)))
($\lambda x \in$*carrier* (*Iod D* (*segment D a*)). *f x*)
⟨*proof*⟩

lemma *ord-isom-Ssegment-Ssegment*:$[\![$*Order D*; *Order E*;
ord-isom D E f; $a \in$ *carrier D* $]\!] \Longrightarrow$
ord-isom (*SIod D* (*Ssegment D a*)) (*SIod E* (*Ssegment E* (*f a*)))
($\lambda x \in$*carrier* (*SIod D* (*Ssegment D a*)). *f x*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-segment-segment*:

$[\![$*Order E*; *ord-equiv D E*; $a \in$ *carrier D*$]\!]$
$\Longrightarrow \exists\, t{\in}$*carrier E*. *ord-equiv* (*Iod D* (*segment D a*)) (*Iod E* (*segment E t*))

⟨*proof*⟩

lemma *ord-equiv-Ssegment-Ssegment*:
$[\![$*Order D*; *Order E*; *ord-equiv D E*; $a \in$ *carrier D*$]\!]$
$\Longrightarrow \exists\, t{\in}$*carrier E*. *ord-equiv* (*SIod D* (*Ssegment D a*)) (*SIod E* (*Ssegment E t*))
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-restricted*:
$[\![$*Order E*; *ord-isom D E f*; *D1* $\subseteq$ *carrier D*$]\!] \Longrightarrow$
ord-isom (*Iod D D1*) (*Iod E* (*f* ‘ *D1*)) ($\lambda x{\in}$*D1*. *f x*)
⟨*proof*⟩

lemma *ord-isom-restrictedS*:
$[\![$*Order D*; *Order E*; *ord-isom D E f*; *D1* $\subseteq$ *carrier D*$]\!] \Longrightarrow$
ord-isom (*SIod D D1*) (*SIod E* (*f* ‘ *D1*)) ($\lambda x{\in}$*D1*. *f x*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-induced*:
$[\![$*Order E*; *ord-isom D E f*; *D1* $\subseteq$ *carrier D* $]\!] \Longrightarrow$
ord-equiv (*Iod D D1*) (*Iod E* (*f* ‘ *D1*))
⟨*proof*⟩

lemma *ord-equiv-inducedS*:
$[\![$*Order D*; *Order E*; *ord-isom D E f*; *D1* $\subseteq$ *carrier D* $]\!] \Longrightarrow$
ord-equiv (*SIod D D1*) (*SIod E* (*f* ‘ *D1*))
⟨*proof*⟩

lemma (**in** *Order*) *equiv-induced-by-inj*:$[\![$*Order E*; *ord-inj D E f*;
D1 $\subseteq$ *carrier D*$]\!] \Longrightarrow$ *ord-equiv* (*Iod D D1*) (*Iod E* (*f* ‘ *D1*))
⟨*proof*⟩

lemma *equiv-induced-by-injS*:$[\![$*Order D*; *Order E*; *ord-inj D E f*;
D1 $\subseteq$ *carrier D*$]\!] \Longrightarrow$ *ord-equiv* (*SIod D D1*) (*SIod E* (*f* ‘ *D1*))
⟨*proof*⟩

lemma (**in** *Torder*) *le-segment-segment*:$[\![a \in$ *carrier D*; $b \in$ *carrier D*$]\!] \Longrightarrow$
$(a \preceq b)$ = (*segment* (*Iod D* (*segment D b*)) *a* = *segment D a*)
⟨*proof*⟩

lemma *le-Ssegment-Ssegment*:$[\![$*Torder D*; $a \in$ *carrier D*; $b \in$ *carrier D*$]\!] \Longrightarrow$
$(a \preceq_D b)$ = (*Ssegment* (*SIod D* (*Ssegment D b*)) *a* = *Ssegment D a*)
⟨*proof*⟩

lemma (**in** *Torder*) *inc-segment-segment*:$[\![b \in$ *carrier D*;
$a \in$ *segment D b*$]\!] \Longrightarrow$ *segment* (*Iod D* (*segment D b*)) *a* = *segment D a*

⟨*proof*⟩

lemma (**in** *Torder*) *segment-segment*:⟦$a \in carrier\ D$; $b \in carrier\ D$⟧ $\Longrightarrow$
$(segment\ (Iod\ D\ (segment\ D\ b))\ a = segment\ D\ a) =$
$((segment\ D\ a) \subseteq (segment\ D\ b))$
⟨*proof*⟩

lemma (**in** *Torder*) *less-in-Iod*:⟦$a \in carrier\ D$; $b \in carrier\ D$; $a \prec b$⟧
$\Longrightarrow (a \prec b) = (a \in carrier\ (Iod\ D\ (segment\ D\ b)))$
⟨*proof*⟩

definition
SS :: - ⇒ ′*a set Order* **where**
$SS\ D = (\!|carrier = \{X.\ \exists\, a{\in}carrier\ D.\ X = segment\ D\ a\},\ rel =$
$\{XX.\ XX \in \{X.\ \exists\, a{\in}carrier\ D.\ X = segment\ D\ a\} \times$
$\{X.\ \exists\, a{\in}carrier\ D.\ X = segment\ D\ a\} \wedge ((fst\ XX) \subseteq (snd\ XX))\}\ |\!)$

definition
segmap::- ⇒ ′*a* ⇒ ′*a set* **where**
$segmap\ D = (\lambda x{\in}(carrier\ D).\ segment\ D\ x)$

lemma *segmap-func*:$segmap\ D \in carrier\ D \to carrier\ (SS\ D)$
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-segmap*: *ord-isom D (SS D) (segmap D)*
⟨*proof*⟩

lemma (**in** *Worder*) *nonequiv-segment*:$a \in carrier\ D \Longrightarrow$
$\neg$ *ord-equiv D (Iod D (segment D a))*
⟨*proof*⟩

lemma *nonequiv-Ssegment*:⟦*Worder D*; $a \in carrier\ D$⟧ $\Longrightarrow$
$\neg$ *ord-equiv D (SIod D (Ssegment D a))*
⟨*proof*⟩

lemma (**in** *Worder*) *subset-Worder*: $T \subseteq carrier\ D \Longrightarrow$
Worder (Iod D T)
⟨*proof*⟩

lemma *SIod-Worder*:⟦*Worder D*; $T \subseteq carrier\ D$⟧ $\Longrightarrow$ *Worder (SIod D T)*
⟨*proof*⟩

lemma (**in** *Worder*) *segment-Worder*:*Worder (Iod D (segment D a))*
⟨*proof*⟩

lemma *Ssegment-Worder*:*Worder D* $\Longrightarrow$ *Worder (SIod D (Ssegment D a))*
⟨*proof*⟩

lemma (**in** *Worder*) *segment-unique1*:$[\![a \in carrier\ D;\ b \in carrier\ D;\ a \prec b]\!] \Longrightarrow$
$\neg$ *ord-equiv* (*Iod D* (*segment D b*)) (*Iod D* (*segment D a*))
⟨*proof*⟩

lemma *Ssegment-unique1*:$[\![Worder\ D;\ a \in carrier\ D;\ b \in carrier\ D;\ a \prec_D b]\!] \Longrightarrow$
$\neg$ *ord-equiv* (*SIod D* (*Ssegment D b*)) (*SIod D* (*Ssegment D a*))
⟨*proof*⟩

lemma (**in** *Worder*) *segment-unique*:$[\![a \in carrier\ D;\ b \in carrier\ D;$
ord-equiv (*Iod D* (*segment D a*)) (*Iod D* (*segment D b*)) $]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma *Ssegment-unique*:$[\![Worder\ D;\ a \in carrier\ D;\ b \in carrier\ D;$
ord-equiv (*SIod D* (*Ssegment D a*)) (*SIod D* (*Ssegment D b*)) $]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *Worder*) *subset-segment*:$[\![T \subseteq carrier\ D;$
$\forall b{\in}T.\ \forall x.\ x \prec b \wedge x \in carrier\ D \longrightarrow x \in T;$
minimum-elem D (*carrier D* $-$ *T*) $a]\!] \Longrightarrow T = segment\ D\ a$
⟨*proof*⟩

lemma *subset-Ssegment*:$[\![Worder\ D;\ T \subseteq carrier\ D;$
$\forall b{\in}T.\ \forall x.\ x \prec_D b \wedge x \in carrier\ D \longrightarrow x \in T;$
minimum-elem D (*carrier D* $-$ *T*) $a]\!] \Longrightarrow T = Ssegment\ D\ a$
⟨*proof*⟩

lemma (**in** *Worder*) *segmentTr*:$[\![T \subseteq carrier\ D;$
$\forall b \in T.\ (\forall x.\ (x \prec b \wedge x \in (carrier\ D) \longrightarrow x \in T))]\!] \Longrightarrow$
$(T = carrier\ D) \vee (\exists a.\ a \in (carrier\ D) \wedge T = segment\ D\ a)$
⟨*proof*⟩

lemma *SsegmentTr*:$[\![Worder\ D;\ T \subseteq carrier\ D;$
$\forall b \in T.\ (\forall x.\ (x \prec_D b \wedge x \in (carrier\ D) \longrightarrow x \in T))]\!] \Longrightarrow$
$(T = carrier\ D) \vee (\exists a.\ a \in (carrier\ D) \wedge T = Ssegment\ D\ a)$
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-segment-segment*:$[\![Worder\ E;$
ord-isom D E f; $a \in carrier\ D$ $]\!] \Longrightarrow$
ord-isom (*Iod D* (*segment D a*)) (*Iod E* (*segment E* (*f a*)))
($\lambda x{\in}carrier$ (*Iod D* (*segment D a*)). *f x*)
⟨*proof*⟩

definition
Tw :: [- , ('b, 'm1) *Order-scheme*] ⇒ 'a ⇒ 'b ((*2Tw*$_{-,-}$) [*60,61*]*60*) **where**
$Tw_{D,T} = (\lambda a{\in}\ carrier\ D.\ SOME\ x.\ x{\in}carrier\ T\ \wedge$
ord-equiv (*Iod D* (*segment D a*)) (*Iod T* (*segment T x*)))

lemma (**in** *Worder*) *Tw-func*:⟦*Worder T*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier T*. *ord-equiv* (*Iod D* (*segment D a*))
(*Iod T* (*segment T b*))⟧ ⟹ $Tw_{D,T}$ ∈ *carrier D* → *carrier T*
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-mem*:⟦*Worder E*; *x* ∈ *carrier D*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier E*. *ord-equiv* (*Iod D* (*segment D a*))
(*Iod E* (*segment E b*))⟧ ⟹ ($Tw_{D,E}$) *x* ∈ *carrier E*
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-equiv*:⟦*Worder T*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier T*. *ord-equiv* (*Iod D* (*segment D a*))
(*Iod T* (*segment T b*)); *x* ∈ *carrier D* ⟧ ⟹
ord-equiv (*Iod D* (*segment D x*)) (*Iod T* (*segment T* (($Tw_{D,T}$) *x*)))
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-inj*:⟦*Worder E*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier E*. *ord-equiv* (*Iod D* (*segment D a*))
(*Iod E* (*segment E b*))⟧ ⟹ *inj-on* ($Tw_{D,E}$) (*carrier D*)

⟨*proof*⟩

lemma (**in** *Worder*) *Tw-eq-ord-isom*:⟦*Worder E*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier E*.
ord-equiv (*Iod D* (*segment D a*)) (*Iod E* (*segment E b*)); *a* ∈ *carrier D*;
ord-isom (*Iod D* (*segment D a*)) (*Iod E* (*segment E* (*Tw D E a*))) *f*;
x ∈ *segment D a* ⟧ ⟹ *f x* = *Tw D E x*
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-ord-injTr*:⟦*Worder E*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier E*.
ord-equiv (*Iod D* (*segment D a*)) (*Iod E* (*segment E b*));
a ∈ *carrier D*; *b* ∈ *carrier D*; *a* ≺ *b*⟧ ⟹
Tw D E a $\prec_E$ (*Tw D E b*)
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-ord-inj*:⟦*Worder E*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier E*. *ord-equiv* (*Iod D* (*segment D a*))
(*Iod E* (*segment E b*))⟧ ⟹ *ord-inj D E* (*Tw D E*)
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-restricted-by-Tw*:⟦*Worder E*;
$\forall$ *a*∈*carrier D*. ∃ *b*∈*carrier E*.
ord-equiv (*Iod D* (*segment D a*)) (*Iod E* (*segment E b*));
D1 ⊆ *carrier D*⟧ ⟹
ord-isom (*Iod D D1*) (*Iod E* ((*Tw D E*) ‘ *D1*))
(*restrict* (*Tw D E*) *D1*)
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-segment-segment*:⟦*Worder E*;
∀ *a*∈*carrier D*.∃ *b*∈*carrier E*.
ord-equiv (*Iod D* (*segment D a*)) (*Iod E* (*segment E b*)); *a* ∈ *carrier D*⟧
⟹ *Tw D E* ‘ (*segment D a*) = *segment E* (*Tw D E a*)
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Tw-segment*:⟦*Worder E*;
∀ *a*∈*carrier D*. ∃ *b*∈*carrier E*.
ord-equiv (*Iod D* (*segment D a*)) (*Iod E* (*segment E b*)); *a*∈*carrier D*⟧ ⟹
ord-isom (*Iod D* (*segment D a*)) (*Iod E* (*segment E* (*Tw D E a*)))
(*restrict* (*Tw D E*) (*segment D a*))
⟨*proof*⟩

lemma (**in** *Worder*) *well-ord-compare1*:⟦*Worder E*;
∀ *a*∈*carrier D*. ∃ *b*∈*carrier E*.
ord-equiv (*Iod D* (*segment D a*)) (*Iod E* (*segment E b*))⟧ ⟹
(*ord-equiv D E*) ∨ (∃ *c*∈*carrier E*. *ord-equiv D* (*Iod E* (*segment E c*)))
⟨*proof*⟩

lemma *bex-nonempty-set*:∃ *x* ∈ *A*. *P x* ⟹ {*x*. *x* ∈ *A* ∧ *P x* } ≠ {}
⟨*proof*⟩

lemma *nonempty-set-sub*:{*x*. *x* ∈ *A* ∧ *P x* } ≠ {} ⟹
{*x*. *x* ∈ *A* ∧ *P x*} ⊆ *A*
⟨*proof*⟩

lemma (**in** *Torder*) *less-minimum*:⟦*minimum-elem D* {*x*. *x* ∈ *carrier D* ∧ *P x*}
d⟧
⟹ ∀ *a*. (((*a* ≺ *d*) ∧ *a* ∈ *carrier D*) ⟶ ¬ (*P a*))
⟨*proof*⟩

lemma (**in** *Torder*) *segment-minimum-empty*:⟦*X* ⊆ *carrier D*; *d* ∈ *X*⟧ ⟹
(*minimum-elem D X d*) = (*segment* (*Iod D X*) *d* = {})
⟨*proof*⟩

end

theory *Algebra2*
imports *Algebra1*
begin

lemma (**in** *Order*) *less-and-segment*:*b* ∈ *carrier D* ⟹
(∀ *a*.((*a* ≺ *b* ∧ *a* ∈ *carrier D*) ⟶ (*Q a*))) =
(∀ *a*∈*carrier* (*Iod D* (*segment D b*)).(*Q a*))
⟨*proof*⟩

lemma (**in** *Worder*) *Word-compare2*:⟦*Worder E*;
¬ (∀ *a*∈*carrier D*. ∃ *b*∈*carrier E*. *ord-equiv* (*Iod D* (*segment D a*))

$(Iod\ E\ (segment\ E\ b)))]\!] \Longrightarrow$
$\exists\, c \in carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ c))\ \ E$
⟨proof⟩

lemma (**in** *Worder*) *Worder-equiv*: $[\![Worder\ E;$
$\forall\, a \in carrier\ D.\ \exists\, b \in carrier\ E.\ \ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))$
$(Iod\ E\ (segment\ E\ b));$
$\forall\, c \in carrier\ E.\ \exists\, d \in carrier\ D.\ \ ord\text{-}equiv\ (Iod\ E\ (segment\ E\ c))$
$(Iod\ D\ (segment\ D\ d))]\!] \Longrightarrow ord\text{-}equiv\ D\ E$
⟨proof⟩

lemma (**in** *Worder*) *Worder-equiv1*: $[\![Worder\ E;\ \neg\ ord\text{-}equiv\ D\ E]\!] \Longrightarrow$
$\neg\ ((\forall\, a \in carrier\ D.\ \exists\, b \in carrier\ E.$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ b)))\ \wedge$
$(\forall\, c \in carrier\ E.\ \exists\, d \in carrier\ D.$
$ord\text{-}equiv\ (Iod\ E\ (segment\ E\ c))\ (Iod\ D\ (segment\ D\ d))))$
⟨proof⟩

lemma (**in** *Worder*) *Word-compare*: $Worder\ E \Longrightarrow$
$(\exists\, a \in carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ E) \vee ord\text{-}equiv\ D\ E\ \vee$
$(\exists\, b \in carrier\ E.\ ord\text{-}equiv\ D\ (Iod\ E\ (segment\ E\ b)))$
⟨proof⟩

lemma (**in** *Worder*) *Word-compareTr1*: $[\![Worder\ E;$
$\exists\, a \in carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ E;\ ord\text{-}equiv\ D\ E\ ]\!] \Longrightarrow$
$False$
⟨proof⟩

lemma (**in** *Worder*) *Word-compareTr2*: $[\![Worder\ E;\ ord\text{-}equiv\ D\ E;$
$\exists\, b \in carrier\ E.\ ord\text{-}equiv\ D\ (Iod\ E\ (segment\ E\ b))]\!] \Longrightarrow False$
⟨proof⟩

lemma (**in** *Worder*) *Word-compareTr3*: $[\![Worder\ E;$
$\exists\, b \in carrier\ E.\ ord\text{-}equiv\ D\ (Iod\ E\ (segment\ E\ b));$
$\exists\, a \in carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ E]\!] \Longrightarrow False$
⟨proof⟩

lemma (**in** *Worder*) *subset-equiv-segment*: $S \subseteq carrier\ D \Longrightarrow$
$ord\text{-}equiv\ D\ (Iod\ D\ S)\ \vee$
$(\exists\, a \in carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ S)\ (Iod\ D\ (segment\ D\ a)))$
⟨proof⟩

definition
ordinal-number :: *'a Order* ⇒ *'a Order set* **where**
$ordinal\text{-}number\ S = \{X.\ Worder\ X \wedge ord\text{-}equiv\ X\ S\}$

definition
ODnums :: *'a Order set set* **where**
$ODnums = \{X.\ \exists\, S.\ Worder\ S \wedge X = ordinal\text{-}number\ S\}$

definition
ODord :: [*'a Order set*, *'a Order set*] ⇒ *bool* (**infix** ⊏ *60*) **where**
$X \sqsubset Y \longleftrightarrow (\exists x \in X. \exists y \in Y. (\exists c{\in}carrier\ y.\ ord\text{-}equiv\ x\ (Iod\ y\ (segment\ y\ c))))$

definition
ODord-le :: [*'a Order set*, *'a Order set*] ⇒ *bool* (**infix** ⊑ *60*) **where**
$X \sqsubseteq Y \longleftrightarrow X = Y \vee ODord\ X\ Y$

definition
ODrel :: (((*'a Order*) *set*) ∗ ((*'a Order*) *set*)) *set* **where**
$ODrel = \{Z.\ Z \in ODnums \times ODnums \wedge ODord\text{-}le\ (fst\ Z)\ (snd\ Z)\}$

definition
ODnods :: (*'a Order set*) *Order* **where**
ODnods = (|*carrier* = *ODnums*, *rel* = *ODrel* |)

lemma *Worder-ord-equivTr*:⟦*Worder S*; *Worder T*⟧ ⟹
ord-equiv S T = (∃*f*. *ord-isom S T f*)
⟨*proof*⟩

lemma *Worder-ord-isom-mem*:⟦*Worder S*; *Worder T*; *ord-isom S T f*; *a* ∈ *carrier S*⟧
⟹ *f a* ∈ *carrier T*
⟨*proof*⟩

lemma *Worder-refl*: *Worder S* ⟹ *ord-equiv S S*
⟨*proof*⟩

lemma *Worder-sym*:⟦*Worder S*; *Worder T*; *ord-equiv S T* ⟧ ⟹ *ord-equiv T S*
⟨*proof*⟩

lemma *Worder-trans*:⟦*Worder S*; *Worder T*; *Worder U*; *ord-equiv S T*; *ord-equiv T U*⟧ ⟹ *ord-equiv S U*
⟨*proof*⟩

lemma *ordinal-inc-self*: *Worder S* ⟹ *S* ∈ *ordinal-number S*
⟨*proof*⟩

lemma *ordinal-number-eq*:⟦*Worder D*; *Worder E*⟧ ⟹
(*ord-equiv D E*) = (*ordinal-number D* = *ordinal-number E*)
⟨*proof*⟩

lemma *mem-ordinal-number-equiv*:⟦*Worder D*;
X ∈ *ordinal-number D*⟧ ⟹ *ord-equiv X D*
⟨*proof*⟩

lemma *mem-ordinal-number-Worder*:⟦*Worder D*;

$X \in$ *ordinal-number* $D]\!] \Longrightarrow$ *Worder* X
⟨*proof*⟩

lemma *mem-ordinal-number-Worder1*:$[\![x \in ODnums;\ X \in x]\!] \Longrightarrow Worder\ X$
⟨*proof*⟩

lemma *mem-ODnums-nonempty*:$X \in ODnums \Longrightarrow \exists x.\ x \in X$
⟨*proof*⟩

lemma *carr-ODnods*:$carrier\ ODnods = ODnums$
⟨*proof*⟩

lemma *ordinal-number-mem-carrier-ODnods*:
$Worder\ D \Longrightarrow ordinal\text{-}number\ D \in carrier\ ODnods$
⟨*proof*⟩

lemma *ordinal-number-mem-ODnums*:
$Worder\ D \Longrightarrow ordinal\text{-}number\ D \in ODnums$
⟨*proof*⟩

lemma *ODordTr1*: $[\![Worder\ D;\ Worder\ E]\!] \Longrightarrow$
$(ODord\ (ordinal\text{-}number\ D)\ (ordinal\text{-}number\ E)) =$
$(\exists b\in carrier\ E.\ ord\text{-}equiv\ D\ (Iod\ E\ (segment\ E\ b)))$
⟨*proof*⟩

lemma *ODord*: $[\![Worder\ D;\ d \in carrier\ D]\!] \Longrightarrow$
$ODord\ (ordinal\text{-}number\ (Iod\ D\ (segment\ D\ d)))\ (ordinal\text{-}number\ D)$
⟨*proof*⟩

lemma *ord-less-ODord*:$[\![Worder\ D;\ c \in carrier\ D;\ d \in carrier\ D;$
$a = ordinal\text{-}number\ (Iod\ D\ (segment\ D\ c));$
$b = ordinal\text{-}number\ (Iod\ D\ (segment\ D\ d))]\!] \Longrightarrow$
$c \prec_D d = a \sqsubset b$
⟨*proof*⟩

lemma *ODord-le-ref*:$[\![X \in ODnums;\ Y \in ODnums;\ ODord\text{-}le\ X\ Y;\ Y \sqsubseteq X\ ]\!]$
$\Longrightarrow$
$X = Y$
⟨*proof*⟩

lemma *ODord-le-trans*:$[\![X \in ODnums;\ Y \in ODnums;\ Z \in ODnums;\ X \sqsubseteq Y;\ Y$
$\sqsubseteq Z\ ]\!]$
$\Longrightarrow X \sqsubseteq Z$
⟨*proof*⟩

lemma *ordinal-numberTr1*: $X \in carrier\ ODnods \Longrightarrow \exists D.\ Worder\ D \wedge D \in X$
⟨*proof*⟩

lemma *ordinal-numberTr1-1*: $X \in ODnums \Longrightarrow \exists D.\ Worder\ D \wedge D \in X$

⟨*proof*⟩

lemma *ordinal-numberTr1-2*:$[\![x \in ODnums; S \in x; T \in x]\!] \Longrightarrow$
$ord\text{-}equiv\ S\ T$
⟨*proof*⟩

lemma *ordinal-numberTr2*:$[\![Worder\ D; x = ordinal\text{-}number\ D]\!] \Longrightarrow$
$D \in x$
⟨*proof*⟩

lemma *ordinal-numberTr3*:$[\![Worder\ D; Worder\ F; ord\text{-}equiv\ D\ F;$
$x = ordinal\text{-}number\ D]\!] \Longrightarrow x = ordinal\text{-}number\ F$
⟨*proof*⟩

lemma *ordinal-numberTr4*:$[\![Worder\ D; X \in carrier\ ODnods; D \in X\ ]\!] \Longrightarrow$
$X = ordinal\text{-}number\ D$
⟨*proof*⟩

lemma *ordinal-numberTr5*:$[\![x \in ODnums; D \in x]\!] \Longrightarrow x = ordinal\text{-}number\ D$
⟨*proof*⟩

lemma *ordinal-number-ord*:$[\![\ X \in carrier\ ODnods; Y \in carrier\ ODnods]\!] \Longrightarrow$
$ODord\ X\ Y \vee X = Y \vee ODord\ Y\ X$
⟨*proof*⟩

lemma *ODnum-subTr*:$[\![Worder\ D; x = ordinal\text{-}number\ D; y \in ODnums; y \sqsubset x;$
$Y \in y]\!]$
$\Longrightarrow \exists\, c{\in}carrier\ D.\ ord\text{-}equiv\ Y\ (Iod\ D\ (segment\ D\ c))$
⟨*proof*⟩

lemma *ODnum-segmentTr*:$[\![Worder\ D; x = ordinal\text{-}number\ D; y \in ODnums; y \sqsubset$
$x]\!] \Longrightarrow$
$\exists\, c.\ c{\in}carrier\ D \wedge (\forall\, Y{\in}y.\ ord\text{-}equiv\ Y\ (Iod\ D\ (segment\ D\ c)))$
⟨*proof*⟩

lemma *ODnum-segmentTr1*:$[\![Worder\ D; x = ordinal\text{-}number\ D; y \in ODnums; y$
$\sqsubset x]\!]$
$\Longrightarrow \exists\, c.\ c \in carrier\ D \wedge (y = ordinal\text{-}number\ (Iod\ D\ (segment\ D\ c)))$
⟨*proof*⟩

lemma *ODnods-less*:$[\![x \in carrier\ ODnods; y \in carrier\ ODnods]\!] \Longrightarrow$
$x \prec_{ODnods} y = x \sqsubset y$
⟨*proof*⟩

lemma *ODord-less-not-eq*:$[\![x \in carrier\ ODnods; y \in carrier\ ODnods; x \sqsubset y]\!] \Longrightarrow$
$x \neq y$
⟨*proof*⟩

lemma *not-ODord*:$[\![a \in ODnums; b \in ODnums; a \sqsubset b]\!] \Longrightarrow \neg\ (b \sqsubseteq a)$

⟨*proof*⟩

lemma *Order-ODnods*: *Order ODnods*
⟨*proof*⟩

lemma *Torder-ODnods*: *Torder ODnods*
⟨*proof*⟩

definition
ODNmap :: *'a Order* ⇒ (*'a Order*) *set* ⇒ *'a* **where**
ODNmap D y = (*SOME z*. (*z* ∈ *carrier D* ∧
(∀ *Y*∈*y*. *ord-equiv Y* (*Iod D* (*segment D z*)))))

lemma *ODNmap-mem*:⟦*Worder D*; *x* = *ordinal-number D*; *y* ∈ *ODnums*; *ODord y x*⟧ ⟹
ODNmap D y ∈ *carrier D* ∧
(∀ *Y*∈*y*. *ord-equiv Y* (*Iod D* (*segment D* (*ODNmap D y*))))
⟨*proof*⟩

lemma *ODNmapTr1*:⟦*Worder D*; *x* = *ordinal-number D*; *y* ∈ *ODnums*; *ODord y x*⟧ ⟹
y = *ordinal-number* (*Iod D* (*segment D* (*ODNmap D y*)))
⟨*proof*⟩

lemma *ODNmap-self*:⟦*Worder D*; *c* ∈ *carrier D*;
a = *ordinal-number* (*Iod D* (*segment D c*))⟧ ⟹ *ODNmap D a* = *c*
⟨*proof*⟩

lemma *ODord-ODNmap-less*:⟦*Worder D*; *c* ∈ *carrier D*;
a = *ordinal-number* (*Iod D* (*segment D c*)); *d* ∈ *carrier D*;
b = *ordinal-number* (*Iod D* (*segment D d*)); *a* ⊏ *b*⟧ ⟹
ODNmap D a $\prec_D$ (*ODNmap D b*)
⟨*proof*⟩

lemma *ODNmap-mem1*: ⟦*Worder D*; *y* ∈ *segment ODnods* (*ordinal-number D*)⟧
⟹ *ODNmap D y* ∈ *carrier D*
⟨*proof*⟩

lemma *ODnods-segment-inc-ODord*:⟦*Worder D*;
y ∈ *segment ODnods* (*ordinal-number D*)⟧ ⟹ *ODord y* (*ordinal-number D*)
⟨*proof*⟩

lemma *restict-ODNmap-func*:⟦*Worder D*; *x* = *ordinal-number D*⟧ ⟹
restrict (*ODNmap D*) (*segment ODnods* (*ordinal-number D*))
∈ *segment ODnods* (*ordinal-number D*) → *carrier D*
⟨*proof*⟩

lemma *ODNmap-ord-isom*:⟦*Worder D*; *x* = *ordinal-number D*⟧ ⟹
ord-isom (*Iod ODnods* (*segment ODnods x*)) *D*

$(\lambda x \in (carrier\ (Iod\ ODnods\ (segment\ ODnods\ x))).\ (ODNmap\ D\ x))$
⟨proof⟩

lemma *ODnum-equiv-segment*:$[\![Worder\ D;\ x = ordinal\text{-}number\ D]\!] \Longrightarrow$
$ord\text{-}equiv\ (Iod\ ODnods\ (segment\ ODnods\ x))\ D$
⟨proof⟩

lemma *ODnods-sub-carrier*:$S \subseteq ODnums \Longrightarrow carrier\ (Iod\ ODnods\ S) = S$
⟨proof⟩

lemma *ODnum-sub-well-ordered*:$S \subseteq ODnums \Longrightarrow Worder\ (Iod\ ODnods\ S)$
⟨proof⟩

2.2 Pre elements

definition
$ExPre :: \text{-} \Rightarrow 'a \Rightarrow bool$ **where**
$ExPre\ D\ a \longleftrightarrow (\exists x.\ x \in carrier\ D \wedge x \prec_D a$
$\wedge \neg (\exists y \in carrier\ D.\ x \prec_D y \wedge y \prec_D a))$

definition
$Pre :: [\text{-} ,\ 'a] \Rightarrow 'a$ **where**
$Pre\ D\ a = (SOME\ x.\ x \in carrier\ D \wedge$
$x \prec_D a \wedge$
$\neg (\exists y \in carrier\ D.\ x \prec_D y \wedge y \prec_D a))$

lemma (**in** *Order*) *Pre-mem*:$[\![a \in carrier\ D;\ ExPre\ D\ a]\!] \Longrightarrow$
$Pre\ D\ a \in carrier\ D$
⟨proof⟩

lemma (**in** *Order*) *Not-ExPre*:$a \in carrier\ D \Longrightarrow \neg\ ExPre\ (Iod\ D\ \{a\})\ a$
⟨proof⟩

lemma (**in** *Worder*) *UniquePre*:$[\![a \in carrier\ D;\ ExPre\ D\ a;$
$a1 \in carrier\ D \wedge a1 \prec a \wedge \neg (\exists y \in carrier\ D.\ (a1 \prec y \wedge y \prec a))\ ]\!] \Longrightarrow$
$Pre\ D\ a = a1$
⟨proof⟩

lemma (**in** *Order*) *Pre-element*:$[\![a \in carrier\ D;\ ExPre\ D\ a]\!] \Longrightarrow$
$Pre\ D\ a \in carrier\ D \wedge (Pre\ D\ a) \prec a \wedge$
$\neg (\exists y \in carrier\ D.\ ((Pre\ D\ a) \prec y \wedge y \prec a))$
⟨proof⟩

lemma (**in** *Order*) *Pre-in-segment*:$[\![a \in carrier\ D;\ ExPre\ D\ a]\!] \Longrightarrow$
$Pre\ D\ a \in segment\ D\ a$
⟨proof⟩

lemma (**in** *Worder*) *segment-forall*:⟦$a \in$ *segment D b*; $b \in$ *carrier D*;
$x \in$ *segment D b*; $x \prec a$; $\forall y \in$*segment D b*. $x \prec y \longrightarrow \neg\, y \prec a$⟧ $\Longrightarrow$
$\forall y \in$*carrier D*. $x \prec y \longrightarrow \neg\, y \prec a$
⟨*proof*⟩

lemma (**in** *Worder*) *segment-Expre*:$a \in$ *segment D b* $\Longrightarrow$
ExPre (*Iod D* (*segment D b*)) a = *ExPre D a*
⟨*proof*⟩

lemma (**in** *Worder*) *Pre-segment*:⟦$a \in$ *segment D b*;
ExPre (*Iod D* (*segment D b*)) a⟧ $\Longrightarrow$
ExPre D a $\wedge$ *Pre D a* = *Pre* (*Iod D* (*segment D b*)) a
⟨*proof*⟩

lemma (**in** *Worder*) *Pre2segment*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*; $b \prec a$;
ExPre D b⟧ $\Longrightarrow$ *ExPre* (*Iod D* (*segment D a*)) b
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Pre1*:⟦*Worder E*; $a \in$ *carrier D*; *ExPre D a*;
ord-isom D E f⟧ $\Longrightarrow$ *ExPre E* (*f a*)
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Pre11*:⟦*Worder E*; $a \in$ *carrier D*; *ord-isom D E f*⟧
$\Longrightarrow$ *ExPre D a* = *ExPre E* (*f a*)
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Pre2*:⟦*Worder E*; $a \in$ *carrier D*; *ExPre D a*;
ord-isom D E f⟧ $\Longrightarrow$ *f* (*Pre D a*) = *Pre E* (*f a*)
⟨*proof*⟩

2.3 Transfinite induction

lemma (**in** *Worder*) *transfinite-induction*:⟦*minimum-elem D* (*carrier D*) *s0*; *P s0*;
$\forall t \in$*carrier D*. (($\forall u \in$ *segment D t*. *P u*) $\longrightarrow$ *P t*)⟧ $\Longrightarrow$ $\forall x \in$*carrier D*. *P x*
⟨*proof*⟩

2.4 *Ordered-set2*. Lemmas to prove Zorn's lemma.

definition
adjunct-ord ::[- , 'a] ⇒ - **where**
adjunct-ord D a = *D* (|*carrier* := *carrier D* ∪ {*a*},
rel := {(*x*,*y*). (*x*, *y*) ∈ *rel D* ∨
(*x* ∈ (*carrier D* ∪ {*a*}) ∧ *y* = *a*)}|)

lemma (**in** *Order*) *carrier-adjunct-ord*:
carrier (*adjunct-ord D a*) = *carrier D* ∪ {*a*}
⟨*proof*⟩

lemma (**in** *Order*) *Order-adjunct-ord*: $a \notin carrier\ D \Longrightarrow$
$Order\ (adjunct\text{-}ord\ D\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *adjunct-ord-large-a*: $[\![Order\ D;\ a \notin carrier\ D]\!] \Longrightarrow$
$\forall x \in carrier\ D.\ x \prec_{adjunct\text{-}ord\ D\ a}\ a$
⟨*proof*⟩

lemma *carr-Ssegment-adjunct-ord*: $[\![Order\ D;\ a \notin carrier\ D]\!] \Longrightarrow$
$carrier\ D = (Ssegment\ (adjunct\text{-}ord\ D\ a)\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *adjunct-ord-selfD*: $a \notin carrier\ D \Longrightarrow$
$D = Iod\ (adjunct\text{-}ord\ D\ a)\ (carrier\ D)$
⟨*proof*⟩

lemma *Ssegment-adjunct-ord*: $[\![Order\ D;\ a \notin carrier\ D]\!] \Longrightarrow$
$D = SIod\ (adjunct\text{-}ord\ D\ a)\ (Ssegment\ (adjunct\text{-}ord\ D\ a)\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *Torder-adjunction*: $[\![X \subseteq carrier\ D;\ a \in carrier\ D;$
$\forall x \in X.\ x \preceq a;\ Torder\ (Iod\ D\ X)]\!] \Longrightarrow Torder\ (Iod\ D\ (X \cup \{a\}))$
⟨*proof*⟩

lemma *Torder-Sadjunction*: $[\![Order\ D;\ X \subseteq carrier\ D;\ a \in carrier\ D;$
$\forall x \in X.\ x \preceq_D a;\ Torder\ (SIod\ D\ X)]\!] \Longrightarrow Torder\ (SIod\ D\ (X \cup \{a\}))$
⟨*proof*⟩

lemma (**in** *Torder*) *Torder-adjunct-ord*: $a \notin carrier\ D \Longrightarrow$
$Torder\ (adjunct\text{-}ord\ D\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *well-ord-adjunction*: $[\![X \subseteq carrier\ D;\ a \in carrier\ D;$
$\forall x \in X.\ x \preceq a;\ Worder\ (Iod\ D\ X)]\!] \Longrightarrow Worder\ (Iod\ D\ (X \cup \{a\}))$
⟨*proof*⟩

lemma *well-ord-Sadjunction*: $[\![Order\ D;\ X \subseteq carrier\ D;\ a \in carrier\ D;$
$\forall x \in X.\ x \preceq_D a;\ Worder\ (SIod\ D\ X)]\!] \Longrightarrow Worder\ (SIod\ D\ (X \cup \{a\}))$
⟨*proof*⟩

lemma (**in** *Worder*) *Worder-adjunct-ord*: $a \notin carrier\ D \Longrightarrow$
$Worder\ (adjunct\text{-}ord\ D\ a)$
⟨*proof*⟩

2.5 Zorn's lemma

definition
Chain :: - ⇒ *'a set* ⇒ *bool* **where**
$Chain\ D\ C \longleftrightarrow C \subseteq carrier\ D \land Torder\ (Iod\ D\ C)$

definition
upper-bound :: [-, 'a set, 'a] ⇒ bool
((*3ub*ı/ -/ -) [100,101]100) **where**
ub_D S b ⟷ b ∈ carrier D ∧ (∀ s∈S. s $\preceq_D$ b)

definition
inductive-set :: - ⇒ bool **where**
inductive-set D ⟷ (∀ C. (Chain D C ⟶ (∃ b. ub_D C b)))

definition
maximal-element :: [-, 'a] ⇒ bool ((*maximal*ı/ -) [101]100) **where**
$maximal_D$ m ⟷ m ∈ carrier D ∧ (∀ b∈carrier D. m $\preceq_D$ b ⟶ m = b)

definition
upper-bounds::[-, 'a set] ⇒ 'a set **where**
upper-bounds D H = {x. ub_D H x}

definition
Sup :: [-, 'a set] ⇒ 'a **where**
Sup D X = (THE x. minimum-elem D (upper-bounds D X) x)

definition
S-inductive-set :: - ⇒ bool **where**
S-inductive-set D ⟷ (∀ C. Chain D C ⟶
(∃ x∈carrier D. minimum-elem D (upper-bounds D C) x))

lemma (**in** *Order*) *mem-upper-bounds*:⟦X ⊆ carrier D; b ∈ carrier D;
∀ x∈X. x ≼ b⟧ ⟹ ub X b
⟨*proof*⟩

lemma (**in** *Order*) *Torder-Chain*:⟦X ⊆ carrier D; Torder (Iod D X)⟧
⟹ Chain D X
⟨*proof*⟩

lemma (**in** *Order*) *Chain-Torder*:Chain D X ⟹
Torder (Iod D X)
⟨*proof*⟩

lemma (**in** *Order*) *Chain-sub*:Chain D X ⟹ X ⊆ carrier D
⟨*proof*⟩

lemma (**in** *Order*) *Chain-sub-Chain*:⟦Chain D X; Y ⊆ X ⟧ ⟹ Chain D Y
⟨*proof*⟩

lemma (**in** *Order*) *upper-bounds-sub*:X ⊆ carrier D ⟹
upper-bounds D X ⊆ carrier D
⟨*proof*⟩

lemma (**in** *Order*) *Sup*:$[\![X \subseteq carrier\ D;\ minimum\text{-}elem\ D\ (upper\text{-}bounds\ D\ X)\ a]\!] \Longrightarrow$ $Sup\ D\ X = a$
⟨*proof*⟩

lemma (**in** *Worder*) *Sup-mem*:$[\![X \subseteq carrier\ D;\ \exists\, b.\ ub\ X\ b]\!] \Longrightarrow$
$Sup\ D\ X \in carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup*:$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X]\!] \Longrightarrow$
$minimum\text{-}elem\ D\ (upper\text{-}bounds\ D\ X)\ (Sup\ D\ X)$
⟨*proof*⟩

lemma (**in** *Order*) *adjunct-Chain*:$[\![Chain\ D\ X;\ b \in carrier\ D;\ \forall\, x{\in}X.\ x \preceq b]\!] \Longrightarrow$
$Chain\ D\ (insert\ b\ X)$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup-mem*:$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X]\!] \Longrightarrow$
$Sup\ D\ X \in carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-Sup-min-bounds*:$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X;$
$ub\ X\ b]\!] \Longrightarrow\ Sup\ D\ X \preceq b$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup-bound*:$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X]\!] \Longrightarrow$
$\forall\, x{\in}X.\ x \preceq (Sup\ D\ X)$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-Sup-in-ChainTr*:
$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X;\ c \in carrier\ (Iod\ D\ (insert\ (Sup\ D\ X)\ X));$
$Sup\ D\ X \notin X;$
$\forall\, y{\in}carrier\ (Iod\ D\ (insert\ (Sup\ D\ X)\ X)).$
$c \prec_{Iod\ D\ (insert\ (Sup\ D\ X)\ X)}\ y \longrightarrow \neg\ y \prec_{Iod\ D\ (insert\ (Sup\ D\ X)\ X)}\ Sup\ D$
$X]\!] \Longrightarrow$
$c \in upper\text{-}bounds\ D\ X$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-Sup-in-Chain*:$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X;$
$ExPre\ (Iod\ D\ (insert\ (Sup\ D\ X)\ X))\ (Sup\ D\ X)]\!] \Longrightarrow Sup\ D\ X \in X$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-bounds-compare*:$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X1;$
$Chain\ D\ X2;\ X1 \subseteq X2]\!] \Longrightarrow upper\text{-}bounds\ D\ X2 \subseteq upper\text{-}bounds\ D\ X1$
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup-compare*:$[\![S\text{-}inductive\text{-}set\ D;\ Chain\ D\ X1;$
$Chain\ D\ X2;\ X1 \subseteq X2]\!] \Longrightarrow Sup\ D\ X1 \preceq\ Sup\ D\ X2$
⟨*proof*⟩

definition
Wa :: [-, $'a$ set, $'a \Rightarrow 'a$, $'a$] $\Rightarrow$ $bool$ **where**
$Wa\ D\ W\ g\ a \longleftrightarrow W \subseteq carrier\ D \wedge Worder\ (Iod\ D\ W) \wedge a \in W \wedge (\forall x \in W.\ a \preceq_D x) \wedge$
$(\forall x \in W.\ (if\ (ExPre\ (Iod\ D\ W)\ x)\ then\ \ g\ (Pre\ (Iod\ D\ W)\ x) = x\ else$
$(if\ a \neq x\ then\ Sup\ D\ (segment\ (Iod\ D\ W)\ x) = x\ else\ a = a)))$

definition
WWa :: [-, $'a \Rightarrow 'a$, $'a$] $\Rightarrow$ $'a$ set set **where**
$WWa\ D\ g\ a = \{W.\ Wa\ D\ W\ g\ a\}$

lemma (**in** $Order$) $mem\text{-}of\text{-}WWa$:$[\![W \subseteq carrier\ D;\ Worder\ (Iod\ D\ W);\ a \in W;$
$(\forall x \in W.\ a \preceq x);$
$(\forall x \in W.\ (if\ (ExPre\ (Iod\ D\ W)\ x)\ then\ \ g\ (Pre\ (Iod\ D\ W)\ x) = x\ else$
$(if\ a \neq x\ then\ Sup\ D\ (segment\ (Iod\ D\ W)\ x) = x\ else\ a = a)))]\!] \Longrightarrow$
$W \in WWa\ D\ g\ a$
⟨*proof*⟩

lemma (**in** $Order$) $mem\text{-}WWa\text{-}then$:$W \in WWa\ D\ g\ a \Longrightarrow W \subseteq carrier\ D \wedge$
$Worder\ (Iod\ D\ W) \wedge a \in W \wedge (\forall x \in W.\ a \preceq x) \wedge$
$(\forall x \in W.\ (if\ (ExPre\ (Iod\ D\ W)\ x)\ then\ \ g\ (Pre\ (Iod\ D\ W)\ x) = x\ else$
$(if\ a \neq x\ then\ Sup\ D\ (segment\ (Iod\ D\ W)\ x) = x\ else\ a = a)))$
⟨*proof*⟩

lemma (**in** $Order$) $mem\text{-}wwa\text{-}Worder$:$W \in WWa\ D\ g\ a \Longrightarrow Worder\ (Iod\ D\ W)$
⟨*proof*⟩

lemma (**in** $Order$) $mem\text{-}WWa\text{-}sub\text{-}carrier$:$W \in WWa\ D\ g\ a \Longrightarrow W \subseteq carrier\ D$
⟨*proof*⟩

lemma (**in** $Order$) $Union\text{-}WWa\text{-}sub\text{-}carrier$:$\bigcup\ (WWa\ D\ g\ a) \subseteq carrier\ D$
⟨*proof*⟩

lemma (**in** $Order$) $mem\text{-}WWa\text{-}inc\text{-}a$:$W \in WWa\ D\ g\ a \Longrightarrow a \in W$
⟨*proof*⟩

lemma (**in** $Order$) $mem\text{-}WWa\text{-}Chain$:$W \in WWa\ D\ g\ a \Longrightarrow Chain\ D\ W$
⟨*proof*⟩

lemma (**in** $Order$) $Sup\text{-}adjunct\text{-}Sup$:$[\![S\text{-}inductive\text{-}set\ D;$
$f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;\ \forall x \in carrier\ D.\ x \preceq f\ x;$
$W \in WWa\ D\ f\ a;\ Sup\ D\ W \notin W]\!]$
$\Longrightarrow Sup\ D\ (insert\ (Sup\ D\ W)\ W) = Sup\ D\ W$

⟨*proof*⟩

lemma (**in** $Order$) $BNTr1$:$a \in carrier\ D \Longrightarrow Worder\ (Iod\ D\ \{a\})$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr2*:⟦$f \in carrier\ D \rightarrow carrier\ D$; $a \in carrier\ D$;
$\forall x{\in}carrier\ D.\ x \preceq (f\ x)$⟧ $\Longrightarrow \{a\} \in WWa\ D\ f\ a$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr2-1*:⟦$f \in carrier\ D \rightarrow carrier\ D$; $a \in carrier\ D$;
$\forall x{\in}carrier\ D.\ x \preceq (f\ x)$; $W \in WWa\ D\ f\ a$⟧ $\Longrightarrow \forall x{\in}W.\ a \preceq x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr3*:⟦$f \in carrier\ D \rightarrow carrier\ D$; $a \in carrier\ D$;
$\forall x{\in}carrier\ D.\ x \preceq (f\ x)$; $W \in WWa\ D\ f\ a$⟧ $\Longrightarrow$ *minimum-elem* (*Iod D W*)
W a

⟨*proof*⟩

lemma (**in** *Order*) *Adjunct-segment-sub*:⟦*S-inductive-set D*; *Chain D X*⟧ $\Longrightarrow$
segment (*Iod D* (*insert* (*Sup D X*) *X*)) (*Sup D X*) $\subseteq$ *X*
⟨*proof*⟩

lemma (**in** *Order*) *Adjunct-segment-eq*:⟦*S-inductive-set D*; *Chain D X*;
Sup D X $\notin$ *X*⟧ $\Longrightarrow$
segment (*Iod D* (*insert* (*Sup D X*) *X*)) (*Sup D X*) = *X*
⟨*proof*⟩

definition
fixp :: $['a \Rightarrow 'a, 'a] \Rightarrow bool$ **where**
fixp f a $\longleftrightarrow$ $f\ a = a$

lemma (**in** *Order*) *fixp-same*:⟦$W1 \subseteq carrier\ D$; $W2 \subseteq carrier\ D$; $t \in W1$;
$b \in carrier\ D$; *ord-isom* (*Iod D W1*) (*Iod* (*Iod D W2*) (*segment* (*Iod D W2*) *b*))
g;
$\forall u{\in}$*segment* (*Iod D W1*) *t*. *fixp g u*⟧ $\Longrightarrow$
segment (*Iod D W1*) *t* = *segment* (*Iod D W2*) (*g t*)

⟨*proof*⟩

lemma (**in** *Order*) *BNTr4-1*:⟦$f \in carrier\ D \rightarrow carrier\ D$; $a \in carrier\ D$;
$b \in carrier\ D$; $\forall x{\in}carrier\ D.\ x \preceq (f\ x)$; $W1 \in WWa\ D\ f\ a$; $W2 \in WWa\ D\ f$
a;
ord-isom (*Iod D W1*) (*Iod D* (*segment* (*Iod D W2*) *b*)) *g*⟧ $\Longrightarrow$
$\forall x{\in}W1.\ g\ x = x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr4-2*:⟦$f \in carrier\ D \rightarrow carrier\ D$; $a \in carrier\ D$;
$b \in carrier\ D$; $\forall x{\in}carrier\ D.\ x \preceq (f\ x)$; $W1 \in WWa\ D\ f\ a$; $W2 \in WWa\ D$
f a;
ord-equiv (*Iod D W1*) (*Iod D* (*segment* (*Iod D W2*) *b*))⟧ $\Longrightarrow$
W1 = *segment* (*Iod D W2*) *b*
⟨*proof*⟩

lemma (**in** *Order*) *BNTr4*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ \ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a;$
$\exists b \in carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ W1)\ (Iod\ D\ (segment\ (Iod\ D\ W2)\ b))]\!] \Longrightarrow$
$W1 \subseteq W2$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-same*: $A = B \Longrightarrow Iod\ D\ A = Iod\ D\ B$
⟨*proof*⟩

lemma (**in** *Order*) *eq-ord-equivTr*: $[\![ord\text{-}equiv\ D\ E;\ E = F]\!] \Longrightarrow ord\text{-}equiv\ D\ F$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr5*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a;$
$ord\text{-}equiv\ (Iod\ D\ W1)\ (Iod\ D\ W2)]\!] \Longrightarrow W1 \subseteq W2$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr6*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a;\ W1 \subset W2]\!] \Longrightarrow$
$(\exists b \in carrier\ (Iod\ D\ W2).\ ord\text{-}equiv\ (Iod\ D\ W1)\ (Iod\ D\ (segment\ (Iod\ D\ W2)\ b)))$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr6-1*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a;\ W1 \subset W2]\!] \Longrightarrow$
$(\exists b \in carrier\ (Iod\ D\ W2).\ W1 = (segment\ (Iod\ D\ W2)\ b))$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a]\!] \Longrightarrow$
$W1 \subseteq W2 \vee W2 \subseteq W1$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-1*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq f\ x;\ x \in W;\ W \in WWa\ D\ f\ a;\ xa \in \bigcup\ (WWa\ D\ f\ a);$
$xa \prec_{Iod\ D\ (\bigcup\ (WWa\ D\ f\ a))}\ x]\!] \Longrightarrow xa \in W$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-1-1*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq f\ x;\ x \in W;\ W \in WWa\ D\ f\ a;\ xa \in \bigcup\ (WWa\ D\ f\ a);$
$xa \prec x]\!] \Longrightarrow xa \in W$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-2*: $[\![f \in carrier\ D \rightarrow carrier\ D;\ a \in carrier\ D;$
$\forall x \in carrier\ D.\ x \preceq f\ x;\ x \in \bigcup (WWa\ D\ f\ a);\ ExPre\ (Iod\ D\ (\bigcup (WWa\ D\ f\ a)))$
$x\ ]\!]$
$\Longrightarrow \forall W \in WWa\ D\ f\ a.\ (x \in W \longrightarrow ExPre\ (Iod\ D\ W)\ x\ )$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-3*: $[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall x{\in}carrier\ D.\ x \preceq f\ x;\ x \in \bigcup(WWa\ D\ f\ a);\ ExPre\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))$
$x\ ]\!]$
$\Longrightarrow \forall\ W{\in}WWa\ D\ f\ a.\ (x \in W \longrightarrow Pre\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))\ x = Pre\ (Iod$
$D\ W)\ x)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-4*: $[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall x{\in}carrier\ D.\ x \preceq f\ x;\ x \in W;\ \ W \in WWa\ D\ f\ a]\!] \Longrightarrow$
$ExPre\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))\ x = ExPre\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-5*: $[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall x{\in}carrier\ D.\ x \preceq f\ x;\ x \in W;\ W \in WWa\ D\ f\ a]\!]$
$\Longrightarrow (segment\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))\ x) = segment\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-6*: $[\![f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x{\in}carrier\ D.\ x \preceq (f\ x)]\!]\ \Longrightarrow a \in \bigcup(WWa\ D\ f\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-7*: $[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x{\in}carrier\ D.\ x \preceq (f\ x);\ \exists xa.\ Wa\ D\ xa\ f\ a \wedge x \in xa]\!] \Longrightarrow$
$x \in \bigcup(WWa\ D\ f\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-8*: $[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;\ a \in$
$carrier\ D;\ \forall x{\in}carrier\ D.\ x \preceq (f\ x);\ \exists xa.\ Wa\ D\ xa\ f\ a \wedge x \in xa]\!] \Longrightarrow x \in carrier$
D
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-9*: $[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall x{\in}carrier\ D.\ x \preceq (f\ x);\ x \in \bigcup(WWa\ D\ f\ a)\ ]\!] \Longrightarrow x \in carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-10*: $[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x{\in}carrier\ D.\ x \preceq (f\ x);\ W \in WWa\ D\ f\ a;\ Sup\ D\ W \notin W]\!]$
$\Longrightarrow$
$\neg\ ExPre\ (Iod\ D\ (insert\ (Sup\ D\ W)\ W))\ (Sup\ D\ W)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-11*: $[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ b \in carrier\ D;\ \forall x{\in}carrier\ D.\ x \preceq f\ x;\ W \in WWa\ D\ f\ a;$
$\forall x{\in}W.\ x \preceq b;\ x \in W]\!] \Longrightarrow$
$ExPre\ (Iod\ D\ (insert\ b\ W))\ x = ExPre\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-12*: $[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$

$a \in carrier\ D;\ b \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq f\ x;\ W \in WWa\ D\ f\ a;$
$\forall x\in W.\ x \preceq b;\ x \in W;\ ExPre\ (Iod\ D\ W)\ x]\!] \Longrightarrow$
$Pre\ (Iod\ D\ (insert\ b\ W))\ x = Pre\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-13*:$[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ b \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq f\ x;\ W \in WWa\ D\ f\ a;$
$\forall x\in W.\ x \preceq b;\ x \in W]\!] \Longrightarrow$
$(segment\ (Iod\ D\ (insert\ b\ W))\ x) = segment\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-14*:$[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq (f\ x);\ W \in WWa\ D\ f\ a]\!] \Longrightarrow$
$(insert\ (Sup\ D\ W)\ W) \in WWa\ D\ f\ a$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-15*:$[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq (f\ x);\ W \in WWa\ D\ f\ a;$
$f\ (Sup\ D\ W) \neq Sup\ D\ W]\!] \Longrightarrow$
$ExPre\ (Iod\ D\ (insert\ (f\ (Sup\ D\ W))\ (insert\ (Sup\ D\ W)\ W)))\ (f\ (Sup\ D\ W))$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-16*:$[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq (f\ x);\ W \in WWa\ D\ f\ a;$
$f\ (Sup\ D\ W) \neq (Sup\ D\ W)]\!] \Longrightarrow$
$Pre\ (Iod\ D\ (insert\ (f\ (Sup\ D\ W))\ (insert\ (Sup\ D\ W)\ W)))\ (f\ (Sup\ D\ W))$
$=$
$(Sup\ D\ W)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-17*:$[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq (f\ x);\ W \in WWa\ D\ f\ a]\!] \Longrightarrow$
$(insert\ (f\ (Sup\ D\ W))\ (insert\ (Sup\ D\ W)\ W)) \in WWa\ D\ f\ a$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr8*:$[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall x\in carrier\ D.\ x \preceq (f\ x)]\!] \Longrightarrow \bigcup\ (WWa\ D\ f\ a) \in (WWa\ D\ f\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr10*:$[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq (f\ x)]\!] \Longrightarrow$
$(Sup\ D\ (\bigcup(WWa\ D\ f\ a))) \in (\bigcup(WWa\ D\ f\ a))$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr11*:$[\![S\text{-}inductive\text{-}set\ D;\ f \in carrier\ D \to carrier\ D;$
$a \in carrier\ D;\ \forall x\in carrier\ D.\ x \preceq (f\ x)]\!] \Longrightarrow$

$f\ (Sup\ D\ (\bigcup(WWa\ D\ f\ a))) = (Sup\ D\ (\bigcup(WWa\ D\ f\ a)))$
⟨proof⟩

lemma (**in** *Order*) *Bourbaki-Nakayama*:⟦*S-inductive-set D*;
$f \in carrier\ D \rightarrow carrier\ D$; $a \in carrier\ D$; $\forall x{\in}carrier\ D.\ x \preceq (f\ x)$⟧ $\Longrightarrow$
$\exists x0{\in}carrier\ D.\ f\ x0 = x0$
⟨proof⟩

definition
maxl-fun :: - ⇒ 'a ⇒ 'a **where**
maxl-fun D = ($\lambda x{\in}carrier\ D.$ *if* $\exists y.\ y{\in}(upper\text{-}bounds\ D\ \{x\}) \wedge y \neq x$ *then*
SOME $z.\ z \in (upper\text{-}bounds\ D\ \{x\}) \wedge z \neq x$ *else* x)

lemma (**in** *Order*) *maxl-funTr*:⟦$x \in carrier\ D$;
$\exists y.\ y \in upper\text{-}bounds\ D\ \{x\} \wedge y \neq x$⟧ $\Longrightarrow$
(*SOME* $z.\ z \in upper\text{-}bounds\ D\ \{x\} \wedge z \neq x$) $\in carrier\ D$
⟨proof⟩

lemma (**in** *Order*) *maxl-fun-func*:*maxl-fun D* $\in carrier\ D \rightarrow carrier\ D$
⟨proof⟩

lemma (**in** *Order*) *maxl-fun-gt*:⟦$x \in carrier\ D$;
$\exists y{\in}\ carrier\ D.\ \ x \preceq y \wedge x \neq y$ ⟧ $\Longrightarrow$
$x \preceq (maxl\text{-}fun\ D\ x) \wedge (maxl\text{-}fun\ D\ x) \neq x$
⟨proof⟩

lemma (**in** *Order*) *maxl-fun-maxl*:⟦$x \in carrier\ D$; *maxl-fun D x* $= x$ ⟧
$\Longrightarrow$ *maximal x*
⟨proof⟩

lemma (**in** *Order*) *maxl-fun-asc*:$\forall x{\in}carrier\ D.\ x \preceq (maxl\text{-}fun\ D\ x)$
⟨proof⟩

lemma (**in** *Order*) *S-inductive-maxl*:⟦*S-inductive-set D*; $carrier\ D \neq \{\}$⟧ $\Longrightarrow$
$\exists m.$ *maximal m*
⟨proof⟩

lemma (**in** *Order*) *maximal-mem*:*maximal m* $\Longrightarrow m \in carrier\ D$
⟨proof⟩

definition
Chains :: - ⇒ ('a set) set **where**
Chains D == $\{C.\ Chain\ D\ C\}$

definition
family-Torder:: - ⇒ ('a set) *Order*
((*fTo* -) [*999*]*1000*) **where**
fTo D = (|*carrier* = *Chains D* , *rel* = $\{Z.\ Z \in (Chains\ D) \times (Chains\ D) \wedge$ (*fst*

$Z) \subseteq (snd\ Z)\}$)

lemma (**in** *Order*) *Chain-mem-fTo*: $Chain\ D\ C \Longrightarrow C \in carrier\ (fTo\ D)$
⟨*proof*⟩

lemma (**in** *Order*) *fTOrder*: $Order\ (fTo\ D)$
⟨*proof*⟩

lemma (**in** *Order*) *fTo-Order-sub*: $[\![A \in carrier\ (fTo\ D);\ B \in carrier\ (fTo\ D)]\!]$
$\Longrightarrow (A \preceq_{(fTo\ D)} B) = (A \subseteq B)$
⟨*proof*⟩

lemma (**in** *Order*) *mem-fTo-Chain*: $X \in carrier\ (fTo\ D) \Longrightarrow Chain\ D\ X$
⟨*proof*⟩

lemma (**in** *Order*) *mem-fTo-sub-carrier*: $X \in carrier\ (fTo\ D) \Longrightarrow X \subseteq carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *Un-fTo-Chain*: $Chain\ (fTo\ D)\ CC \Longrightarrow Chain\ D\ (\bigcup\ CC)$
⟨*proof*⟩

lemma (**in** *Order*) *Un-fTo-Chain-mem-fTo*: $Chain\ (fTo\ D)\ CC \Longrightarrow$
$(\bigcup\ CC) \in carrier\ (fTo\ D)$
⟨*proof*⟩

lemma (**in** *Order*) *Un-upper-bound*: $Chain\ (fTo\ D)\ C \Longrightarrow$
$\bigcup C \in upper\text{-}bounds\ (fTo\ D)\ C$
⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-inc-C*: $C \in carrier\ (fTo\ D) \Longrightarrow$
$C \in carrier\ (Iod\ (fTo\ D)\ \{S \in carrier\ fTo\ D.\ C \subseteq S\})$
⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-Un-Chain-mem1*: $[\![C \in carrier\ (fTo\ D);$
$Chain\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})\ Ca;\ Ca \neq \{\}]\!] \Longrightarrow$
$\bigcup Ca \in upper\text{-}bounds\ (Iod\ (fTo\ D)\ \{S \in carrier\ fTo\ D.\ C \subseteq S\})\ Ca$

⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-min1*: $[\![C \in carrier\ (fTo\ D);$
$Chain\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})\ Ca;\ Ca \neq \{\}]\!] \Longrightarrow$
$minimum\text{-}elem\ (Iod\ (fTo\ D)\ \{S \in carrier\ fTo\ D.\ C \subseteq S\})$
$(upper\text{-}bounds\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})\ Ca)\ (\bigcup Ca)$
⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-Un-Chain-mem2*: $[\![C \in carrier\ (fTo\ D);$
$Chain\ (Iod\ (fTo\ D)\ \{S \in carrier\ fTo\ D.\ C \subseteq S\})\ Ca;\ Ca = \{\}]\!] \Longrightarrow$
$C \in upper\text{-}bounds\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})\ Ca$

⟨proof⟩

lemma (**in** *Order*) *fTo-conditional-min2*: $[\![C \in carrier\ (fTo\ D);$
$Chain\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})\ Ca;\ Ca = \{\}]\!] \Longrightarrow$
$minimum\text{-}elem\ (Iod\ (fTo\ D)\ \{S \in carrier\ fTo\ D.\ C \subseteq S\})$
$(upper\text{-}bounds\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})\ Ca)\ C$
⟨proof⟩

lemma (**in** *Order*) *fTo-S-inductive*:*S-inductive-set* (*fTo D*)
⟨proof⟩

lemma (**in** *Order*) *conditional-min-upper-bound*: $[\![C \in carrier\ (fTo\ D);$
$Chain\ (Iod\ (fTo\ D)\ \{S \in carrier\ fTo\ D.\ C \subseteq S\})\ Ca]\!] \Longrightarrow$
$\exists X.\ minimum\text{-}elem\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})$
$(upper\text{-}bounds\ (Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\})\ Ca)\ X$
⟨proof⟩

lemma (**in** *Order*) *Hausdorff-acTr*:$C \in carrier\ (fTo\ D) \Longrightarrow$
$S\text{-}inductive\text{-}set\ (Iod\ (fTo\ D)\ \{S.\ S \in (carrier\ (fTo\ D)) \wedge\ C \subseteq S\})$
⟨proof⟩

lemma *satisfy-cond-mem-set*:$[\![\ x \in A;\ P\ x\ ]\!] \Longrightarrow x \in \{y \in A.\ P\ y\}$
⟨proof⟩

lemma (**in** *Order*) *maximal-conditional-maximal*: $[\![C \in carrier\ (fTo\ D);$
$maximal_{Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\}}\ m]\!] \Longrightarrow maximal_{(fTo\ D)}\ m$
⟨proof⟩

lemma (**in** *Order*) *Hausdorff-ac*:$C \in carrier\ (fTo\ D) \Longrightarrow$
$\exists M{\in}carrier\ (fTo\ D).\ C \subseteq M \wedge maximal_{(fTo\ D)}\ M$
⟨proof⟩

lemma (**in** *Order*) *Zorn-lemmaTr*:$[\![Chain\ D\ C;\ M \in carrier\ fTo\ D;\ C \subseteq M;$
$maximal_{fTo\ D}\ M;\ b \in carrier\ D;\ \forall s{\in}M.\ s \preceq b\ ]\!] \Longrightarrow$
$maximal\ b \wedge b \in upper\text{-}bounds\ D\ C$
⟨proof⟩

lemma (**in** *Order*) *g-Zorn-lemma1*:$[\![inductive\text{-}set\ D;\ Chain\ D\ C]\!] \Longrightarrow \exists m.\ maximal\ m \wedge m \in upper\text{-}bounds\ D\ C$
⟨proof⟩

lemma (**in** *Order*) *g-Zorn-lemma2*:$[\![inductive\text{-}set\ D;\ a \in carrier\ D]\!] \Longrightarrow$
$\exists m{\in}carrier\ D.\ maximal\ m \wedge a \preceq m$
⟨proof⟩

lemma (**in** *Order*) *g-Zorn-lemma3*:$inductive\text{-}set\ D \Longrightarrow \exists m{\in}carrier\ D.\ maximal\ m$

$\langle proof \rangle$

Chapter 3

Group Theory. Focused on Jordan Hoelder theorem

3.1 Definition of a Group

record *'a Group* = *'a carrier* +
  top :: [*'a*, *'a*] ⇒ *'a* (**infixl** $\cdot_1$ *70*)
  iop :: *'a* ⇒ *'a* (ϱ_1 - [*81*] *80*)
  one :: *'a* ($\mathbf{1}_1$)

locale *Group* =
 fixes *G* (**structure**)
 assumes *top-closed*: $top\ G \in carrier\ G \to carrier\ G \to carrier\ G$
 and *tassoc* : $[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G]\!] \Longrightarrow (a \cdot b) \cdot c = a \cdot (b \cdot c)$
 and *iop-closed*: $iop\ G \in carrier\ G \to carrier\ G$
 and *l-i* : $a \in carrier\ G \Longrightarrow (\varrho\ a) \cdot a = \mathbf{1}$
 and *unit-closed*: $\mathbf{1} \in carrier\ G$
 and *l-unit*: $a \in carrier\ G \Longrightarrow \mathbf{1} \cdot a = a$

lemma (**in** *Group*) *mult-closed*: $[\![a \in carrier\ G;\ b \in carrier\ G]\!] \Longrightarrow a \cdot b \in carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *i-closed*: $a \in carrier\ G \Longrightarrow (\varrho\ a) \in carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *r-mult-eqn*: $[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;\ a = b]\!] \Longrightarrow a \cdot c = b \cdot c$
⟨*proof*⟩

lemma (**in** *Group*) *l-mult-eqn*: $[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;\ a = b]\!] \Longrightarrow c \cdot a = c \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *r-i*:$a \in carrier\ G \Longrightarrow$
$a \cdot (\varrho\ a) = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *r-unit*:$a \in carrier\ G \Longrightarrow a \cdot \mathbf{1} = a$
⟨*proof*⟩

lemma (**in** *Group*) *l-i-unique*:$[\![a \in carrier\ G;\ b \in carrier\ G;$
$b \cdot a = \mathbf{1}\ ]\!] \Longrightarrow (\varrho\ a) = b$
⟨*proof*⟩

lemma (**in** *Group*) *l-i-i*:$a \in carrier\ G \Longrightarrow (\varrho\ (\varrho\ a)) \cdot (\varrho\ a) = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *l-div-eqn*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ y \in carrier\ G;$
$a \cdot x = a \cdot y\ ]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Group*) *r-div-eqn*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ y \in carrier\ G;$
$x \cdot a = y \cdot a\ ]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Group*) *l-mult-eqn1*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ \ y \in carrier\ G;$
$(\varrho\ a) \cdot x = (\varrho\ a) \cdot y]\!] \ \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr41*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G]\!] \Longrightarrow a \cdot b \cdot c \cdot d = a \cdot b \cdot (c \cdot d)$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr42*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G]\!] \Longrightarrow a \cdot b \cdot c \cdot d = a \cdot (b \cdot c) \cdot d$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr44*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G\ ]\!] \Longrightarrow (\varrho\ a) \cdot b \cdot ((\varrho\ c) \cdot d) =$
$(\varrho\ \ a) \cdot ((b \cdot (\varrho\ c)) \cdot d)$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr45*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G]\!] \Longrightarrow a \cdot b \cdot c \cdot d = a \cdot (b \cdot (c \cdot d))$
⟨*proof*⟩

lemma (**in** *Group*) *one-unique*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ \ x \cdot a = x]\!] \Longrightarrow$
$a = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *i-one*:$\varrho\ \mathbf{1} = \mathbf{1}$

⟨*proof*⟩

lemma (**in** *Group*) *eqn-inv1*:$[\![a \in carrier\ G; x \in carrier\ G; a = (\varrho\ x)\ ]\!] \Longrightarrow$
$x = (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *eqn-inv2*:$[\![a \in carrier\ G; x \in carrier\ G;\ \ x \cdot a = x \cdot (\varrho\ x)]\!]$
$\Longrightarrow$
$x = (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *r-one-unique*:$[\![a \in carrier\ G; x \in carrier\ G; a \cdot x = a]\!] \Longrightarrow$
$x = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *r-i-unique*:$[\![a \in carrier\ G; x \in carrier\ G; a \cdot x = \mathbf{1}]\!] \Longrightarrow$
$x = (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *iop-i-i*:$a \in carrier\ G \Longrightarrow \varrho\ (\varrho\ a) = a$
⟨*proof*⟩

lemma (**in** *Group*) *i-ab*:$[\![a \in carrier\ G; b \in carrier\ G]\!] \Longrightarrow$
$\varrho\ (a \cdot b) = (\varrho\ b) \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *sol-eq-l*:$[\![a \in carrier\ G; b \in carrier\ G; x \in carrier\ G;$
$a \cdot x = b]\!] \Longrightarrow x = (\varrho\ a) \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *sol-eq-r*:$[\![a \in carrier\ G; b \in carrier\ G; x \in carrier\ G;$
$x \cdot a = b\ ]\!] \Longrightarrow\ x = b \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *r-div-eq*:$[\![a \in carrier\ G; b \in carrier\ G; a \cdot (\varrho\ b) = \mathbf{1}]\!] \Longrightarrow$
$a = b$
⟨*proof*⟩

lemma (**in** *Group*) *l-div-eq*:$[\![a \in carrier\ G; b \in carrier\ G; (\varrho\ a) \cdot b = \mathbf{1}]\!] \Longrightarrow$
$a = b$
⟨*proof*⟩

lemma (**in** *Group*) *i-m-closed*:$[\![a \in carrier\ G\ ; b \in carrier\ G]\!] \Longrightarrow$
$(\varrho\ a) \cdot b \in carrier\ G$
⟨*proof*⟩

3.2 Subgroups

definition

sg ::[- , $'a$ set] $\Rightarrow$ $bool$ (- $\gg$ - [60, 61]60) **where**
$G \gg H \longleftrightarrow H \neq \{\} \wedge H \subseteq carrier\ G \wedge (\forall\, a \in H.\ \forall\, b \in H.\ a \cdot_G (\varrho_G\ b) \in H)$

definition
Gp :: - $\Rightarrow$ $'a$ set $\Rightarrow$ - (($\natural$1-) 70) **where**
$\natural_G H \equiv G\ (\!|\ carrier := H,\ top := top\ G,\ iop := iop\ G,\ one := one\ G|\!)$

definition
rcs :: [- , $'a$ set, $'a$] $\Rightarrow$ $'a$ set (**infix** $\bullet$1 70) **where**
$H \bullet_G a = \{b.\ \exists\ h \in H.\ h \cdot_G a = b\}$

definition
lcs :: [- , $'a$, $'a$ set] $\Rightarrow$ $'a$ set (**infix** $\diamondsuit$1 70) **where**
$a \diamondsuit_G H = \{b.\ \exists\ h \in H.\ a \cdot_G h = b\}$

definition
nsg :: - $\Rightarrow$ $'a$ set $\Rightarrow$ $bool$ (- $\rhd$ - [60,61]60) **where**
$G \rhd H \longleftrightarrow G \gg H \wedge (\forall\, x \in carrier\ G.\ H \bullet_G x = x \diamondsuit_G H)$

definition
$set\text{-}rcs$ :: [- , $'a$ set] $\Rightarrow$ $'a$ set set **where**
$set\text{-}rcs\ G\ H = \{C.\ \exists\, a \in carrier\ G.\ C = H \bullet_G a\}$

definition
$c\text{-}iop$ :: [- , $'a$ set] $\Rightarrow$ $'a$ set $\Rightarrow$ $'a$ set **where**
$c\text{-}iop\ G\ H = (\lambda X{\in}set\text{-}rcs\ G\ H.\ \{z.\ \exists\ x \in X\ .\ \exists\, h \in H.\ h \cdot_G (\varrho_G\ x) = z\})$

definition
$c\text{-}top$ :: [-, $'a$ set] $\Rightarrow$ ([$'a$ set, $'a$ set] $\Rightarrow$ $'a$ set) **where**
$c\text{-}top\ G\ H = (\lambda X{\in}set\text{-}rcs\ G\ H.\ \lambda Y{\in}set\text{-}rcs\ G\ H.$
$\{z.\ \exists\, x \in X.\ \exists\ y \in Y.\ x \cdot_G y = z\})$

lemma (**in** $Group$) $sg\text{-}subset$:$G \gg H \Longrightarrow H \subseteq carrier\ G$
⟨$proof$⟩

lemma (**in** $Group$) $one\text{-}Gp\text{-}one$:$G \gg H \Longrightarrow \mathbf{1}_{(Gp\ G\ H)} = \mathbf{1}$
⟨$proof$⟩

lemma (**in** $Group$) $carrier\text{-}Gp$:$G \gg H \Longrightarrow (carrier\ (\natural H)) = H$
⟨$proof$⟩

lemma (**in** $Group$) $sg\text{-}subset\text{-}elem$:$[\![G \gg H;\ h \in H\ ]\!] \Longrightarrow h \in carrier\ G$
⟨$proof$⟩

lemma (**in** *Group*) *sg-mult-closedr*:$\llbracket G \gg H;\ x \in carrier\ G;\ h \in H\rrbracket \Longrightarrow$
$x \cdot h \in carrier\ G$
$\langle proof \rangle$

lemma (**in** *Group*) *sg-mult-closedl*:$\llbracket G \gg H;\ x \in carrier\ G;\ h \in H\rrbracket \Longrightarrow$
$h \cdot x \in carrier\ G$
$\langle proof \rangle$

lemma (**in** *Group*) *sg-condTr1*:$\llbracket H \subseteq carrier\ G;\ H \neq \{\};$
$\forall a. \forall b.\ a \in H \wedge b \in H \longrightarrow\ a \cdot (\varrho\ b) \in H\rrbracket \Longrightarrow \mathbf{1} \in H$
$\langle proof \rangle$

lemma (**in** *Group*) *sg-unit-closed*:$G \gg H \Longrightarrow \mathbf{1} \in H$
$\langle proof \rangle$

lemma (**in** *Group*) *sg-i-closed*:$\llbracket G \gg H;\ x \in H\rrbracket \Longrightarrow (\varrho\ x) \in H$
$\langle proof \rangle$

lemma (**in** *Group*) *sg-mult-closed*:$\llbracket G \gg H;\ x \in H;\ y \in H\rrbracket \Longrightarrow$
$x \cdot y \in H$
$\langle proof \rangle$

lemma (**in** *Group*) *nsg-sg*: $G \triangleright H \Longrightarrow G \gg H$
$\langle proof \rangle$

lemma (**in** *Group*) *nsg-subset*:$G \triangleright N \Longrightarrow N \subseteq carrier\ G$
$\langle proof \rangle$

lemma (**in** *Group*) *nsg-lr-cst-eq*:$\llbracket G \triangleright N;\ a \in carrier\ G\rrbracket \Longrightarrow$
$a \diamondsuit N = N \bullet a$
$\langle proof \rangle$

lemma (**in** *Group*) *sg-i-m-closed*:$\llbracket G \gg H;\ a \in H\ ;\ b \in H\rrbracket \Longrightarrow (\varrho\ a) \cdot b \in H$
$\langle proof \rangle$

lemma (**in** *Group*) *sg-m-i-closed*:$\llbracket G \gg H;\ a \in H\ ;\ b \in H\ \rrbracket \Longrightarrow a \cdot (\varrho\ b) \in H$
$\langle proof \rangle$

definition
sg-gen :: [- , *'a set*] $\Rightarrow$ *'a set* **where**
$sg\text{-}gen\ G\ A = \bigcap\{H.\ G \gg H \wedge A \subseteq H\}$

lemma (**in** *Group*) *smallest-sg-gen*:$\llbracket A \subseteq carrier\ G;\ G \gg H;\ A \subseteq H\rrbracket \Longrightarrow$
$sg\text{-}gen\ G\ A \subseteq H$
$\langle proof \rangle$

lemma (**in** *Group*) *special-sg-G*: $G \gg (carrier\ G)$
$\langle proof \rangle$

lemma (**in** *Group*) *special-sg-self*: $G = \natural(carrier\ G)$
⟨*proof*⟩

lemma (**in** *Group*) *special-sg-e*: $G \gg \{\mathbf{1}\}$
⟨*proof*⟩

lemma (**in** *Group*) *inter-sgs*:$[\![G \gg H;\ G \gg K]\!] \Longrightarrow G \gg (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *subg-generated*:$A \subseteq carrier\ G \Longrightarrow G \gg (sg\text{-}gen\ G\ A)$
⟨*proof*⟩

definition
  Qg :: [- , 'a set] ⇒
    (|*carrier*:: 'a set set, *top*:: ['a set, 'a set] ⇒ 'a set,
     iop:: 'a set ⇒ 'a set, *one*:: 'a set |) **where**
  Qg G H = (|*carrier* = *set-rcs G H*, *top* = *c-top G H*, *iop* = *c-iop G H*, *one* = *H* |)

definition
  Pj :: [- , 'a set] ⇒ ('a => 'a set) **where**
  Pj G H = $(\lambda x \in carrier\ G.\ H \bullet_G x)$

no-notation *inverse-divide* (**infixl** '/ *70*)

abbreviation
  QGRP :: ([('a, 'more) *Group-scheme*, 'a set] => ('a set) *Group*)
   (**infixl** '/ *70*) **where**
  G / H == *Qg G H*

definition
  gHom ::[('a, 'more) *Group-scheme*, ('b, 'more1) *Group-scheme*] ⇒
    ('a ⇒ 'b) *set* **where**
  gHom G F = $\{f.\ (f \in extensional\ (carrier\ G) \wedge f \in carrier\ G \rightarrow carrier\ F) \wedge (\forall x \in carrier\ G.\ \forall y \in carrier\ G.\ f\ (x \cdot_G y) = (f\ x) \cdot_F (f\ y))\}$

definition
  gkernel ::[('a, 'more) *Group-scheme* , ('b, 'more1) *Group-scheme*, 'a ⇒ 'b]
    ⇒ 'a set **where**
  gkernel G F f = $\{x.\ (x \in carrier\ G) \wedge (f\ x = \mathbf{1}_F)\}$

definition
  iim :: [('a, 'more) *Group-scheme*, ('b, 'more1) *Group-scheme*, 'a ⇒ 'b,
    'b set] ⇒ 'a set **where**
  iim G F f K = $\{x.\ (x \in carrier\ G) \wedge (f\ x \in K)\}$

abbreviation
GKER :: [('*a*, '*more*) *Group-scheme*, ('*b*, '*more1*) *Group-scheme*, '*a* ⇒ '*b*] ⇒ '*a* *set*
((*3gker*$_{-,-}$ -) [*88*,*88*,*89*]*88*) **where**
$gker_{G,F}\ f$ == *gkernel G F f*

definition
gsurjec :: [('*a*, '*more*) *Group-scheme*, ('*b*, '*more1*) *Group-scheme*,
'*a* ⇒ '*b*] ⇒ *bool* ((*3gsurj*$_{-,-}$ -) [*88*,*88*,*89*]*88*) **where**
$gsurj_{F,G}\ f \longleftrightarrow f \in gHom\ F\ G \wedge surj\text{-}to\ f\ (carrier\ F)\ (carrier\ G)$

definition
ginjec :: [('*a*, '*more*) *Group-scheme*, ('*b*, '*more1*) *Group-scheme*,
'*a* ⇒ '*b*] ⇒ *bool* ((*3ginj*$_{-,-}$ -) [*88*,*88*,*89*]*88*) **where**
$ginj_{F,G}\ f \longleftrightarrow f \in gHom\ F\ G \wedge inj\text{-}on\ f\ (carrier\ F)$

definition
gbijec :: [('*a*, '*m*) *Group-scheme*, ('*b*, '*m1*) *Group-scheme*, '*a* ⇒ '*b*]
⇒ *bool* ((*3gbij*$_{-,-}$ -) [*88*,*88*,*89*]*88*) **where**
$gbij_{F,G}\ f \longleftrightarrow gsurj_{F,G}\ f \wedge ginj_{F,G}\ f$

definition
Ug :: - ⇒ ('*a*, '*more*) *Group-scheme* **where**
$Ug\ G = \natural_G\ \{\mathbf{1}_G\}$

definition
Ugp :: - ⇒ *bool* **where**
$Ugp\ G == Group\ G \wedge carrier\ G = \{\mathbf{1}_G\}$

definition
isomorphic :: [('*a*, '*m*) *Group-scheme*, ('*b*, '*m1*) *Group-scheme*]
⇒ *bool* (**infix** ≅ *100*) **where**
$F \cong G \longleftrightarrow (\exists f.\ gbij_{F,G}\ f)$

definition
constghom :: [('*a*, '*m*) *Group-scheme*, ('*b*, '*m1*) *Group-scheme*]
⇒ ('*a* ⇒ '*b*) (('*1*'$_{-,-}$) [*88*,*89*]*88*) **where**
$1_{F,G} = (\lambda x{\in}carrier\ F.\ \mathbf{1}_G)$

definition
cmpghom :: [('*a*, '*m*) *Group-scheme*, '*b* ⇒ '*c*, '*a* ⇒ '*b*] ⇒ '*a* ⇒ '*c* **where**
cmpghom F g f = *compose* (*carrier F*) *g f*

abbreviation
GCOMP :: ['*b* ⇒ '*c*, ('*a*, '*m*) *Group-scheme*, '*a* ⇒ '*b*] ⇒ '*a* ⇒ '*c*
((*3*- ∘$_{-}$ -) [*88*, *88*, *89*]*88*) **where**
$g \circ_F f$ == *cmpghom F g f*

lemma *Group-Ugp*:$Ugp\ G \Longrightarrow Group\ G$
⟨*proof*⟩

lemma (**in** *Group*) *r-mult-in-sg*:$[\![G \ggg H;\ a \in carrier\ G;\ x \in carrier\ G;\ x \cdot a \in H]\!]$
$\Longrightarrow \exists h \in H.\ h \cdot (\varrho\ a) = x$
⟨*proof*⟩

lemma (**in** *Group*) *r-unit-sg*:$[\![G \ggg H;\ h \in H]\!] \Longrightarrow h \cdot \mathbf{1} = h$
⟨*proof*⟩

lemma (**in** *Group*) *sg-l-unit*:$[\![G \ggg H;\ h \in H]\!] \Longrightarrow \mathbf{1} \cdot h = h$
⟨*proof*⟩

lemma (**in** *Group*) *sg-l-i*:$[\![G \ggg H;\ x \in H\ ]\!] \Longrightarrow (\varrho\ x) \cdot x = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *sg-tassoc*:$[\![G \ggg H;\ x \in H;\ y \in H;\ z \in H]\!] \Longrightarrow$
$x \cdot y \cdot z = x \cdot (y \cdot z)$
⟨*proof*⟩

lemma (**in** *Group*) *sg-condition*:$[\![H \subseteq carrier\ G;\ H \neq \{\};$
$\forall a.\ \forall b.\ a \in H \wedge b \in H \longrightarrow\ a \cdot (\varrho\ b) \in H]\!] \Longrightarrow G \ggg H$
⟨*proof*⟩

definition
Gimage :: [(*'a*, *'m*) *Group-scheme*, (*'b*, *'m1*) *Group-scheme*, *'a* ⇒ *'b*] ⇒
(*'b*, *'m1*) *Group-scheme* **where**
Gimage F G f = *Gp G* (*f* ‘(*carrier F*))

abbreviation
GIMAGE :: [(*'a*, *'m*) *Group-scheme*, (*'b*, *'m1*) *Group-scheme*,
'a ⇒ *'b*] ⇒ (*'b*, *'m1*) *Group-scheme* ((*3Img*$_{-,-}$ -) [*88*,*88*,*89*]*88*) **where**
$Img_{F,G}\ f$ == *Gimage F G f*

lemma (**in** *Group*) *Group-Gp*:$G \ggg H \Longrightarrow Group\ (\natural\ H)$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-carrier*:$G \ggg H \Longrightarrow carrier\ (Gp\ G\ H) = H$
⟨*proof*⟩

lemma (**in** *Group*) *sg-sg*:$[\![G \ggg K;\ G \ggg H;\ H \subseteq K]\!] \Longrightarrow Gp\ G\ K \ggg H$
⟨*proof*⟩

lemma (**in** *Group*) *sg-subset-of-subG*:$[\![G \ggg K;\ Gp\ G\ K \ggg H]\!] \Longrightarrow H \subseteq K$
⟨*proof*⟩

lemma *const-ghom*:$[\![Group\ F;\ Group\ G]\!] \Longrightarrow 1_{F,G} \in gHom\ F\ G$
⟨*proof*⟩

lemma (**in** *Group*) *const-gbij*:$gbij_{(\natural\ \{\mathbf{1}\}),(\natural\ \{\mathbf{1}\})}\ (1_{(\natural\{\mathbf{1}\}),(\natural\{\mathbf{1}\})})$
⟨*proof*⟩

lemma (**in** *Group*) *unit-Groups-isom*: $(\natural\ \{\mathbf{1}\}) \cong (\natural\ \{\mathbf{1}\})$
⟨*proof*⟩

lemma *Ugp-const-gHom*:$[\![Ugp\ G;\ Ugp\ E]\!] \Longrightarrow (\lambda x{\in}carrier\ G.\ \mathbf{1}_E) \in gHom\ G\ E$
⟨*proof*⟩

lemma *Ugp-const-gbij*:$[\![Ugp\ G;\ Ugp\ E]\!] \Longrightarrow gbij_{G,E}\ (\lambda x{\in}carrier\ G.\ \mathbf{1}_E)$
⟨*proof*⟩

lemma *Ugps-isomorphic*:$[\![Ugp\ G;\ Ugp\ E]\!] \Longrightarrow G \cong E$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-mult-induced*:$[\![G \gg L;\ a \in L;\ b \in L]\!] \Longrightarrow$
$a \cdot_{(Gp\ G\ L)} b = a \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *sg-i-induced*:$[\![G \gg L;\ a \in L]\!] \Longrightarrow \varrho_{(Gp\ G\ L)}\ a = \varrho\ a$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-mult-induced1*:$[\![G \gg H\ ;\ G \gg K;\ a \in H \cap K;\ b \in H \cap K]\!]$

$\Longrightarrow a \cdot_{\natural(H\ \cap\ K)} b = a \cdot_{(\natural H)} b$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-mult-induced2*:$[\![G \gg H\ ;\ G \gg K;\ a \in H \cap K;\ b \in H \cap K]\!]$

$\Longrightarrow a \cdot_{\natural(H\ \cap\ K)} b = a \cdot_{(\natural K)} b$
⟨*proof*⟩

lemma (**in** *Group*) *sg-i-induced1*:$[\![G \gg H\ ;\ G \gg K;\ a \in H \cap K]\!]$
$\Longrightarrow \varrho_{\natural(H\ \cap\ K)}\ a = \varrho_{(\natural H)}\ a$
⟨*proof*⟩

lemma (**in** *Group*) *sg-i-induced2*:$[\![G \gg H\ ;\ G \gg K;\ a \in H \cap K]\!]$
$\Longrightarrow \varrho_{\natural(H\ \cap\ K)}\ a = \varrho_{\natural K}\ a$
⟨*proof*⟩

lemma (**in** *Group*) *subg-sg-sg*:$[\![G \gg K;\ (Gp\ G\ K) \gg H\ ]\!] \Longrightarrow G \gg H$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-inherited*:$[\![G \gg K;\ G \gg L;\ K \subseteq L]\!] \Longrightarrow$
$Gp\ (Gp\ G\ L)\ K = Gp\ G\ K$
⟨*proof*⟩

3.3 Cosets

lemma (**in** *Group*) *mem-lcs*: $[\![G \gg H;\ a \in carrier\ G;\ x \in a \diamondsuit H]\!] \Longrightarrow$
$x \in carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-subset*: $[\![G \gg H;\ a \in carrier\ G]\!] \Longrightarrow a \diamondsuit H \subseteq carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *a-in-lcs*: $[\![G \gg H;\ a \in carrier\ G]\!] \Longrightarrow a \in a \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *eq-lcs1*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$x \in a \diamondsuit H;\ a \diamondsuit H = b \diamondsuit H]\!] \Longrightarrow x \in b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *eq-lcs2*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$a \diamondsuit H = b \diamondsuit H]\!] \Longrightarrow a \in b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-mem-ldiv*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G]\!] \Longrightarrow$
$(a \in b \diamondsuit H) = ((\varrho\ b) \cdot a \in H)$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr5*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$(\varrho\ a) \cdot b \in H;\ x \in a \diamondsuit H]\!] \Longrightarrow ((\varrho\ b) \cdot x) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr6*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$(\varrho\ a) \cdot b \in H;\ x \in a \diamondsuit H]\!] \Longrightarrow x \in b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-Unit1*: $G \gg H \Longrightarrow \mathbf{1} \diamondsuit H = H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-Unit2*: $[\![G \gg H;\ h \in H]\!] \Longrightarrow h \diamondsuit H = H$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr7*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;\ (\varrho\ a) \cdot b \in H]\!]$
$\Longrightarrow a \diamondsuit H \subseteq b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr8*: $[\![G \gg H;\ a \in carrier\ G;\ h \in H]\!] \Longrightarrow a \cdot h \in a \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-tool1*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$(\varrho\ a) \cdot b \in H]\!] \Longrightarrow (\varrho\ b) \cdot a \in H$

⟨*proof*⟩

theorem (**in** *Group*) *lcs-eq*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G]\!] \Longrightarrow$
$((\varrho\ a) \cdot b \in H) = (a \diamondsuit H = b \diamondsuit H)$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-subset*:$[\![G \gg H;\ a \in \mathit{carrier}\ G]\!] \Longrightarrow H \bullet a \subseteq \mathit{carrier}\ G$
⟨*proof*⟩

lemma (**in** *Group*) *mem-rcs*:$[\![G \gg H;\ x \in H \bullet a]\!] \Longrightarrow \exists h \in H.\ h \cdot a = x$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-subset-elem*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ x \in H \bullet a]\!] \Longrightarrow$
$x \in \mathit{carrier}\ G$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-in-set-rcs*:$[\![G \gg H;\ a \in \mathit{carrier}\ G]\!] \Longrightarrow$
$H \bullet a \in \mathit{set\text{-}rcs}\ G\ H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr0*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G]\!] \Longrightarrow$
$H \bullet (a \cdot b) \in \mathit{set\text{-}rcs}\ G\ H$
⟨*proof*⟩

lemma (**in** *Group*) *a-in-rcs*:$[\![G \gg H;\ a \in \mathit{carrier}\ G]\!] \Longrightarrow a \in H \bullet a$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-nonempty*:$[\![G \gg H;\ X \in \mathit{set\text{-}rcs}\ G\ H]\!] \Longrightarrow X \neq \{\}$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-tool0*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G;$
$a \cdot (\varrho\ b) \in H]\!] \Longrightarrow b \cdot (\varrho\ a) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr1*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G;$
$x \in H \bullet a;\ H \bullet a = H \bullet b]\!] \Longrightarrow x \in H \bullet b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eqTr*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G;$
$H \bullet a = H \bullet b]\!] \Longrightarrow a \in H \bullet b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eqTr1*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G]\!] \Longrightarrow$
$(a \in H \bullet b) = (a \cdot (\varrho\ b) \in H)$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr2*:$[\![G \gg H;\ a \in \mathit{carrier}\ G;\ b \in H \bullet (\varrho\ a)]\!] \Longrightarrow$
$b \cdot a \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr5*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$b \cdot (\varrho\ a) \in H;\ x \in H \bullet a]\!] \Longrightarrow x \cdot (\varrho\ b) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr6*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$b \cdot (\varrho\ a) \in H\ ;\ x \in H \bullet a]\!] \Longrightarrow x \in H \bullet b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-Unit1*: $G \gg H \Longrightarrow\ H \bullet \mathbf{1} = H$
⟨*proof*⟩

lemma (**in** *Group*) *unit-rcs-in-set-rcs*: $G \gg H \Longrightarrow H \in set\text{-}rcs\ G\ H$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-Unit2*: $[\![G \gg H;\ h \in H]\!] \Longrightarrow H \bullet h = H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr7*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;\ b \cdot (\varrho\ a) \in H]\!]$
$\Longrightarrow H \bullet a\ \subseteq\ H \bullet b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-tool1*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G;$
$b \cdot (\varrho\ a) \in H]\!] \Longrightarrow\ a \cdot (\varrho\ b) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-tool2*: $[\![G \gg H;\ a \in carrier\ G;\ \ x \in H \bullet a]\!] \Longrightarrow$
$\exists\ h \in H.\ h \cdot a = x$
⟨*proof*⟩

theorem (**in** *Group*) *rcs-eq*: $[\![G \gg H;\ a \in carrier\ G;\ b \in carrier\ G]\!] \Longrightarrow$
$(b \cdot (\varrho\ a) \in H) = (H \bullet a = H \bullet b)$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eq1*: $[\![G \gg H;\ a \in carrier\ G;\ x \in H \bullet a]\!] \Longrightarrow$
$H \bullet a = H \bullet x$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eq2*: $[\![G \gg H;\ a \in carrier\ G;\ \ b \in carrier\ G;$
$(H \bullet a) \cap (H \bullet b) \neq \{\}]\!] \Longrightarrow (H \bullet a) = (H \bullet b)$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-meet*: $[\![G \gg H;\ X \in set\text{-}rcs\ G\ H;\ Y \in set\text{-}rcs\ G\ H;$
$X \cap Y \neq \{\}]\!] \Longrightarrow X = Y$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr8*: $[\![G \gg H;\ a \in carrier\ G;\ h \in H;\ x \in H \bullet a]\!] \Longrightarrow$
$h \cdot x \in H \bullet a$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr9*: $[\![G \gg H;\ a \in carrier\ G;\ h \in H;\ (\varrho\ x) \in H \bullet a]\!] \Longrightarrow$
$h \cdot (\varrho\ x) \in H \bullet a$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr10*: $[\![G \gg H;\ a \in carrier\ G;\ x \in H \bullet a;\ y \in H \bullet a]\!] \Longrightarrow$
$x \cdot (\varrho\ y) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *PrSubg4-2*: $[\![G \gg H;\ a \in carrier\ G;\ x \in H \bullet (\varrho\ a)]\!] \Longrightarrow$
$x \in \{z.\ \exists v \in (H \bullet a).\ \exists h \in H.\ h \cdot (\varrho\ v) = z\}$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-fixed*: $[\![G \gg H;\ a \in carrier\ G;\ H \bullet a = H]\!] \Longrightarrow a \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-fixed1*: $[\![G \gg H;\ a \in carrier\ G;\ h \in H]\!] \Longrightarrow$
$H \bullet a = (H \bullet (h \cdot a))$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-fixed2*: $G \gg H \Longrightarrow \forall h \in H.\ H \bullet h = H$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-rcs*: $[\![G \gg H;\ G \gg K;\ H \subseteq K;\ x \in K]\!] \Longrightarrow$
$H \bullet_{(Gp\ G\ K)}\ x = (H \bullet x)$
⟨*proof*⟩

lemma (**in** *Group*) *subg-lcs*: $[\![G \gg H;\ G \gg K;\ H \subseteq K;\ x \in K]\!] \Longrightarrow$
$x \diamondsuit_{(Gp\ G\ K)}\ H = x \diamondsuit H$
⟨*proof*⟩

3.4 Normal subgroups and Quotient groups

lemma (**in** *Group*) *nsg1*: $[\![G \gg H;\ b \in carrier\ G;\ h \in H;$
$\forall a \in carrier\ G.\ \forall h \in H.\ (a \cdot h) \cdot (\varrho\ a) \in H]\!] \Longrightarrow b \cdot h \cdot (\varrho\ b) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *nsg2*: $[\![G \gg H;\ b \in carrier\ G;\ h \in H;$
$\forall a \in carrier\ G.\ \forall h \in H.\ (a \cdot h) \cdot (\varrho\ a) \in H]\!] \Longrightarrow (\varrho\ b) \cdot h \cdot b \in H$
⟨*proof*⟩

lemma (**in** *Group*) *nsg-subset-elem*: $[\![G \rhd H;\ h \in H]\!] \Longrightarrow h \in carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *nsg-l-rcs-eq*: $[\![G \rhd N;\ a \in carrier\ G]\!] \Longrightarrow a \diamondsuit N = N \bullet a$
⟨*proof*⟩

lemma (**in** *Group*) *sg-nsg1*: $[\![G \gg H;\ \forall a \in carrier\ G.\ \forall h \in H.\ (a \cdot h) \cdot (\varrho\ a) \in H;$

$b \in carrier\ G\ \rrbracket \Longrightarrow H \bullet b = \ b \diamondsuit H$

⟨*proof*⟩

lemma (**in** *Group*) *cond-nsg*: $\llbracket G \gg H;\ \forall a{\in}carrier\ G.\ \forall h{\in}H.\ a \cdot h \cdot (\varrho\ a) \in H\ \rrbracket \Longrightarrow G \rhd H$

⟨*proof*⟩

lemma (**in** *Group*) *special-nsg-e*: $G \gg H \Longrightarrow Gp\ G\ H \rhd \{\mathbf{1}\}$

⟨*proof*⟩

lemma (**in** *Group*) *special-nsg-G*: $G \rhd (carrier\ G)$

⟨*proof*⟩

lemma (**in** *Group*) *special-nsg-G1*: $G \gg H \Longrightarrow Gp\ G\ H \rhd H$

⟨*proof*⟩

lemma (**in** *Group*) *nsgTr0*: $\llbracket G \rhd N;\ a \in carrier\ G;\ b \in carrier\ G;\ b \in N \bullet a\ \rrbracket \Longrightarrow\ (a \cdot (\varrho\ b) \in N) \wedge ((\varrho\ a) \cdot b \in N)$

⟨*proof*⟩

lemma (**in** *Group*) *nsgTr1*: $\llbracket G \rhd N;\ a \in carrier\ G;\ b \in carrier\ G;\ b \cdot (\varrho\ a) \in N\rrbracket \Longrightarrow (\varrho\ b) \cdot a \in N$

⟨*proof*⟩

lemma (**in** *Group*) *nsgTr2*: $\llbracket a \in carrier\ G;\ b \in carrier\ G;\ a1 \in carrier\ G;\ b1 \in carrier\ G\ \rrbracket \Longrightarrow (a \cdot b) \cdot (\varrho\ (a1 \cdot b1)) = a \cdot (((b \cdot (\varrho\ b1)) \cdot ((\varrho\ a1) \cdot a)) \cdot (\varrho\ a))$

⟨*proof*⟩

lemma (**in** *Group*) *nsgPr1*: $\llbracket G \rhd N;\ a \in carrier\ G;\ h \in N\rrbracket \Longrightarrow a \cdot (h \cdot (\varrho\ a)) \in N$

⟨*proof*⟩

lemma (**in** *Group*) *nsgPr1-1*: $\llbracket G \rhd N;\ a \in carrier\ G\ ;\ h \in N\rrbracket \Longrightarrow (a \cdot h) \cdot (\varrho\ a) \in N$

⟨*proof*⟩

lemma (**in** *Group*) *nsgPr2*: $\llbracket G \rhd N;\ a \in carrier\ G;\ h \in N\rrbracket \Longrightarrow (\varrho\ a) \cdot (h \cdot a) \in N$

⟨*proof*⟩

lemma (**in** *Group*) *nsgPr2-1*: $\llbracket G \rhd N;\ a \in carrier\ G;\ h \in N\rrbracket \Longrightarrow (\varrho\ a) \cdot h \cdot a \in N$

⟨*proof*⟩

lemma (**in** *Group*) *nsgTr3*: $\llbracket G \rhd N;\ a \in carrier\ G;\ b \in carrier\ G;\ a1 \in carrier\ G;\ b1 \in carrier\ G;\ \ a \cdot (\varrho\ a1) \in N;\ b \cdot (\varrho\ b1) \in N\rrbracket \Longrightarrow (a \cdot b) \cdot (\varrho\ (a1 \cdot b1)) \in N$

⟨*proof*⟩

lemma (**in** *Group*) *nsg-in-Gp*:$[\![G \triangleright N;\ G \gg H;\ N \subseteq H]\!] \Longrightarrow (Gp\ G\ H) \triangleright N$
⟨*proof*⟩

lemma (**in** *Group*) *nsgTr4*:$[\![G \triangleright N;\ a \in carrier\ G;\ \ x \in N \cdot a]\!] \Longrightarrow$
$(\varrho\ x) \in N \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-topTr1*:$[\![G \triangleright N;\ a \in carrier\ G;\ b \in carrier\ G;$
$a1 \in carrier\ G;\ b1 \in carrier\ G;\ N \cdot a = N \cdot a1;\ N \cdot b = N \cdot b1]\!] \Longrightarrow$
$N \cdot (a \cdot b) = N \cdot (a1 \cdot b1)$
⟨*proof*⟩

lemma (**in** *Group*) *c-topTr2*:$[\![G \triangleright N;\ a \in carrier\ G;\ a1 \in carrier\ G;$
$N \cdot a = N \cdot a1\ ]\!] \Longrightarrow N \cdot (\varrho\ a) = N \cdot (\varrho\ a1)$
⟨*proof*⟩

lemma (**in** *Group*) *c-iop-welldefTr1*:$[\![G \triangleright N;\ a \in carrier\ G]\!] \Longrightarrow$
$c\text{-}iop\ G\ N\ (N \cdot a) \subseteq\ N \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-iop-welldefTr2*:$[\![G \triangleright N;\ a \in carrier\ G]\!] \Longrightarrow$
$N \cdot (\varrho\ a) \subseteq\ c\text{-}iop\ G\ N\ (N \cdot a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-iop-welldef*:$[\![G \triangleright N;\ a \in carrier\ G]\!] \Longrightarrow$
$c\text{-}iop\ G\ N\ (N \cdot a) =\ N \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr1*:$[\![G \triangleright N;\ a \in carrier\ G;$
$b \in carrier\ G;\ x \in N \cdot a;\ y \in N \cdot b]\!] \Longrightarrow x \cdot y \in N \cdot (a \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr2*:$[\![G \triangleright N;\ a \in carrier\ G;\ b \in carrier\ G\ ]\!]$
$\Longrightarrow c\text{-}top\ G\ N\ (N \cdot a)\ (N \cdot b) \subseteq\ N \cdot (a \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr4*:$[\![G \triangleright N;\ a \in carrier\ G;\ b \in carrier\ G;$
$x \in N \cdot (a \cdot b)]\!] \Longrightarrow x \in c\text{-}top\ G\ N\ (N \cdot a)\ (N \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr5*:$[\![G \triangleright N;\ a \in carrier\ G;\ b \in carrier\ G]\!] \Longrightarrow$
$N \cdot (a \cdot b) \subseteq c\text{-}top\ G\ N\ (N \cdot a)\ (N \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldef*:$[\![G \triangleright N;\ a \in carrier\ G;\ b \in carrier\ G]\!] \Longrightarrow$
$N \cdot (a \cdot b) = c\text{-}top\ G\ N\ (N \cdot a)\ (N \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unitTr*:$[\![G \triangleright N;\ a \in carrier\ G]\!] \Longrightarrow$
$c\text{-}top\ G\ N\ N\ (N \cdot a) = N \cdot a$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unit*:$G \triangleright N \Longrightarrow \forall x{\in}set\text{-}rcs\ G\ N.\ c\text{-}top\ G\ N\ N\ x = x$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-iTr*:$[\![G \triangleright N;\ a \in carrier\ G]\!] \Longrightarrow$
$c\text{-}top\ G\ N\ (c\text{-}iop\ G\ N\ (N \cdot a))\ (N \cdot a) = N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-i*:$G \triangleright N \Longrightarrow$
$\forall x \in set\text{-}rcs\ G\ N.\ c\text{-}top\ G\ N\ (c\text{-}iop\ G\ N\ x)\ x = N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-tassocTr*:
$[\![G \triangleright N;\ a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G\ ]\!] \Longrightarrow$
$c\text{-}top\ G\ N\ (N \cdot a)\ (c\text{-}top\ G\ N\ (N \cdot b)\ (N \cdot c)) =$
$c\text{-}top\ G\ N\ (c\text{-}top\ G\ N\ (N \cdot a)\ (N \cdot b))\ (N \cdot c)$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-tassoc*: $G \triangleright N \Longrightarrow$
$\forall X{\in}set\text{-}rcs\ G\ N.\ \forall Y{\in}set\text{-}rcs\ G\ N.\ \forall Z{\in}set\text{-}rcs\ G\ N.\ c\text{-}top\ G\ N\ X\ (c\text{-}top\ G\ N\ Y$
$Z)$
$= c\text{-}top\ G\ N\ (c\text{-}top\ G\ N\ X\ Y)\ Z$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-top*:$G \triangleright N \Longrightarrow$
$c\text{-}top\ G\ N : set\text{-}rcs\ G\ N \to set\text{-}rcs\ G\ N \to set\text{-}rcs\ G\ N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-top-closed*:$[\![G \triangleright N;\ A \in set\text{-}rcs\ G\ N;\ B \in set\text{-}rcs\ G\ N]\!] \Longrightarrow$
$c\text{-}top\ G\ N\ A\ B \in set\text{-}rcs\ G\ N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-iop*: $G \triangleright N \Longrightarrow$
$c\text{-}iop\ G\ N : set\text{-}rcs\ G\ N \to set\text{-}rcs\ G\ N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-iop-closed*:$[\![G \triangleright N;\ A \in set\text{-}rcs\ G\ N]\!] \Longrightarrow$
$c\text{-}iop\ G\ N\ A \in set\text{-}rcs\ G\ N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unit-closed*: $G \triangleright N \Longrightarrow\ N \in set\text{-}rcs\ G\ N$
⟨*proof*⟩

theorem (**in** *Group*) *Group-Qg*:$G \triangleright N \Longrightarrow\ Group\ (Qg\ G\ N)$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-one*:$G \triangleright N \Longrightarrow$ *one* $(G \ / \ N) = N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-carrier*:*carrier* $(G \ / \ (N::'a\ set))$ = *set-rcs* $G\ N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unit-group*:$G \triangleright N \Longrightarrow$
(*set-rcs* $G\ N = \{N\}$) = (*carrier* $G = N$)
⟨*proof*⟩

lemma (**in** *Group*) *Gp-Qg*:$G \triangleright N \Longrightarrow$ *Gp*$(G \ / \ N)$ (*carrier*$(G \ / \ N)$) = $G \ / \ N$
⟨*proof*⟩

lemma (**in** *Group*) *Pj-hom0*:⟦$G \triangleright N$; $x \in$ *carrier* G; $y \in$ *carrier* G⟧
$\Longrightarrow$ *Pj* $G\ N\ (x \cdot y)$ = (*Pj* $G\ N\ x$) $\cdot_{(G \ / \ N)}$ (*Pj* $G\ N\ y$)
⟨*proof*⟩

lemma (**in** *Group*) *Pj-ghom*:$G \triangleright N \Longrightarrow$ (*Pj* $G\ N$) $\in$ *gHom* $G\ (G \ / \ N)$
⟨*proof*⟩

lemma (**in** *Group*) *Pj-mem*:⟦$G \triangleright N$; $x \in$ *carrier* G⟧ $\Longrightarrow$ (*Pj* $G\ N$) $x = N \bullet x$
⟨*proof*⟩

lemma (**in** *Group*) *Pj-gsurjec*:$G \triangleright N \Longrightarrow$ *gsurjec* $G\ (G/N)$ (*Pj* $G\ N$)
⟨*proof*⟩

lemma (**in** *Group*) *lcs-in-Gp*:⟦$G \gg H$; $G \gg K$; $K \subseteq H$; $a \in H$⟧ $\Longrightarrow$
$a \diamondsuit K = a \diamondsuit_{(Gp\ G\ H)} K$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-in-Gp*:⟦$G \gg H$; $G \gg K$; $K \subseteq H$; $a \in H$ ⟧ $\Longrightarrow$
$K \bullet a = K \bullet_{(Gp\ G\ H)} a$
⟨*proof*⟩

end

theory *Algebra3* **imports** *Algebra2* **begin**

3.5 Setproducts

definition
commutators:: - ⇒ *'a set* **where**
commutators $G = \{z.\ \exists\ a \in$ *carrier* $G.\ \exists\, b \in$ *carrier* $G.$
$((a \cdot_G b) \cdot_G (\varrho_G\ a)) \cdot_G (\varrho_G\ b) = z\}$

lemma (**in** *Group*) *contain-commutator*:⟦$G \gg H$; (*commutators* G) $\subseteq H$⟧ $\Longrightarrow G$
$\triangleright H$

⟨*proof*⟩

definition
s-top :: [- , ′*a set*, ′*a set*] ⇒ ′*a set* **where**
s-top $G\ H\ K = \{z.\ \exists x \in H.\ \exists y \in K.\ (x \cdot_G y = z)\}$

abbreviation
S-TOP :: [(′*a*, ′*m*) *Group-scheme*, ′*a set*, ′*a set*] ⇒ ′*a set*
((*3*- ⋄₁ -) [*66*,*67*]*66*) **where**
$H \diamond_G K$ == *s-top* $G\ H\ K$

lemma (**in** *Group*) *s-top-induced*:⟦$G \gg L$; $H \subseteq L$; $K \subseteq L$⟧ ⟹
$H \diamond_{Gp\ G\ L} K = H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-l-unit*:$G \gg K \Longrightarrow \{\mathbf{1}\} \diamond_G K = K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-r-unit*:$G \gg K \Longrightarrow K \diamond_G \{\mathbf{1}\} = K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-sub*:⟦$G \gg H$; $G \gg K$⟧ ⟹ $H \diamond_G K \subseteq$ *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *sg-inc-set-mult*:⟦$G \gg L$; $H \subseteq L$; $K \subseteq L$⟧ ⟹ $H \diamond_G K \subseteq L$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-sub1*:⟦$H \subseteq$ (*carrier G*); $K \subseteq$ (*carrier G*)⟧ ⟹
$H \diamond_G K \subseteq$ *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-elem*:⟦$G \gg H$; $G \gg K$; $a \in H$; $b \in K$⟧ ⟹ $a \cdot b \in H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-elem1*:⟦$H \subseteq$ *carrier G*; $K \subseteq$ *carrier G*; $a \in H$; $b \in K$⟧ ⟹
$a \cdot b \in H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *mem-s-top*:⟦$H \subseteq$ *carrier G*; $K \subseteq$ *carrier G*; $u \in H \diamond_G K$⟧ ⟹
$\exists a \in H.\ \exists b \in K.\ (a \cdot b = u)$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-mono*:⟦$H \subseteq$ *carrier G*; $K \subseteq$ *carrier G*; $H1 \subseteq H$; $K1 \subseteq K$⟧
⟹ $H1 \diamond_G K1 \subseteq H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-unit-closed*:$[\![G \gg H;\ G \gg K]\!] \Longrightarrow\ \mathbf{1} \in H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-commute*:$[\![G \gg H;\ G \gg K;\ K \diamond_G H = H \diamond_G K;$
$u \in H \diamond_G K;\ \ v \in\ H \diamond_G K]\!] \Longrightarrow\ u \cdot v \in H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-commute1*:$[\![G \gg H;\ G \gg K;\ K \diamond_G H = H \diamond_G K;$
$u \in H \diamond_G K]\!] \Longrightarrow (\varrho\ u) \in H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-commute-sg*:$[\![G \gg H;\ G \gg K;\ K \diamond_G H = H \diamond_G K]\!] \Longrightarrow$
$G \gg (H \diamond_G K)$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-assoc*:$[\![G \gg H;\ G \gg K;\ G \gg L]\!] \Longrightarrow$
$(H \diamond_G K) \diamond_G L =\ H \diamond_G (K \diamond_G L)$
⟨*proof*⟩

lemma (**in** *Group*) *s-topTr6*:$[\![G \gg H1;\ G \gg H2;\ G \gg K;\ H1 \subseteq K]\!] \Longrightarrow$
$(H1 \diamond_G H2) \cap K = H1 \diamond_G (H2 \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *s-topTr6-1*:$[\![G \gg H1;\ G \gg H2;\ G \gg K;\ H2 \subseteq K]\!] \Longrightarrow$
$(H1 \diamond_G H2) \cap K = (H1 \cap K) \diamond_G H2$
⟨*proof*⟩

lemma (**in** *Group*) *l-sub-smult*:$[\![G \gg H;\ G \gg K]\!] \Longrightarrow H \subseteq H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *r-sub-smult*:$[\![G \gg H;\ G \gg K]\!] \Longrightarrow K \subseteq H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-topTr8*:$G \gg H \Longrightarrow H = H \diamond_G H$
⟨*proof*⟩

3.6 Preliminary lemmas for Zassenhaus

lemma (**in** *Group*) *Gp-sg-subset*:$[\![G \gg H;\ Gp\ G\ H \gg K]\!] \Longrightarrow K \subseteq H$
⟨*proof*⟩

lemma (**in** *Group*) *inter-Gp-nsg*:$[\![G \triangleright N;\ G \gg H\ ]\!] \Longrightarrow (\natural H) \triangleright (H \cap N)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr0*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;$
$Gp\ G\ H \triangleright H1;\ Gp\ G\ K \triangleright K1]\!] \Longrightarrow Gp\ G\ (H \cap K) \triangleright (H \cap K1)$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-sub-s-mult*:$[\![G \gg H;\ G \gg N;\ a \in H]\!] \Longrightarrow a \diamondsuit N \subseteq H \diamond_G$

N
⟨*proof*⟩

lemma (**in** *Group*) *rcs-sub-smult*:$[\![G \gg H;\ G \gg N;\ a \in H]\!] \Longrightarrow N \cdot a \subseteq N \diamond_G H$
⟨*proof*⟩

lemma (**in** *Group*) *smult-commute-sg-nsg*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow H \diamond_G N = N \diamond_G H$
⟨*proof*⟩

lemma (**in** *Group*) *smult-sg-nsg*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow G \gg H \diamond_G N$
⟨*proof*⟩

lemma (**in** *Group*) *smult-nsg-sg*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow G \gg N \diamond_G H$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-smult-sg-nsg*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow Group\ (Gp\ G\ (H \diamond_G N))$
⟨*proof*⟩

lemma (**in** *Group*) *N-sg-HN*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow Gp\ G\ (H \diamond_G N) \gg N$
⟨*proof*⟩

lemma (**in** *Group*) *K-absorb-HK*:$[\![G \gg H;\ G \gg K;\ H \subseteq K]\!] \Longrightarrow H \diamond_G K = K$
⟨*proof*⟩

lemma (**in** *Group*) *nsg-Gp-nsg*:$[\![G \gg H;\ G \rhd N;\ N \subseteq H]\!] \Longrightarrow Gp\ G\ H \rhd N$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-smult-nsg*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow Gp\ G\ (H \diamond_G N) \rhd N$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-smult-nsg1*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow Gp\ G\ (N \diamond_G H) \rhd N$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-3*:$[\![G \gg H;\ G \gg H1;\ Gp\ G\ H \rhd H1]\!] \Longrightarrow H1 \subseteq H$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-4*:$[\![G \gg H;\ G \gg H1;\ Gp\ G\ H \rhd H1;\ h \in H;\ h1 \in H1]\!] \Longrightarrow h \cdot h1 \cdot (\varrho\ h) \in H1$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr1*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G\ H \rhd H1;\ Gp\ G\ K \rhd K1]\!] \Longrightarrow H1 \diamond_G (H \cap K1) = (H \cap K1) \diamond_G H1$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr1-1*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G\ H \rhd H1;\ Gp\ G\ K \rhd K1]\!] \Longrightarrow G \gg (H1 \diamond_G (H \cap K1))$

⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ Gp\ G\ H \triangleright H1]\!] \Longrightarrow$
$H1 \diamond_G (H \cap K) = (H \cap K) \diamond_G H1$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-1*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ Gp\ G\ H \triangleright H1]\!]$
$\Longrightarrow G \gg H1 \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-2*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;$
$Gp\ G\ H \triangleright H1;\ Gp\ G\ K \triangleright K1]\!] \Longrightarrow H1 \diamond_G (H \cap K1) \subseteq\ H1 \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-5*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G$
$H \triangleright H1;$
$Gp\ G\ K \triangleright K1;\ a{\in}\ H1;\ b \in H \cap K1;\ c \in H1]\!] \Longrightarrow$
$a \cdot b \cdot c \in H1 \diamond_G (H \cap K1)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-6*:$[\![u \in carrier\ G;\ v \in carrier\ G;$
$x \in carrier\ G;\ y \in carrier\ G]\!] \Longrightarrow$
$(u \cdot v) \cdot (x \cdot y) \cdot (\varrho\ (u \cdot v)) =$
$u \cdot v \cdot x \cdot (\varrho\ v) \cdot (v \cdot y \cdot (\varrho\ v)) \cdot (\varrho\ u)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-7*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ y \in carrier$
$G]\!]$
$\Longrightarrow a \cdot (\ x \cdot y) \cdot (\varrho\ a) = a \cdot x \cdot (\varrho\ a) \cdot (a \cdot y \cdot (\varrho\ a))$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G\ H$
$\triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow\ Gp\ G\ (H1 \diamond_G (H \cap K)) \triangleright (H1 \diamond_G (H \cap K1))$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3-2*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G$
$H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow G \gg H1 \diamond_G (H \cap K1) \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3-3*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G$
$H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow (H1 \cap K) \diamond_G (H \cap K1) = (K1 \cap H) \diamond_G (K \cap H1)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3-4*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G$
$H \triangleright H1;$
$Gp\ G\ K \triangleright K1;\ g \in H \cap K;\ h \in H \cap K1]\!] \Longrightarrow g \cdot h \cdot (\varrho\ g) \in H \cap K1$

$\langle proof \rangle$

lemma (**in** *Group*) *ZassenhausTr3-5*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow (Gp\ G\ (H \cap K)) \triangleright (H1 \cap K) \diamond_G (H \cap K1)$
$\langle proof \rangle$

lemma (**in** *Group*) *ZassenhausTr4*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow (H1 \diamond_G (H \cap K1)) \diamond_G (H1 \diamond_G (H \cap K)) = H1 \diamond_G (H \cap K)$
$\langle proof \rangle$

lemma (**in** *Group*) *ZassenhausTr4-0*: $[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow\ H1 \diamond_G (H \cap K) = (H1 \diamond_G (H \cap K1)) \diamond_G (H \cap K)$
$\langle proof \rangle$

lemma (**in** *Group*) *ZassenhausTr4-1*:$[\![G \gg H;\ (Gp\ G\ H) \triangleright H1;\ (Gp\ G\ H) \gg (H \cap K)]\!]$
$\Longrightarrow (Gp\ G\ (H1 \diamond_G (H \cap K))) \triangleright H1$
$\langle proof \rangle$

3.7 Homomorphism

lemma *gHom*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ;\ x \in carrier\ F;$
$y \in carrier\ F]\!] \Longrightarrow f\ (x \cdot_F y) = (f\ x) \cdot_G (f\ y)$
$\langle proof \rangle$

lemma *gHom-mem*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ;\ x \in carrier\ F]\!] \Longrightarrow$
$(f\ x) \in carrier\ G$
$\langle proof \rangle$

lemma *gHom-func*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G]\!] \Longrightarrow$
$f \in carrier\ F \to carrier\ G$
$\langle proof \rangle$

lemma *gHomcomp*:$[\![Group\ F;\ Group\ G;\ Group\ H;\ f \in gHom\ F\ G;\ g \in gHom\ G\ H]\!]$
$\Longrightarrow (g \circ_F f) \in gHom\ F\ H$
$\langle proof \rangle$

lemma *gHom-comp-gsurjec*:$[\![Group\ F;\ Group\ G;\ Group\ H;\ gsurj_{F,G}\ f;$
$gsurj_{G,H}\ g]\!] \Longrightarrow gsurj_{F,H}\ (g \circ_F f)$
$\langle proof \rangle$

lemma *gHom-comp-ginjec*:$[\![Group\ F;\ Group\ G;\ Group\ H;\ ginj_{F,G}\ f;\ ginj_{G,H}\ g]\!]$
$\Longrightarrow$

$ginj_{F,H}\ (g \circ_F f)$
⟨proof⟩

lemma *ghom-unit-unit*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow f\ (\mathbf{1}_F) = \mathbf{1}_G$
⟨proof⟩

lemma *ghom-inv-inv*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ;\ x \in carrier\ F]\!] \Longrightarrow f\ (\varrho_F\ x) = \varrho_G\ (f\ x)$
⟨proof⟩

lemma *ghomTr3*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ;\ x \in carrier\ F;\ y \in carrier\ F;\ f\ (x \cdot_F (\varrho_F\ y)) = \mathbf{1}_G\ ]\!] \Longrightarrow f\ x = f\ y$
⟨proof⟩

lemma *iim-nonempty*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ G \gg K]\!] \Longrightarrow (iim\ F\ G\ f\ K) \neq \{\}$
⟨proof⟩

lemma *ghomTr4*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ G \gg K]\!] \Longrightarrow F \gg (iim\ F\ G\ f\ K)$
⟨proof⟩

lemma (**in** *Group*) *IdTr0*: $idmap\ (carrier\ G) \in gHom\ G\ G$
⟨proof⟩

abbreviation
IDMAP ((*I*_) [*999*]*1000*) **where**
$I_F == idmap\ (carrier\ F)$

abbreviation
INVFUN ((*3Ifn* - - -) [*88*,*88*,*89*]*88*) **where**
$Ifn\ F\ G\ f == invfun\ (carrier\ F)\ (carrier\ G)\ f$

lemma *IdTr1*: $[\![Group\ F;\ x \in carrier\ F]\!] \Longrightarrow (I_F)\ x = x$
⟨proof⟩

lemma *IdTr2*: $Group\ F \Longrightarrow gbij_{F,F}\ (I_F)$
⟨proof⟩

lemma *Id-l-unit*: $[\![Group\ G;\ gbij_{G,G}\ f]\!] \Longrightarrow I_G \circ_G f = f$
⟨proof⟩

3.8 Gkernel

lemma *gkernTr1*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ x \in gker_{F,G}\ f]\!] \Longrightarrow x \in carrier\ F$
⟨proof⟩

lemma *gkernTr1-1*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G*$]\!] \Longrightarrow$ $gker_{F,G}\, f \subseteq$ *carrier F*
⟨*proof*⟩

lemma *gkernTr2*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G*; $x \in gker_{F,G}\, f$; $y \in gker_{F,G}\, f]\!]$
$\Longrightarrow (x \cdot_F y) \in gker_{F,G}\, f$
⟨*proof*⟩

lemma *gkernTr3*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G* ; $x \in gker_{F,G}\, f]\!] \Longrightarrow$
$(\varrho_F\ x) \in gker_{F,G}\, f$
⟨*proof*⟩

lemma *gkernTr6*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G*$]\!] \Longrightarrow (\mathbf{1}_F) \in gker_{F,G}\, f$
⟨*proof*⟩

lemma *gkernTr7*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G* $]\!] \Longrightarrow F \gg gker_{F,G}\, f$
⟨*proof*⟩

lemma *gker-normal*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G*$]\!] \Longrightarrow F \triangleright gker_{F,G}\, f$
⟨*proof*⟩

lemma *Group-coim*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G*$]\!] \Longrightarrow$ *Group* (F / $gker_{F,G}\, f$)
⟨*proof*⟩

lemma *gkern1*:$[\![$*Group F*; *Ugp E*; $f \in$ *gHom F E*$]\!] \Longrightarrow gker_{F,E}\, f =$ *carrier F*
⟨*proof*⟩

lemma *gkern2*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G*; $ginj_{F,G}\, f]\!] \Longrightarrow$
$gker_{F,G}\, f = \{\mathbf{1}_F\}$
⟨*proof*⟩

lemma *gkernTr9*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G*; $a \in$ *carrier F*; $b \in$ *carrier F*$]\!]$
$\Longrightarrow ((gker_{F,G}\, f) \bullet_F a = (gker_{F,G}\, f) \bullet_F b) = (f\ a = f\ b)$

⟨*proof*⟩

lemma *gkernTr11*:$[\![$*Group F*; *Group G*; $f \in$ *gHom F G* ; $a \in$ *carrier F*$]\!] \Longrightarrow$
$(iim\ F\ G\ f\ \{f\ a\}) = (gker_{F,G}\, f) \bullet_F a$
⟨*proof*⟩

lemma *gbij-comp-bij*:$[\![$*Group F*; *Group G*; *Group H*; $gbij_{F,G}\, f$; $gbij_{G,H}\, g]\!]$
$\Longrightarrow gbij_{F,H}\, (g \circ_F f)$
⟨*proof*⟩

lemma *gbij-automorph*:$[\![$*Group G*; $gbij_{G,G}\, f$; $gbij_{G,G}\, g]\!] \Longrightarrow$
$gbij_{G,G}\, (g \circ_G f)$

$\langle proof \rangle$

lemma *l-unit-gHom*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G]\!] \Longrightarrow (I_G) \circ_F f = f$
$\langle proof \rangle$

lemma *r-unit-gHom*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow f \circ_F (I_F) = f$
$\langle proof \rangle$

3.9 Image

lemma *inv-gHom*:$[\![Group\ F;\ Group\ G;\ gbij_{F,G}\ f]\!] \Longrightarrow (Ifn\ F\ G\ f) \in gHom\ G\ F$
$\langle proof \rangle$

lemma *inv-gbijec-gbijec*:$[\![Group\ F;\ Group\ G;\ gbij_{F,G}\ f]\!] \Longrightarrow gbij_{G,F}\ (Ifn\ F\ G\ f)$
$\langle proof \rangle$

lemma *l-inv-gHom*:$[\![Group\ F;\ Group\ G;\ gbij_{F,G}\ f]\!] \Longrightarrow (Ifn\ F\ G\ f) \circ_F f = (I_F)$
$\langle proof \rangle$

lemma *img-mult-closed*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ u \in f\ `(carrier\ F);$
$v \in f\ `(carrier\ F)]\!] \Longrightarrow u \cdot_G v \in f\ `(carrier\ F)$
$\langle proof \rangle$

lemma *img-unit-closed*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G]\!] \Longrightarrow$
$\mathbf{1}_G \in f\ `(carrier\ F)$
$\langle proof \rangle$

lemma *imgTr7*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ u \in f\ `(carrier\ F)]\!]$
$\Longrightarrow \varrho_G\ u\ \in f\ `(carrier\ F)$
$\langle proof \rangle$

lemma *imgTr8*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \gg H;\ u \in f\ `\ H;$
$v \in f`\ H\ ]\!] \Longrightarrow u \cdot_G v \in f\ `\ H$
$\langle proof \rangle$

lemma *imgTr9*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \gg H;\ u \in f\ `\ H]\!] \Longrightarrow$
$\varrho_G\ u \in f\ `\ H$
$\langle proof \rangle$

lemma *imgTr10*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \gg H]\!] \Longrightarrow \mathbf{1}_G \in f\ `\ H$
$\langle proof \rangle$

lemma *imgTr11*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \gg H]\!] \Longrightarrow G \gg (f\ `\ H)$
$\langle proof \rangle$

lemma *sg-gimg*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!]\ \Longrightarrow G \gg f`(carrier\ F)$
$\langle proof \rangle$

lemma *Group-Img*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow Group\ (Img_{F,G}\ f)$

⟨*proof*⟩

lemma *Img-carrier*:⟦*Group F*; *Group G*; *f* ∈ *gHom F G* ⟧ ⟹
carrier ($Img_{F,G}$ *f*) = *f* ‘ (*carrier F*)
⟨*proof*⟩

lemma *hom-to-Img*:⟦*Group F*; *Group G*; *f* ∈ *gHom F G*⟧ ⟹ *f* ∈ *gHom F* ($Img_{F,G}$ *f*)
⟨*proof*⟩

lemma *gker-hom-to-img*:⟦*Group F*; *Group G*; *f* ∈ *gHom F G* ⟧ ⟹
$gker_{F,(Img_{F,G}\, f)}$ *f* = $gker_{F,G}$ *f*
⟨*proof*⟩

lemma *Pj-im-subg*:⟦*Group G*; *G* ≫ *H*; *G* ⊳ *K*; *K* ⊆ *H*⟧ ⟹
Pj G K ‘ *H* = *carrier* ((*Gp G H*) / *K*)
⟨*proof*⟩

lemma (**in** *Group*) *subg-Qsubg*:⟦*G* ≫ *H*; *G* ⊳ *K*; *K* ⊆ *H*⟧ ⟹
(*G* / *K*) ≫ *carrier* ((*Gp G H*) / *K*)
⟨*proof*⟩

3.10 Induced homomorphisms

lemma *inducedhomTr*:⟦*Group F*; *Group G*; *f* ∈ *gHom F G*;
S ∈ *set-rcs F* ($gker_{F,G}$ *f*); *s1* ∈ *S*; *s2* ∈ *S* ⟧ ⟹ *f s1* = *f s2*
⟨*proof*⟩

definition
induced-ghom :: [(′*a*, ′*more*) *Group-scheme*, (′*b*, ′*more1*) *Group-scheme*,
(′*a* ⇒ ′*b*)] ⇒ (′*a set* ⇒ ′*b*) **where**
induced-ghom F G f = (λ*X*∈ (*set-rcs F* ($gker_{F,G}$ *f*)). *f* (*SOME x*. *x* ∈ *X*))

abbreviation
INDUCED-GHOM :: [′*a* ⇒ ′*b*, (′*a*, ′*m*) *Group-scheme*, (′*b*, ′*m1*) *Group-scheme*]
⇒ (′*a set* ⇒ ′*b*) ((*3*-¨-,-) [*82*,*82*,*83*]*82*) **where**
$f^{\circ\circ}{}_{F,G}$ == *induced-ghom F G f*

lemma *induced-ghom-someTr*:⟦*Group F*; *Group G*; *f* ∈ *gHom F G*;
X ∈ *set-rcs F* ($gker_{F,G}$ *f*)⟧ ⟹ *f* (*SOME xa*. *xa* ∈ *X*) ∈ *f* ‘(*carrier F*)
⟨*proof*⟩

lemma *induced-ghom-someTr1*:⟦*Group F*; *Group G*; *f* ∈ *gHom F G*; *a* ∈ *carrier F*⟧ ⟹
f (*SOME xa*. *xa* ∈ ($gker_{F,G}$ *f*) $\cdot_F$ *a*) = *f a*
⟨*proof*⟩

lemma *inducedHOMTr0*:⟦*Group F*; *Group G*; *f* ∈ *gHom F G*; *a* ∈ *carrier F*⟧ ⟹

$(f^{\cdot\cdot}{}_{F,G})$ $((gker_{F,G}\ f) \cdot_F\ a) = f\ a$
⟨proof⟩

lemma *inducedHOMTr0-1*:⟦*Group F*; *Group G*; $f \in$ *gHom F G*⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in$ *set-rcs F* $(gker_{F,G}\ f) \rightarrow$ *carrier G*
⟨proof⟩

lemma *inducedHOMTr0-2*:⟦*Group F*; *Group G*; $f \in$ *gHom F G*⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in$ *set-rcs F* $(gker_{F,G}\ f) \rightarrow f$ ‘ (*carrier F*)
⟨proof⟩

lemma *inducedHom*:⟦*Group F*; *Group G*; $f \in$ *gHom F G*⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in$ *gHom* $(F/(gker_{F,G}\ f))$ *G*
⟨proof⟩

lemma *induced-ghom-ginjec*: ⟦*Group F*; *Group G*; $f \in$ *gHom F G* ⟧ ⟹
$ginj_{(F/(gker_{F,G}\ f)),G}\ (f^{\cdot\cdot}{}_{F,G})$
⟨proof⟩

lemma *inducedhomgsurjec*:⟦*Group F*; *Group G*; $gsurj_{F,G}\ f$⟧ ⟹
$gsurj_{(F/(gker_{F,G}\ f)),G}\ (f^{\cdot\cdot}{}_{F,G})$
⟨proof⟩

lemma *homomtr*: ⟦*Group F*; *Group G*; $f \in$ *gHom F G*⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in$ *gHom* $(F\ /\ (gker_{F,G}\ f))\ (Img_{F,G}\ f)$
⟨proof⟩

lemma *homom2img*: ⟦*Group F*; *Group G*; $f \in$ *gHom F G* ⟧ ⟹
$(f^{\cdot\cdot}{}_{F,(Img_{F,G}\ f)}) \in$ *gHom* $(F\ /\ (gker_{F,G}\ f))\ (Img_{F,G}\ f)$
⟨proof⟩

lemma *homom2img1*:⟦*Group F*; *Group G*; $f \in$ *gHom F G*; $X \in$ *set-rcs F* $(gker_{F,G}$
$f)$⟧
⟹ $(f^{\cdot\cdot}{}_{F,(Img_{F,G}\ f)})\ X = (f^{\cdot\cdot}{}_{F,G})\ X$
⟨proof⟩

3.10.1 Homomorphism therems

definition
iota :: $('a, 'm)$ *Group-scheme* ⇒ $('a \Rightarrow 'a)$

$((\iota_{\text{-}}))$ [*1000*]*999*) **where**
$\iota_F = (\lambda x \in carrier\ F.\ x)$

lemma *iotahomTr0*:⟦*Group G*; $G \gg H$; $h \in H$ ⟧ ⟹ $(\iota_{(Gp\ G\ H)})\ h = h$
⟨proof⟩

lemma *iotahom*:⟦*Group G*; $G \gg H$; $G \triangleright N$⟧ ⟹

$\iota_{(Gp\ G\ H)} \in gHom\ (Gp\ G\ H)\ (Gp\ G\ (H \diamond_G N))$
⟨proof⟩

lemma *iotaTr0*: $[\![Group\ G;\ G \gg H;\ G \triangleright N]\!] \Longrightarrow$
$ginj_{(Gp\ G\ H),(Gp\ G\ (H \diamond_G N))}\ (\iota_{(Gp\ G\ H)})$
⟨proof⟩

theorem *homomthm1*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow$
$gbij_{(F/\ (gkernel\ F\ G\ f)),\ (Gimage\ F\ G\ f)}\ (f^{\cdot\cdot}{}_{F,\ (Gimage\ F\ G\ f)})$
⟨proof⟩

lemma *isomTr0* [*simp*]:$Group\ F \Longrightarrow F \cong F$
⟨proof⟩

lemma *isomTr1*:$[\![Group\ F;\ Group\ G;\ F \cong\ \ G\ ]\!] \Longrightarrow G \cong F$
⟨proof⟩

lemma *isomTr2*:$[\![Group\ F;\ Group\ G;\ Group\ H;\ F \cong G;\ G \cong H\ ]\!] \Longrightarrow F \cong H$
⟨proof⟩

lemma *gisom1*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow$
$(F/\ (gker_{F,G}\ f)) \cong (Img_{F,G}\ f)$
⟨proof⟩

lemma *homomth2Tr0*: $[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ G \triangleright N]\!] \Longrightarrow$
$F \triangleright (iim\ F\ G\ f\ N)$
⟨proof⟩

lemma *kern-comp-gHom*:$[\![Group\ F;\ Group\ G;\ gsurj_{F,G}\ f;\ G \triangleright N]\!] \Longrightarrow$
$gker_{F,\ (G/N)}\ ((Pj\ G\ N) \circ_F f) = iim\ F\ G\ f\ N$
⟨proof⟩

lemma *QgrpUnit-1*:$[\![Group\ G;\ Ugp\ E;\ G \triangleright H;\ (G\ /\ H) \cong E\ ]\!] \Longrightarrow carrier\ G =$
H
⟨proof⟩

lemma *QgrpUnit-2*:$[\![Group\ G;\ Ugp\ E;\ G \triangleright H;\ carrier\ G = H]\!] \Longrightarrow (G/H) \cong E$
⟨proof⟩

lemma *QgrpUnit-3*:$[\![Group\ G;\ Ugp\ E;\ G \gg H;\ G \gg H1;\ (Gp\ G\ H) \triangleright H1;$
$((Gp\ G\ H)\ /\ H1) \cong E\ ]\!] \Longrightarrow H = H1$
⟨proof⟩

lemma *QgrpUnit-4*:$[\![Group\ G;\ Ugp\ E;\ G \gg H;\ G \gg H1;\ (Gp\ G\ H) \triangleright H1;$
$\neg\ ((Gp\ G\ H)\ /\ H1) \cong E\ ]\!] \Longrightarrow H \neq H1$
⟨proof⟩

definition

Qmp :: [('*a*, '*m*) *Group-scheme*, '*a set*, '*a set*] ⇒ ('*a set* ⇒ '*a set*) **where**
Qmp G H N = ($\lambda X \in$ *set-rcs G H*. $\{z.\ \exists\ x \in X.\ \exists\ y \in N.\ (y \cdot_G x = z)\}$)

abbreviation
QP :: [-, '*a set*, '*a set*] ⇒ ('*a set* ⇒ '*a set*)
((*3Qm*- -,-) [*82*,*82*,*83*]*82*) **where**
$Qm_{G\ H,N}$ == *Qmp G H N*

lemma (**in** *Group*) *QmpTr0*:⟦$G \gg H$; $G \gg N$; $H \subseteq N$; $a \in$ *carrier G*⟧ ⟹
Qmp G H N ($H \cdot a$) = ($N \cdot a$)
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr1*:⟦$G \gg H$; $G \gg N$; $H \subseteq N$; $a \in$ *carrier G*; $b \in$ *carrier G*;
$H \cdot a = H \cdot b$⟧ ⟹ $N \cdot a = N \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr2*:⟦$G \gg H$; $G \gg N$; $H \subseteq N$; $X \in$ *carrier* (G/H)⟧
⟹ (*Qmp G H N*) $X \in$ *carrier* (G/N)
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr2-1*:⟦$G \gg H$; $G \gg N$; $H \subseteq N$ ⟧ ⟹
Qmp G H N ∈ *carrier* (G/H) → *carrier* (G/N)
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr3*:⟦$G \rhd H$; $G \rhd N$; $H \subseteq N$; $X \in$ *carrier* (G/H);
$Y \in$ *carrier* (G/H)⟧ ⟹
(*Qmp G H N*) (*c-top G H X Y*) = *c-top G N* ((*Qmp G H N*) *X*) ((*Qmp G H N*) *Y*)
⟨*proof*⟩

lemma (**in** *Group*) *Gp-s-mult-nsg*:⟦$G \rhd H$; $G \rhd N$; $H \subseteq N$; $a \in N$ ⟧ ⟹
$H \cdot_{(Gp\ G\ N)} a = H \cdot a$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr5*:⟦$G \rhd H$; $G \rhd N$; $H \subseteq N$; $X \in$ *carrier* (G/H);
$Y \in$ *carrier* (G/H) ⟧ ⟹ (*Qmp G H N*) ($X \cdot_{(G\ /\ H)} Y$) =
((*Qmp G H N*) *X*) $\cdot_{(G\ /\ N)}$ ((*Qmp G H N*) *Y*)
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr*:⟦$G \rhd H$; $G \rhd N$; $H \subseteq N$ ⟧ ⟹
($Qm_{G\ H,N}$) ∈ *gHom* (G / H) (G / N)
⟨*proof*⟩

lemma (**in** *Group*) *Qmpgsurjec*:$[\![G \rhd H;\ G \rhd N;\ H \subseteq N\ ]\!] \Longrightarrow$
$gsurj_{(G\ /\ H),(G\ /\ N)}\ (Qm_{G\ H,N})$
$\langle proof \rangle$

lemma (**in** *Group*) *gkerQmp*:$[\![G \rhd H;\ G \rhd N;\ H \subseteq N\ ]\!] \Longrightarrow$
$gker_{(G\ /\ H),(G\ /\ N)}\ (Qm_{G\ H,N}) = carrier\ ((Gp\ G\ N)/\ H)$
$\langle proof \rangle$

theorem (**in** *Group*) *homom2*:$[\![G \rhd H;\ G \rhd N;\ H \subseteq N]\!] \Longrightarrow$
$gbij_{((G/H)/(carrier\ ((Gp\ G\ N)/H))),(G/N)}\ ((Qm_{G\ H,N})^{\circ\circ}{}_{(G/H),(G/N)})$
$\langle proof \rangle$

3.11 Isomorphims

theorem (**in** *Group*) *isom2*:$[\![G \rhd H;\ G \rhd N;\ H \subseteq N]\!] \Longrightarrow$
$((G/H)/(carrier\ ((Gp\ G\ N)/H))) \cong\ (G/N)$
$\langle proof \rangle$

theorem *homom3*:$[\![$ *Group F*; *Group G*; $G \rhd N;\ gsurj_{F,G}\ f;$
$N1 = (iim\ F\ G\ f)\ N\ ]\!] \Longrightarrow (F\ /\ N1) \cong (G\ /\ N)$
$\langle proof \rangle$

lemma (**in** *Group*) *homom3Tr1*:$[\![G \gg H;\ G \rhd N]\!] \Longrightarrow H \cap N =$
$gker_{(Gp\ G\ H),((Gp\ G\ (H \diamond_G N))/N)}$
$((Pj\ (Gp\ G\ (H \diamond_G N))\ N) \circ_{(Gp\ G\ H)} (\iota_{(Gp\ G\ H)}))$
$\langle proof \rangle$

3.11.1 An automorphism groups

definition
automg :: - $\Rightarrow$
$(\!|\ carrier :: ('a \Rightarrow 'a)\ set,\ top :: ['a \Rightarrow 'a,'a \Rightarrow 'a] \Rightarrow ('a \Rightarrow 'a),$
$iop :: ('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a),\ one :: ('a \Rightarrow 'a)|\!)$ **where**
$automg\ G = \ (\!|\ carrier = \{f.\ gbij_{G,G}\ f\},$
$top = \lambda g{\in}\{f.\ gbij_{G,G}\ f\}.\ \lambda f{\in}\{f.\ gbij_{G,G}\ f\}.\ (\ g \circ_G f),$
$iop = \lambda f{\in}\{f.\ gbij_{G,G}\ f\}.\ (Ifn\ G\ G\ f),\ one = I_G\ |\!)$

lemma *automgroupTr1*:$[\![Group\ G;\ gbij_{G,G}\ f;\ gbij_{G,G}\ g;\ gbij_{G,G}\ h]\!] \Longrightarrow$
$(h \circ_G g) \circ_G f = \ h \circ_G (g \circ_G f)$
$\langle proof \rangle$

lemma *automgroup*:$Group\ G \ \Longrightarrow Group\ (automg\ G)$
$\langle proof \rangle$

3.11.2 Complete system of representatives

definition
gcsrp :: - $\Rightarrow$ $'a\ set \Rightarrow\ 'a\ set \Rightarrow bool$ **where**

gcsrp G H S == ∃ *f*. (*bij-to f* (*set-rcs G H*) *S*)

definition
gcsrp-map::- ⇒ *'a set* ⇒ *'a set* ⇒ *'a* **where**
gcsrp-map G H == *λX*∈(*set-rcs G H*). *SOME x*. *x* ∈ *X*

lemma (**in** *Group*) *gcsrp-func*:*G* ≫ *H* ⟹ *gcsrp-map G H* ∈ *set-rcs G H* → *UNIV*
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-func1*:*G* ≫ *H* ⟹
gcsrp-map G H ∈ *set-rcs G H* → (*gcsrp-map G H*) ‘ (*set-rcs G H*)
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-map-bij*:*G* ≫ *H* ⟹
bij-to (*gcsrp-map G H*) (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*))
⟨*proof*⟩

lemma (**in** *Group*) *image-gcsrp*:*G* ≫ *H* ⟹
gcsrp G H ((*gcsrp-map G H*) ‘(*set-rcs G H*))
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-exists*:*G* ≫ *H* ⟹ ∃ *S*. *gcsrp G H S*
⟨*proof*⟩

definition
gcsrp-top :: [- , *'a set*] ⇒ *'a* ⇒ *'a* ⇒ *'a* **where**
gcsrp-top G H == *λx* ∈ ((*gcsrp-map G H*) ‘(*set-rcs G H*)).
λy ∈ ((*gcsrp-map G H*) ‘(*set-rcs G H*)).
gcsrp-map G H
(*c-top G H*
((*invfun* (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*)) (*gcsrp-map G H*)) *x*)
((*invfun* (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*)) (*gcsrp-map G H*)) *y*))

definition
gcsrp-iop::[- , *'a set*] ⇒ *'a* ⇒ *'a* **where**
gcsrp-iop G H = (*λx* ∈ ((*gcsrp-map G H*) ‘(*set-rcs G H*)).
gcsrp-map G H
(*c-iop G H*
((*invfun* (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*)) (*gcsrp-map G H*)) *x*)))

definition
gcsrp-one::[- , *'a set*] ⇒ *'a* **where**
gcsrp-one G H = *gcsrp-map G H H*

definition
Gcsrp :: - ⇒ *'a set* ⇒ *'a Group* **where**
Gcsrp G N = (|*carrier* = (*gcsrp-map G N*) ‘(*set-rcs G N*),

$top = gcsrp\text{-}top\ G\ N,\ iop = gcsrp\text{-}iop\ G\ N,\ one = gcsrp\text{-}one\ G\ N\|)$

lemma (**in** *Group*) *gcsrp-top-closed*:$[\![Group\ G;\ G \triangleright N;$
$a \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N));\ b \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N))]\!]$
$\Longrightarrow gcsrp\text{-}top\ G\ N\ a\ b \in (gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N)$
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-tassoc*:$[\![Group\ G;\ G \triangleright N;$
$a \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N));$
$b \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N));$
$c \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N))]\!] \Longrightarrow$
$(gcsrp\text{-}top\ G\ N\ (gcsrp\text{-}top\ G\ N\ a\ b)\ c) =$
$(gcsrp\text{-}top\ G\ N\ a\ (gcsrp\text{-}top\ G\ N\ b\ c))$
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-l-one*:$[\![Group\ G;\ G \triangleright N;$
$a \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N))]\!] \Longrightarrow$
$(gcsrp\text{-}top\ G\ N\ (gcsrp\text{-}one\ G\ N)\ a) = a$
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-l-i*:$[\![G \triangleright N;\ a \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N))]\!] \Longrightarrow$
$gcsrp\text{-}top\ G\ N\ (gcsrp\text{-}iop\ G\ N\ a)\ a = gcsrp\text{-}one\ G\ N$
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-i-closed*:$[\![G \triangleright N;\ a \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N))]\!]$
$\Longrightarrow gcsrp\text{-}iop\ G\ N\ a \in ((gcsrp\text{-}map\ G\ N)\ `(set\text{-}rcs\ G\ N))$
⟨*proof*⟩

lemma (**in** *Group*) *Group-Gcsrp*:$G \triangleright N \Longrightarrow Group\ (Gcsrp\ G\ N)$
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-map-gbijec*:$G \triangleright N \Longrightarrow$
$gbij_{(G/N),\ (Gcsrp\ G\ N)}\ (gcsrp\text{-}map\ G\ N)$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-equiv-Gcsrp*:$G \triangleright N \Longrightarrow (G\ /\ N) \cong Gcsrp\ G\ N$
⟨*proof*⟩

3.12 Zassenhaus

we show $H \to H\ N/N$ is gsurjective

lemma (**in** *Group*) *homom4Tr1*:$[\![G \triangleright N;\ G \gg H]\!] \Longrightarrow\ Group\ ((Gp\ G\ (H \diamond_G N))$
$/\ N)$
⟨*proof*⟩

lemma *homom3Tr2*:$[\![Group\ G;\ G \gg H;\ G \triangleright N]\!] \Longrightarrow$
$gsurj_{(Gp\ G\ H),((Gp\ G\ (H \diamond_G N))/N)}$
$((Pj\ (Gp\ G\ (H \diamond_G N))\ N) \circ_{(Gp\ G\ H)} (\iota_{(Gp\ G\ H)}))$

⟨*proof*⟩

theorem *homom4*:⟦*Group G*; *G* ▷ *N*; *G* ≫ *H*⟧ ⟹*gbij*((*Gp G H*)/(*H* ∩ *N*)),((*Gp G* (*H* ⋄*G* *N*)) / *N*) (((*Pj* (*Gp G* (*H* ⋄*G* *N*)) *N*) ∘(*Gp G H*) (ι(*Gp G H*)))¨(*Gp G H*),((*Gp G* (*H* ⋄*G* *N*)) / *N*))

⟨*proof*⟩

lemma (**in** *Group*) *homom4-2*:⟦*G* ▷ *N*; *G* ≫ *H*⟧ ⟹ *Group* ((*Gp G H*) / (*H* ∩ *N*))
⟨*proof*⟩

lemma *isom4*:⟦*Group G*; *G* ▷ *N*; *G* ≫ *H*⟧ ⟹
((*Gp G H*)/(*H* ∩ *N*)) ≅ ((*Gp G* (*N* ⋄*G* *H*)) / *N*)
⟨*proof*⟩

lemma *ZassenhausTr5*:⟦*Group G*; *G* ≫ *H*; *G* ≫ *H1*; *G* ≫ *K*; *G* ≫ *K1*; *Gp G H* ▷ *H1*;
Gp G K ▷ *K1*⟧ ⟹
((*Gp G* (*H* ∩ *K*))/((*H1* ∩ *K*) ⋄*G* (*H* ∩ *K1*))) ≅
((*Gp G* (*H1* ⋄*G* (*H* ∩ *K*)))/(*H1* ⋄*G* (*H* ∩ *K1*)))
⟨*proof*⟩

lemma *ZassenhausTr5-1*:⟦*Group G*; *G* ≫ *H*; *G* ≫ *H1*; *G* ≫ *K*; *G* ≫ *K1*; *Gp G H* ▷ *H1*;
Gp G K ▷ *K1*⟧ ⟹ ((*Gp G* (*K* ∩ *H*))/((*K1* ∩ *H*) ⋄*G* (*K* ∩ *H1*))) ≅
((*Gp G* (*K1* ⋄*G* (*K* ∩ *H*)))/(*K1* ⋄*G* (*K* ∩ *H1*)))

⟨*proof*⟩

lemma *ZassenhausTr5-2*: ⟦*Group G*; *G* ≫ *H*; *G* ≫ *H1*; *G* ≫ *K*; *G* ≫ *K1*; *Gp G H* ▷ *H1*;
Gp G K ▷ *K1*⟧ ⟹
((*Gp G* (*H* ∩ *K*))/((*H1* ∩ *K*) ⋄*G* (*H* ∩ *K1*))) =
((*Gp G* (*K* ∩ *H*))/((*K1* ∩ *H*) ⋄*G* (*K* ∩ *H1*)))
⟨*proof*⟩

lemma *ZassenhausTr6-1*:⟦*Group G*; *G* ≫ *H*; *G* ≫ *H1*; *G* ≫ *K*; *G* ≫ *K1*; *Gp G H* ▷ *H1*;
Gp G K ▷ *K1*⟧ ⟹ *Group* (*Gp G* (*H* ∩ *K*) / (*H1* ∩ *K* ⋄*G* *H* ∩ *K1*))
⟨*proof*⟩

lemma *ZassenhausTr6-2*:⟦*Group G*; *G* ≫ *H*; *G* ≫ *H1*; *G* ≫ *K*; *G* ≫ *K1*; *Gp G H* ▷ *H1*;
Gp G K ▷ *K1*⟧ ⟹ *Group* (*Gp G* (*H1* ⋄*G* *H* ∩ *K*) / (*H1* ⋄*G* *H* ∩ *K1*))
⟨*proof*⟩

lemma *ZassenhausTr6-3*:⟦*Group G*; *G* ≫ *H*; *G* ≫ *H1*; *G* ≫ *K*; *G* ≫ *K1*; *Gp G H* ▷ *H1*;

$Gp\ G\ K \triangleright K1]\!] \Longrightarrow Group\ (Gp\ G\ (K1 \diamond_G K \cap H)\ /\ (K1 \diamond_G K \cap H1))$
⟨proof⟩

theorem *Zassenhaus*: $[\![Group\ G;\ G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow (Gp\ G\ (H1 \diamond_G H \cap K)\ /\ (H1 \diamond_G H \cap K1)) \cong$
$(Gp\ G\ (K1 \diamond_G K \cap H)\ /\ (K1 \diamond_G K \cap H1))$
⟨proof⟩

3.13 Chain of groups I

definition
$d\text{-}gchain :: [\text{-} ,\ nat,\ (nat \Rightarrow 'a\ set)] \Rightarrow bool$ **where**
$d\text{-}gchain\ G\ n\ g = (if\ n{=}0\ then\ G \gg g\ 0\ else\ (\forall l{\le}\ n.\ G \gg (g\ l) \wedge$
$(\forall l \le (n - Suc\ 0).\ g\ (Suc\ l) \subseteq g\ l\ )))$

definition
$D\text{-}gchain :: [\text{-} ,\ nat,\ (nat \Rightarrow 'a\ set)] \Rightarrow bool$ **where**
$D\text{-}gchain\ G\ n\ g = (if\ n = 0\ then\ G \gg (g\ 0)\ else\ (d\text{-}gchain\ G\ n\ g) \wedge$
$(\forall l \le (n - Suc\ 0).\ (g\ (Suc\ l)) \subset (g\ l)))$

definition
$td\text{-}gchain :: [\text{-} ,\ nat,\ (nat \Rightarrow 'a\ set)] \Rightarrow bool$ **where**
$td\text{-}gchain\ G\ n\ g = (if\ n{=}0\ then\ g\ 0 = carrier\ G \wedge g\ 0 = \{\mathbf{1}_G\}\ else$
$d\text{-}gchain\ G\ n\ g \wedge g\ 0 = carrier\ G \wedge g\ n = \{\mathbf{1}_G\})$

definition
$tD\text{-}gchain :: [\text{-},\ nat,\ (nat \Rightarrow 'a\ set)] \Rightarrow bool$ **where**
$tD\text{-}gchain\ G\ n\ g = (if\ n{=}0\ then\ g\ 0 = carrier\ G \wedge g\ 0 = \{\mathbf{1}_G\}\ else$
$D\text{-}gchain\ G\ n\ g \wedge (g\ 0 = carrier\ G) \wedge (g\ n = \{\mathbf{1}_G\}))$

definition
$w\text{-}cmpser :: [\text{-} ,\ nat,\ (nat \Rightarrow 'a\ set)] \Rightarrow bool$ **where**
$w\text{-}cmpser\ G\ n\ g = (if\ n = 0\ then\ d\text{-}gchain\ G\ n\ g\ else\ d\text{-}gchain\ G\ n\ g \wedge$
$(\forall l \le (n - 1).\ (Gp\ G\ (g\ l)) \triangleright (g\ (Suc\ l))))$

definition
$W\text{-}cmpser :: [\text{-} ,\ nat,\ (nat \Rightarrow 'a\ set)] \Rightarrow bool$ **where**
$W\text{-}cmpser\ G\ n\ g = (if\ n = 0\ then\ d\text{-}gchain\ G\ 0\ g\ else\ D\text{-}gchain\ G\ n\ g \wedge$
$(\forall l \le (n - 1).\ (Gp\ G\ (g\ l)) \triangleright (g\ (Suc\ l))))$

definition
$tw\text{-}cmpser :: [\text{-} ,\ nat,\ (nat \Rightarrow 'a\ set)] \Rightarrow bool$ **where**

tw-cmpser G n g = (*if n* = *0 then td-gchain G 0 g else td-gchain G n g* ∧
(∀ *l* ≤ (*n* − *1*). (*Gp G* (*g l*)) ▷ (*g* (*Suc l*))))

definition
tW-cmpser :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
tW-cmpser G n g = (*if n* = *0 then td-gchain G 0 g else tD-gchain G n g* ∧
(∀ *l* ≤ (*n* − *1*). (*Gp G* (*g l*)) ▷ (*g* (*Suc l*))))

definition
Qw-cmpser :: [- , *nat* ⇒ *'a set*] ⇒ (*nat* ⇒ (*'a set*) *Group*) **where**
Qw-cmpser G f l = ((*Gp G* (*f l*)) / (*f* (*Suc l*)))

definition
red-chn :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ (*nat* ⇒ *'a set*) **where**
red-chn G n f = (*SOME g. g* ∈ {*h*.(*tW-cmpser G* (*card* (*f* ' {*i. i* ≤ *n*}) − *1*) *h*)
∧ *h* '{*i. i* ≤ (*card* (*f* ' {*i. i* ≤ *n*}) − *1*)} = *f* '{*i. i* ≤ *n*}})

definition
chain-cutout :: [*nat*, (*nat* ⇒ *'a set*)] ⇒ (*nat* ⇒ *'a set*) **where**
chain-cutout l f = (λ*j. f* (*slide l j*))

lemma (**in** *Group*) *d-gchainTr0*:⟦*0* < *n*; *d-gchain G n f*; *k* ≤ (*n* − *1*)⟧
⟹ *f* (*Suc k*) ⊆ *f k*
⟨*proof*⟩

lemma (**in** *Group*) *d-gchain-mem-sg*:*d-gchain G n f* ⟹ ∀ *i*≤ *n*. *G* ≫ (*f i*)
⟨*proof*⟩

lemma (**in** *Group*) *d-gchain-pre*:*d-gchain G* (*Suc n*) *f* ⟹ *d-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *d-gchainTr1*:*0* < *n* ⟶ (∀ *f*. *d-gchain G n f* ⟶
(∀ *l* ≤ *n*. ∀ *j* ≤ *n*. *l* < *j* ⟶ *f j* ⊆ *f l*))
⟨*proof*⟩

lemma (**in** *Group*) *d-gchainTr2*:⟦*0* < *n*; *d-gchain G n f*; *l* ≤ *n*; *j* ≤ *n*; *l* ≤ *j* ⟧
⟹ *f j* ⊆ *f l*
⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainTr1*:⟦*d-gchain G n f*;
f l ∈ (*f* ' {*i. i* ≤ *n*}) − {*f 0*}⟧ ⟹
f (*LEAST j. f j* ∈ (*f* ' {*i. i* ≤ *n*}) − {*f 0*}) ∈ (*f* ' {*i. i* ≤ *n*} − {*f 0*})
⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainTr1-0*:⟦*d-gchain G n f*;

$f\ l \in (f\ `\ \{i.\ i \le n\}) - \{f\ 0\}]\!] \implies$
$0 < (LEAST\ j.\ f\ j \in (f\ `\ \{i.\ i \le n\}) - \{f\ 0\})$

⟨proof⟩

lemma (**in** *Group*) *im-d-gchainTr1-1*:
$[\![d\text{-}gchain\ G\ n\ f;\ \exists\ i.\ f\ i \in (f\ `\ \{i.\ i \le n\}) - \{f\ 0\}]\!] \implies$
$f\ (LEAST\ j.\ f\ j \in ((f\ `\ \{i.\ i \le n\}) - \{f\ 0\})) \in ((f`\ \{i.\ i \le n\}) - \{f\ 0\})$

⟨proof⟩

lemma (**in** *Group*) *im-d-gchainsTr1-2*:
$[\![d\text{-}gchain\ G\ n\ f;\ i \le n;\ f\ i \in f\ `\{i.\ i \le n\} - \{f\ 0\}]\!] \implies$
$(LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\} - \{f\ 0\})) \le i$

⟨proof⟩

lemma (**in** *Group*) *im-d-gchainsTr1-3*:$[\![d\text{-}gchain\ G\ n\ f;\ \exists\, i \le n.$
$f\ i \in f`\ \{i.\ i \le n\} - \{f\ 0\};$
$k < (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\} - \{f\ 0\}))]\!] \implies f\ k = f\ 0$

⟨proof⟩

lemma (**in** *Group*) *im-gdchainsTr1-4*:$[\![d\text{-}gchain\ G\ n\ f;$
$\exists\, v{\in}f\ `\{i.\ i \le n\}.\ v \notin \{f\ 0\};\ i < (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\}) \wedge$
$f\ j \ne f\ 0)\ ]\!] \implies f\ i = f\ 0$

⟨proof⟩

lemma (**in** *Group*) *im-d-gchainsTr1-5*:$[\![0 < n;\ d\text{-}gchain\ G\ n\ f;\ i \le n;$
$f\ i \in (f\ `\ \{i.\ i \le n\} - \{f\ 0\});\ (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\} - \{f\ 0\})) = j]\!]$
$\implies \quad f\ `\{i.\ i \le (j - (Suc\ 0))\} = \{f\ 0\}$

⟨proof⟩

lemma (**in** *Group*) *im-d-gchains1*:$[\![0 < n;\ d\text{-}gchain\ G\ n\ f;\ i \le n;$
$f\ i \in (f\ `\ \{i.\ i \le n\} - \{f\ 0\});$
$(LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\} - \{f\ 0\})) = j\ ]\!] \implies$
$f\ `\{i.\ i \le n\} = \{f\ 0\} \cup \{f\ i\ |i.\ j \le i \wedge i \le n\}$

⟨proof⟩

lemma (**in** *Group*) *im-d-gchains1-1*:$[\![d\text{-}gchain\ G\ n\ f;\ f\ n \ne f\ 0]\!] \implies$
$f\ `\{i.\ i \le n\} = \{f\ 0\} \cup$
$\{f\ i\ |i.\ (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\} - \{f\ 0\})) \le i \wedge i \le n\}$

⟨proof⟩

lemma (**in** *Group*) *d-gchains-leastTr*:$[\![d\text{-}gchain\ G\ n\ f;\ f\ n \ne f\ 0]\!] \implies$
$(LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\} - \{f\ 0\})) \in \{i.\ i \le n\} \wedge$
$f\ \ (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \le n\} - \{f\ 0\})) \ne f\ 0$

⟨proof⟩

lemma (**in** *Group*) *im-d-gchainTr2*:$[\![d\text{-}gchain\ G\ n\ f;\ j \le n;\ f\ j \ne f\ 0]\!] \implies$
$\forall\, i \le n.\ f\ 0 = f\ i \longrightarrow \neg\ j \le i$

⟨proof⟩

lemma (**in** *Group*) *D-gchain-pre*:$[\![$*D-gchain G* (*Suc n*) *f*$]\!] \Longrightarrow$ *D-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *D-gchain0*:$[\![$*D-gchain G n f*; $i \le n$; $j \le n$; $i < j$$]\!] \Longrightarrow$
$f\ j \subset f\ i$
⟨*proof*⟩

lemma (**in** *Group*) *D-gchain1*:*D-gchain G n f* $\Longrightarrow$ *inj-on f* $\{i.\ i \le n\}$
⟨*proof*⟩

lemma (**in** *Group*) *card-im-D-gchain*:$[\![0 < n$; *D-gchain G n f*$]\!]$
$\Longrightarrow$ *card* $(f\ `\{i.\ i \le n\}) =$ *Suc n*
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-gr*:$[\![0 < r$; *w-cmpser G r f*; $i \le r]\!]$
$\Longrightarrow G \gg (f\ i)$
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-ns*:$[\![0 < r$; *w-cmpser G r f*; $i \le (r - 1)]\!] \Longrightarrow$
$(Gp\ G\ (f\ i)) \triangleright (f\ (Suc\ i))$
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-pre*:*w-cmpser G* (*Suc n*) *f* $\Longrightarrow$ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *W-cmpser-pre*:*W-cmpser G* (*Suc n*) *f* $\Longrightarrow$ *W-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *td-gchain-n*:$[\![$*td-gchain G n f*; *carrier G* $\neq \{\mathbf{1}\}]\!] \Longrightarrow 0 < n$
⟨*proof*⟩

3.14 Existence of reduced chain

lemma (**in** *Group*) *D-gchain-is-d-gchain*:*D-gchain G n f* $\Longrightarrow$ *d-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *joint-d-gchains*:$[\![$*d-gchain G n f*; *d-gchain G m g*;
$g\ 0 \subseteq f\ n\ ]\!] \Longrightarrow$ *d-gchain G* (*Suc* $(n + m)$) (*jointfun n f m g*)
⟨*proof*⟩

lemma (**in** *Group*) *joint-D-gchains*:$[\![$*D-gchain G n f*; *D-gchain G m g*;
$g\ 0 \subset f\ n\ ]\!] \Longrightarrow$ *D-gchain G* (*Suc* $(n + m)$) (*jointfun n f m g*)
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-is-d-gchain*:*w-cmpser G n f* $\Longrightarrow$ *d-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *joint-w-cmpser*:$[\![$*w-cmpser G n f*; *w-cmpser G m g*;
$Gp\ G\ (f\ n) \triangleright (g\ 0)]\!] \Longrightarrow$ *w-cmpser G* (*Suc* $(n + m)$) (*jointfun n f m g*)

⟨*proof*⟩

lemma (**in** *Group*) *W-cmpser-is-D-gchain*: *W-cmpser G n f* $\Longrightarrow$ *D-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *W-cmpser-is-w-cmpser*: *W-cmpser G n f* $\Longrightarrow$ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *tw-cmpser-is-w-cmpser*: *tw-cmpser G n f* $\Longrightarrow$ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *tW-cmpser-is-W-cmpser*: *tW-cmpser G n f* $\Longrightarrow$ *W-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *joint-W-cmpser*: ⟦*W-cmpser G n f*; *W-cmpser G m g*;
(*Gp G* (*f n*)) ▷ (*g 0*); *g 0* ⊂ *f n*⟧ $\Longrightarrow$
W-cmpser G (*Suc* (*n* + *m*)) (*jointfun n f m g*)
⟨*proof*⟩

lemma (**in** *Group*) *joint-d-gchain-n0*: ⟦*d-gchain G n f*; *d-gchain G 0 g*;
g 0 ⊆ *f n* ⟧ $\Longrightarrow$ *d-gchain G* (*Suc n*) (*jointfun n f 0 g*)
⟨*proof*⟩

lemma (**in** *Group*) *joint-D-gchain-n0*: ⟦*D-gchain G n f*; *D-gchain G 0 g*;
g 0 ⊂ *f n* ⟧ $\Longrightarrow$ *D-gchain G* (*Suc n*) (*jointfun n f 0 g*)
⟨*proof*⟩

lemma (**in** *Group*) *joint-w-cmpser-n0*: ⟦*w-cmpser G n f*; *w-cmpser G 0 g*;
(*Gp G* (*f n*)) ▷ (*g 0*)⟧ $\Longrightarrow$ *w-cmpser G* (*Suc n*) (*jointfun n f 0 g*)
⟨*proof*⟩

lemma (**in** *Group*) *joint-W-cmpser-n0*: ⟦*W-cmpser G n f*; *W-cmpser G 0 g*;
(*Gp G* (*f n*)) ▷ (*g 0*); *g 0* ⊂ *f n* ⟧ $\Longrightarrow$
W-cmpser G (*Suc n*) (*jointfun n f 0 g*)
⟨*proof*⟩

definition
simple-Group :: - ⇒ *bool* **where**
simple-Group G ⟷ {*N*. *G* ⋙ *N*} = {*carrier G*, {$\mathbf{1}_G$}}

definition
compseries:: [- , *nat*, *nat* ⇒ *'a set*] ⇒ *bool* **where**
compseries G n f ⟷ *tW-cmpser G n f* ∧ (*if n* = *0 then f 0* = {$\mathbf{1}_G$} *else*
(∀ *i* ≤ (*n* − *1*). (*simple-Group* ((*Gp G* (*f i*))/(*f* (*Suc i*))))))

definition
length-twcmpser :: [- , *nat*, *nat* ⇒ *'a set*] ⇒ *nat* **where**
length-twcmpser G n f = *card* (*f* ‘{*i*. *i* ≤ *n*}) − *Suc 0*

lemma (**in** *Group*) *compseriesTr0*:$[\![$*compseries G n f*; $i \leq n]\!] \Longrightarrow$
$G \gg (f\ i)$
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr1*:*compseries G n f* $\Longrightarrow$ *tW-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr2*:*compseries G n f* $\Longrightarrow$ *f 0* = *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr3*:*compseries G n f* $\Longrightarrow$ *f n* = $\{\mathbf{1}\}$
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr4*:*compseries G n f* $\Longrightarrow$ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *im-jointfun1Tr1*:$\forall\, l \leq n.\ G \gg (f\ l) \Longrightarrow$
$f \in \{i.\ i \leq n\} \rightarrow$ *Collect* (*sg G*)
⟨*proof*⟩

lemma (**in** *Group*) *Nset-Suc-im*:$\forall\, l \leq$ (*Suc n*). $G \gg (f\ l) \Longrightarrow$
insert (*f* (*Suc n*)) (*f* ‘ $\{i.\ i \leq n\}$) = *f* ‘ $\{i.\ i \leq$ (*Suc n*)$\}$
⟨*proof*⟩

definition
NfuncPair-neq-at::[*nat* ⇒ *'a set*, *nat* ⇒ *'a set*, *nat*] ⇒ *bool* **where**
NfuncPair-neq-at f g i $\longleftrightarrow$ $f\ i \neq g\ i$

lemma *LeastTr0*:$[\![$ (*i*::*nat*) < (*LEAST l. P* (*l*))$]\!] \Longrightarrow \neg\ P\ (i)$
⟨*proof*⟩

lemma (**in** *Group*) *funeq-LeastTr1*:$[\![\forall\, l \leq n.\ G \gg f\ l;\ \forall\, l \leq n.\ G \gg g\ l;$
(*l* :: *nat*) < (*LEAST k*. (*NfuncPair-neq-at f g k*)) $]\!] \Longrightarrow f\ l = g\ l$
⟨*proof*⟩

lemma (**in** *Group*) *funeq-LeastTr1-1*:$[\![\forall\, l \leq$ (*n*::*nat*). $G \gg f\ l;\ \forall\, l \leq n.\ G \gg\ g\ l;$
(*l* :: *nat*) < (*LEAST k*. ($f\ k \neq\ g\ k$)) $]\!] \Longrightarrow f\ l = g\ l$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-1*:$[\![i \leq n;\ \forall\, l \leq n.\ G \gg f\ l;\ \forall\, l \leq n.\ G \gg g\ l;$
NfuncPair-neq-at f g i$]\!] \Longrightarrow$
NfuncPair-neq-at f g (*LEAST k*. (*NfuncPair-neq-at f g k*))
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-2*:$[\![i \leq n;\ \forall\, l \leq n.\ G \gg f\ l;\ \forall\, l \leq n.\ G \gg g\ l;$
NfuncPair-neq-at f g i$]\!] \Longrightarrow$
(*LEAST k*. (*NfuncPair-neq-at f g k*)) $\leq i$

$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-LeastTr2-2-1*: $[\![i \le (n::nat); \forall l \le n.\ G \gg f\ l;$
$\forall l \le n.\ G \gg g\ l; f\ i \ne g\ i]\!] \Longrightarrow (LEAST\ k.\ (f\ k \ne g\ k)) \le i$
$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-LeastTr2-3*: $[\![\forall l \le (n::nat).\ G \gg f\ l; \forall l \le n.\ G \gg g\ l;$
$i \le n; f\ i \ne g\ i]\!] \Longrightarrow$
$f\ (LEAST\ k.\ (f\ k \ne\ g\ k)) \ne g\ (LEAST\ k.\ (f\ k \ne\ g\ k))$
$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-LeastTr2-4*: $[\![\forall l \le (n::nat).\ G \gg f\ l; \forall l \le n.\ G \gg g\ l;$
$i \le n; f\ i \ne g\ i]\!] \Longrightarrow (LEAST\ k.\ (f\ k \ne g\ k)) \le n$
$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-LeastTr2-5*: $[\![\forall l \le (n::nat).\ G \gg f\ l; \forall l \le n.\ G \gg g\ l;$
$\exists i \le n.\ (f\ i \ne g\ i)]\!] \Longrightarrow$
$f\ (LEAST\ k.\ (f\ k \ne g\ k)) \ne g\ ((LEAST\ k.\ f\ k \ne g\ k))$
$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-LeastTr2-6*: $[\![\forall l \le (n::nat).\ G \gg f\ l; \forall l \le n.\ G \gg g\ l;$
$\exists i \le n.\ (f\ i \ne g\ i)]\!] \Longrightarrow (LEAST\ k.\ (f\ k \ne g\ k)) \le n$
$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-Least-sym*: $[\![\forall l \le (n::nat).\ G \gg f\ l; \forall l \le n.\ G \gg g\ l;$
$\exists i \le n.\ (f\ i \ne g\ i)]\!] \Longrightarrow$
$(LEAST\ k.\ (f\ k \ne g\ k)) = (LEAST\ k.\ (g\ k \ne f\ k))$
$\langle proof \rangle$

lemma *Nfunc-iNJTr*: $[\![inj\text{-}on\ g\ \{i.\ i \le (n::nat)\}; i \le n; j \le n; i < j\ ]\!] \Longrightarrow g\ i \ne$
$g\ j$
$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-LeastTr2-7*: $[\![\forall l \le (n::nat).\ G \gg f\ l; \forall l \le n.\ G \gg g\ l;$
$inj\text{-}on\ g\ \{i.\ i \le n\}; \exists i \le n.\ (f\ i \ne g\ i);$
$f\ k = g\ (LEAST\ k.(f\ k \ne g\ k))]\!] \Longrightarrow (LEAST\ k.(f\ k \ne g\ k)) < k$
$\langle proof \rangle$

lemma (**in** *Group*) *Nfunc-LeastTr2-8*: $[\![\forall l \le n.\ G \gg f\ l; \forall l \le n.\ G \gg g\ l;$
$inj\text{-}on\ g\ \{i.\ i \le n\}; \exists i \le n.\ f\ i \ne g\ i; f\ `\{i.\ i \le n\} = g\ `\{i.\ i \le n\}]\!]$
$\Longrightarrow$
$\exists\ k \in (nset\ (Suc\ (LEAST\ i.\ (f\ i \ne g\ i)))\ n).\ f\ k = g\ (LEAST\ i.\ (f\ i \ne g\ i))$
$\langle proof \rangle$

lemma (**in** *Group*) *ex-redchainTr1*: $[\![d\text{-}gchain\ G\ n\ f;$
$D\text{-}gchain\ G\ (card\ (f\ `\ \{i.\ i \le n\}) - Suc\ 0)\ g;$
$g\ `\ \{i.\ i \le (card\ (f\ `\ \{i.\ i \le n\}) - Suc\ 0)\} = f\ `\ \{i.\ i \le n\}]\!] \Longrightarrow$
$g\ (card\ (f\ `\ \{i.\ i \le n\}) - Suc\ 0) = f\ n$
$\langle proof \rangle$

lemma (**in** *Group*) *ex-redchainTr1-1*:⟦*d-gchain G* (*n*::*nat*) *f*;
D-gchain G (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *Suc 0*) *g*;
g ‘ {*i*. *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *Suc 0*)} = *f* ‘ {*i*. *i* ≤ *n*}⟧ ⟹
g 0 = *f 0*
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr2*:*d-gchain G* (*Suc n*) *f*
⟹ *D-gchain G 0* (*constmap* {*0*::*nat*} {*f* (*Suc n*)})
⟨*proof*⟩

lemma (**in** *Group*) *last-mem-excluded*:⟦*d-gchain G* (*Suc n*) *f*; *f n* ≠ *f* (*Suc n*)⟧
⟹
f (*Suc n*) ∉ *f* ‘ {*i*. *i* ≤ *n*}
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr4*:⟦*d-gchain G* (*Suc n*) *f*; *f n* ≠ *f* (*Suc n*)⟧ ⟹
card (*f* ‘ {*i*. *i* ≤ (*Suc n*)}) = *Suc* (*card* (*f* ‘ {*i*. *i* ≤ *n*}))
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr5*:*d-gchain G n f* ⟹ *0* < *card* (*f* ‘ {*i*. *i*≤ *n*})
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr6*:∀ *f*. *d-gchain G n f* ⟶
(∃ *g*. *D-gchain G* (*card* (*f* ‘{*i*. *i* ≤ *n*}) − *1*) *g* ∧
(*g* ‘ {*i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *n*}) − *1*)} = *f* ‘{*i*. *i* ≤ *n*}))
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchain*:*d-gchain G n f* ⟹
(∃ *g*. *D-gchain G* (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *1*) *g* ∧
g ‘ {*i*. *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *n*})
⟨*proof*⟩

lemma (**in** *Group*) *const-W-cmpser*:*d-gchain G* (*Suc n*) *f* ⟹
W-cmpser G 0 (*constmap* {*0*::*nat*} {*f* (*Suc n*)})
⟨*proof*⟩

lemma (**in** *Group*) *ex-W-cmpserTr0m*:∀ *f*. *w-cmpser G m f* ⟶
(∃ *g*. (*W-cmpser G* (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*) *g* ∧
g ‘{*i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*)} = *f* ‘{*i*. *i* ≤ *m*}))
⟨*proof*⟩

lemma (**in** *Group*) *ex-W-cmpser*:*w-cmpser G m f* ⟹
∃ *g*. *W-cmpser G* (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*) *g* ∧
g ‘ {*i*. *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *m*}
⟨*proof*⟩

3.15 Existence of reduced chain and composition series

lemma (**in** *Group*) *ex-W-cmpserTr3m1*:$[\![$*tw-cmpser G* (*m*::*nat*) *f*;
W-cmpser G ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*) *g*;
g ‘ {*i*. *i* ≤ ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*)} = *f* ‘{*i*. *i* ≤ *m*}$]\!]$ ⟹
tW-cmpser G ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*) *g*
⟨*proof*⟩

lemma (**in** *Group*) *ex-W-cmpserTr3m*:*tw-cmpser G m f* ⟹
∃ *g*. *tW-cmpser G* ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*) *g* ∧
g ‘{ *i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *m*}
⟨*proof*⟩

definition
red-ch-cd :: [- , *nat* ⇒ ′*a set*, *nat*, *nat* ⇒ ′*a set*] ⇒ *bool* **where**
red-ch-cd G f m g ⟷ *tW-cmpser G* (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*) *g* ∧
(*g* ‘{*i* . *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*)} = *f*‘ {*i*. *i* ≤ *m*})

definition
red-chain :: [- , *nat*, *nat* ⇒ ′*a set*] ⇒ (*nat* ⇒ ′*a set*) **where**
red-chain G m f = (*SOME g*. *g* ∈ {*h*. *red-ch-cd G f m h*})

lemma (**in** *Group*) *red-chainTr0m1-1*:*tw-cmpser G m f* ⟹
(*SOME g*. *g* ∈ {*h*. *red-ch-cd G f m h*}) ∈ {*h*. *red-ch-cd G f m h*}
⟨*proof*⟩

lemma (**in** *Group*) *red-chain-m*:*tw-cmpser G m f* ⟹
tW-cmpser G (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*) (*red-chain G m f*) ∧
(*red-chain G m f*) ‘{*i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *m*}
⟨*proof*⟩

3.16 Chain of groups II

definition
Gchain :: [*nat*, *nat* ⇒ ((′*a set*), ′*more*) *Group-scheme*] ⇒ *bool* **where**
Gchain n g ⟷ (∀ *l* ≤ *n*. *Group* (*g l*))

definition
isom-Gchains :: [*nat*, *nat* ⇒ *nat*, *nat* ⇒ ((′*a set*), ′*more*) *Group-scheme*,
nat ⇒ ((′*a set*), ′*more*) *Group-scheme*] ⇒ *bool* **where**
isom-Gchains n f g h ⟷ (∀ *i* ≤ *n*. (*g i*) ≅ (*h* (*f i*)))

definition
Gch-bridge :: [*nat*, *nat* ⇒ ((′*a set*), ′*more*) *Group-scheme*, *nat* ⇒
((′*a set*), ′*more*) *Group-scheme*, *nat* ⇒ *nat*] ⇒ *bool* **where**
Gch-bridge n g h f ⟷ (∀ *l* ≤ *n*. *f l* ≤ *n*) ∧ *inj-on f* {*i*. *i* ≤ *n*} ∧

$isom\text{-}Gchains\ n\ f\ g\ h$

lemma $Gchain\text{-}pre$: $Gchain\ (Suc\ n)\ g \Longrightarrow Gchain\ n\ g$
⟨proof⟩

lemma (**in** $Group$) $isom\text{-}unit$: $\llbracket G \gg H;\ G \gg K;\ H = \{\mathbf{1}\}\rrbracket \Longrightarrow Gp\ G\ H \cong Gp\ G\ K \longrightarrow K = \{\mathbf{1}\}$
⟨proof⟩

lemma $isom\text{-}gch\text{-}unitsTr4$: $\llbracket Group\ F;\ Group\ G;\ Ugp\ E;\ F \cong G;\ F \cong E\rrbracket \Longrightarrow G \cong E$
⟨proof⟩

lemma $isom\text{-}gch\text{-}cmp$: $\llbracket Gchain\ n\ g;\ Gchain\ n\ h;\ f1 \in \{i.\ i \le n\} \to \{i.\ i \le n\};\ f2 \in \{i.\ i \le n\} \to \{i.\ i \le n\};\ isom\text{-}Gchains\ n\ (cmp\ f2\ f1)\ g\ h\rrbracket \Longrightarrow isom\text{-}Gchains\ n\ f1\ g\ (cmp\ h\ f2)$
⟨proof⟩

lemma $isom\text{-}gch\text{-}transp$: $\llbracket Gchain\ n\ f;\ i \le n;\ j \le n;\ i < j\rrbracket \Longrightarrow isom\text{-}Gchains\ n\ (transpos\ i\ j)\ f\ (cmp\ \ f\ (transpos\ i\ j))$
⟨proof⟩

lemma $isom\text{-}gch\text{-}units\text{-}transpTr0$: $\llbracket Ugp\ E;\ Gchain\ n\ g;\ Gchain\ n\ h;\ i \le n;\ j \le n;\ i < j;\ isom\text{-}Gchains\ n\ (transpos\ i\ j)\ g\ h\rrbracket \Longrightarrow \{i.\ i \le n \land g\ i \cong E\} - \{i, j\} = \{i.\ i \le n \land h\ i \cong E\} - \{i, j\}$
⟨proof⟩

lemma $isom\text{-}gch\text{-}units\text{-}transpTr1$: $\llbracket Ugp\ E;\ Gchain\ n\ g;\ i \le n;\ j \le n;\ g\ j \cong E;\ i \ne j\rrbracket \Longrightarrow insert\ j\ (\{i.\ i \le n \land g\ i \cong E\} - \{i, j\}) = \{i.\ i \le n \land g\ i \cong E\} - \{i\}$
⟨proof⟩

lemma $isom\text{-}gch\text{-}units\text{-}transpTr2$: $\llbracket Ugp\ E;\ Gchain\ n\ g;\ i \le n;\ j \le n;\ i < j;\ g\ i \cong E\rrbracket \Longrightarrow \{i.\ i \le n \land g\ i \cong E\} = insert\ i\ (\{i.\ i \le n \land\ \ g\ i \cong E\} - \{i\})$
⟨proof⟩

lemma $isom\text{-}gch\text{-}units\text{-}transpTr3$: $\llbracket Ugp\ E;\ Gchain\ n\ g;\ i \le n\rrbracket \Longrightarrow finite\ (\{i.\ i \le n \land g\ i \cong E\} - \{i\})$
⟨proof⟩

lemma $isom\text{-}gch\text{-}units\text{-}transpTr4$: $\llbracket Ugp\ E;\ Gchain\ n\ g;\ i \le n\rrbracket \Longrightarrow finite\ (\{i.\ i \le n \land g\ i \cong E\} - \{i, j\})$
⟨proof⟩

lemma $isom\text{-}gch\text{-}units\text{-}transpTr5\text{-}1$: $\llbracket Ugp\ E;\ Gchain\ n\ g;\ Gchain\ n\ h;\ i \le (n::nat);\ j \le n;\ i < j;\ isom\text{-}Gchains\ n\ (transpos\ i\ j)\ g\ h\rrbracket \Longrightarrow g\ i \cong h\ j$
⟨proof⟩

lemma *isom-gch-units-transpTr5-2*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; $i \leq n$;
$j \leq n$; $i < j$; *isom-Gchains n* (*transpos i j*) *g h*⟧ ⟹ *g j* ≅ *h i*
⟨*proof*⟩

lemma *isom-gch-units-transpTr6*:⟦*Gchain n g*; $i \leq n$⟧ ⟹ *Group* (*g i*)
⟨*proof*⟩

lemma *isom-gch-units-transpTr7*:⟦*Ugp E*; $i \leq n$; $j \leq n$; *g j* ≅ *h i*;
Group (*h i*); *Group* (*g j*); ¬ *g j* ≅ *E*⟧ ⟹ ¬ *h i* ≅ *E*
⟨*proof*⟩

lemma *isom-gch-units-transpTr8-1*:⟦*Ugp E*; *Gchain n g*; $i \leq n$; $j \leq n$;
g i ≅ *E*; ¬ *g j* ≅ *E*⟧ ⟹
{*i*. $i \leq n$ ∧ *g i* ≅ *E*} = {*i*. $i \leq n$ ∧ *g i* ≅ *E*} − { *j* }
⟨*proof*⟩

lemma *isom-gch-units-transpTr8-2*:⟦*Ugp E*; *Gchain n g*; $i \leq n$; $j \leq n$;
¬ *g i* ≅ *E*; ¬ *g j* ≅ *E*⟧ ⟹
{*i*. $i \leq n$ ∧ *g i* ≅ *E*} = {*i*. $i \leq n$ ∧ *g i* ≅ *E*} − {*i*, *j* }
⟨*proof*⟩

lemma *isom-gch-units-transp*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; $i \leq n$; $j \leq n$;
$i < j$; *isom-Gchains n* (*transpos i j*) *g h*⟧ ⟹
card {*i*. $i \leq n$ ∧ *g i* ≅ *E*} = *card* {*i*. $i \leq n$ ∧ *h i* ≅ *E*}
⟨*proof*⟩

lemma *TR-isom-gch-units*:⟦*Ugp E*; *Gchain n f*; $i \leq n$; $j \leq n$; $i < j$⟧ ⟹
card {*k*. $k \leq n$ ∧ *f k* ≅ *E*} = *card* {*k*. $k \leq n$ ∧
(*cmp f* (*transpos i j*)) *k* ≅ *E*}
⟨*proof*⟩

lemma *TR-isom-gch-units-1*:⟦*Ugp E*; *Gchain n f*; $i \leq n$; $j \leq n$; $i < j$⟧ ⟹
card {*k*. $k \leq n$ ∧ *f k* ≅ *E*} = *card* {*k*. $k \leq n$ ∧ *f* (*transpos i j k*) ≅ *E*}
⟨*proof*⟩

lemma *isom-tgch-unitsTr0-1*:⟦*Ugp E*; *Gchain* (*Suc n*) *g*; *g* (*Suc n*) ≅ *E*⟧ ⟹
{*i*. *i* ≤ (*Suc n*) ∧ *g i* ≅ *E*} = *insert* (*Suc n*) {*i*. $i \leq n$ ∧ *g i* ≅ *E*}
⟨*proof*⟩

lemma *isom-tgch-unitsTr0-2*:*Ugp E* ⟹ *finite* ({*i*. *i* ≤ (*n*::*nat*) ∧ *g i* ≅ *E*})
⟨*proof*⟩

lemma *isom-tgch-unitsTr0-3*:⟦*Ugp E*; *Gchain* (*Suc n*) *g*; ¬ *g* (*Suc n*) ≅ *E*⟧
⟹ {*i*. *i* ≤ (*Suc n*) ∧ *g i* ≅ *E*} = {*i*. $i \leq n$ ∧ *g i* ≅ *E*}
⟨*proof*⟩

lemma *isom-tgch-unitsTr0*:⟦*Ugp E*;
card {*i*. $i \leq n$ ∧ *g i* ≅ *E*} = *card* {*i*. $i \leq n$ ∧ *h i* ≅ *E*};
Gchain (*Suc n*) *g* ∧ *Gchain* (*Suc n*) *h* ∧ *Gch-bridge* (*Suc n*) *g h f*;

$f\ (Suc\ n) = Suc\ n]\!] \Longrightarrow$
$card\ \{i.\ i \leq (Suc\ n) \wedge g\ i \cong E\} =$
$card\ \{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\}$
⟨proof⟩

lemma *isom-gch-unitsTr1-1*: $[\![Ugp\ E;\ Gchain\ (Suc\ n)\ g \wedge\ Gchain\ (Suc\ n)\ h$
$\wedge\ Gch\text{-}bridge\ (Suc\ n)\ g\ h\ f;\ f\ (Suc\ n) = Suc\ n]\!] \Longrightarrow$
$Gchain\ n\ g \wedge Gchain\ n\ h \wedge Gch\text{-}bridge\ n\ g\ h\ f$
⟨proof⟩

lemma *isom-gch-unitsTr1-2*: $[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;\ inj\text{-}on\ f\ \{i.\ i{\leq}(Suc\ n)\};$
$\forall\, l \leq (Suc\ n).\ f\ l \leq (Suc\ n)]\!] \Longrightarrow$
$(cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)\ (Suc\ n) = Suc\ n$
⟨proof⟩

lemma *isom-gch-unitsTr1-3*: $[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;$
$\forall\, l \leq (Suc\ n).\ f\ l \leq (Suc\ n);\ inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\}]\!] \Longrightarrow$
$inj\text{-}on\ (cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)\ \{i.\ i \leq (Suc\ n)\}$
⟨proof⟩

lemma *isom-gch-unitsTr1-4*: $[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;\ inj\text{-}on\ f\ \{i.\ i{\leq}(Suc\ n)\};$
$\forall\, l \leq (Suc\ n).\ f\ l \leq (Suc\ n)]\!] \Longrightarrow$
$inj\text{-}on\ (cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)\ \{i.\ i \leq n\}$
⟨proof⟩

lemma *isom-gch-unitsTr1-5*: $[\![Ugp\ E;\ Gchain\ (Suc\ n)\ g \wedge Gchain\ (Suc\ n)\ h\ \wedge$
$Gch\text{-}bridge\ (Suc\ n)\ g\ h\ f;\ f\ (Suc\ n) \neq Suc\ n\ ]\!] \Longrightarrow$
$Gchain\ n\ g \wedge Gchain\ n\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n)))\ \wedge$
$Gch\text{-}bridge\ n\ g\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n)))$
$(cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)$
⟨proof⟩

lemma *isom-gch-unitsTr1-6*: $[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;\ Gchain\ (Suc\ n)\ g\ \wedge$
$Gchain\ (Suc\ n)\ h \wedge Gch\text{-}bridge\ (Suc\ n)\ g\ h\ f]\!]\ \Longrightarrow Gchain\ (Suc\ n)\ g\ \wedge$
$Gchain\ (Suc\ n)\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n)))\ \wedge$
$Gch\text{-}bridge\ (Suc\ n)\ g\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n)))$
$(cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)$
⟨proof⟩

lemma *isom-gch-unitsTr1-7-0*: $[\![Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;\ k \leq (Suc\ n)]\!]$
$\Longrightarrow Gchain\ (Suc\ n)\ (cmp\ h\ (transpos\ k\ (Suc\ n)))$
⟨proof⟩

lemma *isom-gch-unitsTr1-7-1*: $[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;\ k \leq (Suc$
$n)]\!]$
$\Longrightarrow \{i.\ i \leq (Suc\ n) \wedge cmp\ h\ (transpos\ k\ (Suc\ n))\ i \cong E\} - \{k\ ,\ Suc\ n\} =$
$\{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\} - \{k\ ,\ Suc\ n\}$
⟨proof⟩

lemma *isom-gch-unitsTr1-7-2*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ \ h\ (Suc\ n) \cong E]\!] \Longrightarrow$
$cmp\ h\ (transpos\ k\ (Suc\ n))\ k \cong E$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-3*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ h\ k \cong E]\!] \Longrightarrow cmp\ h\ (transpos\ k\ (Suc\ n))\ (Suc\ n) \cong E$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-4*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ \neg\ h\ (Suc\ n) \cong E]\!] \Longrightarrow$
$\neg\ \ cmp\ h\ (transpos\ k\ (Suc\ n))\ k \cong E$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-5*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ \neg\ h\ k \cong E]\!] \Longrightarrow$
$\neg\ \ cmp\ h\ (transpos\ k\ (Suc\ n))\ (Suc\ n) \cong E$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-6*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ h\ (Suc\ n) \cong E;\ h\ k \cong E]\!] \Longrightarrow$
$\{i.\ i \leq (Suc\ n) \wedge\ \ h\ i \cong E\} =$
$insert\ k\ (insert\ (Suc\ n)\ (\{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\} - \{k,\ Suc\ n\}))$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-7*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ h\ (Suc\ n) \cong E;\ \neg\ h\ k \cong E]\!] \Longrightarrow$
$\{i.\ i \leq (Suc\ n) \wedge\ \ h\ i \cong E\} =$
$insert\ (Suc\ n)\ (\{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\} - \{k,\ Suc\ n\})$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-8*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ \neg\ h\ (Suc\ n) \cong E;\ \ h\ k \cong E]\!] \Longrightarrow$
$\{i.\ i \leq (Suc\ n) \wedge\ \ h\ i \cong E\} =$
$insert\ k\ (\{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\} - \{k,\ Suc\ n\})$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-9*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n);\ \neg\ h\ (Suc\ n) \cong E;\ \neg\ h\ k \cong E]\!] \Longrightarrow$
$\{i.\ i \leq (Suc\ n) \wedge\ \ h\ i \cong E\} =$
$\{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\} - \{k,\ Suc\ n\}$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;$
$k \leq (Suc\ n)]\!] \Longrightarrow card\ \{i.\ i \leq (Suc\ n) \wedge$
$cmp\ h\ (transpos\ k\ (Suc\ n))\ i \cong E\} =\ \ card\ \{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\}$
⟨*proof*⟩

lemma *isom-gch-unitsTr1*:$Ugp\ E \Longrightarrow \forall g.\ \forall h.\ \forall f.\ \ Gchain\ n\ g\ \wedge$

$Gchain\ n\ h \wedge Gch\text{-}bridge\ n\ g\ h\ f \longrightarrow card\ \{i.\ i \leq n \wedge g\ i \cong E\} = card\ \{i.\ i \leq n \wedge h\ i \cong E\}$
⟨proof⟩

lemma *isom-gch-units*: $[\![Ugp\ E;\ Gchain\ n\ g;\ Gchain\ n\ h;\ Gch\text{-}bridge\ n\ g\ h\ f]\!] \Longrightarrow$

$card\ \{i.\ i \leq n \wedge g\ i \cong E\} = card\ \{i.\ i \leq n \wedge h\ i \cong E\}$
⟨proof⟩

lemma *isom-gch-units-1*: $[\![Ugp\ E;\ Gchain\ n\ g;\ Gchain\ n\ h;\ \exists f.\ Gch\text{-}bridge\ n\ g\ h\ f]\!]$
$\Longrightarrow card\ \{i.\ i \leq n \wedge g\ i \cong E\} = card\ \{i.\ i \leq n \wedge h\ i \cong E\}$
⟨proof⟩

3.17 Jordan Hoelder theorem

3.17.1 *Rfn-tools*. Tools to treat refinement of a cmpser, rtos.

lemma *rfn-tool1*: $[\![\ 0 < (r::nat);\ (k::nat) = i * r + j;\ j < r\ ]\!] \Longrightarrow (k\ div\ r) = i$
⟨proof⟩

lemma *pos-mult-pos*: $[\![\ 0 < (r::nat);\ 0 < s]\!] \Longrightarrow 0 < r * s$
⟨proof⟩

lemma *rfn-tool1-1*: $[\![\ 0 < (r::nat);\ j < r\ ]\!] \Longrightarrow (i * r + j)\ div\ r = i$
⟨proof⟩

lemma *rfn-tool2*: $(a::nat) < s \Longrightarrow a \leq s - Suc\ 0$
⟨proof⟩

lemma *rfn-tool3*: $(0::nat) \leq m \Longrightarrow (m + n) - n = m$
⟨proof⟩

lemma *rfn-tool11*: $[\![0 < b;\ (a::nat) \leq b - Suc\ 0]\!] \Longrightarrow a < b$
⟨proof⟩

lemma *rfn-tool12*: $[\![0 < (s::nat);\ (i::nat)\ mod\ s = s - 1\ ]\!] \Longrightarrow Suc\ (i\ div\ s) = (Suc\ i)\ div\ s$
⟨proof⟩

lemma *rfn-tool12-1*: $[\![0 < (s::nat);\ (l::nat)\ mod\ s < s - 1\ ]\!] \Longrightarrow Suc\ (l\ mod\ s) = (Suc\ l)\ mod\ s$
⟨proof⟩

lemma *rfn-tool12-2*: $[\![0 < (s::nat);\ (i::nat)\ mod\ s = s - Suc\ 0]\!] \Longrightarrow (Suc\ i)\ mod\ s = 0$
⟨proof⟩

lemma *rfn-tool13*:⟦ $(0::nat) < r$; $a = b$ ⟧ $\Longrightarrow$ a *mod* $r = b$ *mod* r
⟨*proof*⟩

lemma *rfn-tool13-1*:⟦ $(0::nat) < r$; $a = b$ ⟧ $\Longrightarrow$ a *div* $r = b$ *div* r
⟨*proof*⟩

lemma *div-Tr1*:⟦ $(0::nat) < r$; $0 < s$; $l \leq s * r$⟧ $\Longrightarrow$ l *div* $s \leq r$
⟨*proof*⟩

lemma *div-Tr2*:⟦$(0::nat) < r$; $0 < s$; $l < s * r$⟧ $\Longrightarrow$ l *div* $s \leq r - Suc\ 0$
⟨*proof*⟩

lemma *div-Tr3*:⟦$(0::nat) < r$; $0 < s$; $l < s * r$⟧ $\Longrightarrow$ $Suc\ (l$ *div* $s) \leq r$
⟨*proof*⟩

lemma *div-Tr3-1*:⟦$(0::nat) < r$; $0 < s$; l *mod* $s = s - 1$⟧ $\Longrightarrow$ $Suc\ l$ *div* $s = Suc$
$(l$ *div* $s)$
⟨*proof*⟩

lemma *div-Tr3-2*:⟦$(0::nat) < r$; $0 < s$; l *mod* $s < s - 1$⟧ $\Longrightarrow$
l *div* $s = Suc\ l$ *div* s
⟨*proof*⟩

lemma *mod-div-injTr*:⟦$(0::nat) < r$; x *mod* $r = y$ *mod* r; x *div* $r = y$ *div* r⟧
$\Longrightarrow$ $x = y$
⟨*proof*⟩

definition
rtos :: [*nat*, *nat*] ⇒ (*nat* ⇒ *nat*) **where**
rtos r s i = (*if* $i < r * s$ *then* $(i$ *mod* $s) * r + i$ *div* s *else* $r * s$)

lemma *rtos-hom0*:⟦$(0::nat) < r$; $(0::nat) < s$; $i \leq (r * s - Suc\ 0)$⟧ $\Longrightarrow$
i *div* $s < r$
⟨*proof*⟩

lemma *rtos-hom1*:⟦$(0::nat) < r$; $0 < s$; $l \leq (r * s - Suc\ 0)$⟧ $\Longrightarrow$
(*rtos r s*) $l \leq (s * r - Suc\ 0)$
⟨*proof*⟩

lemma *rtos-hom2*:⟦$(0::nat) < r$; $(0::nat) < s$; $l \leq (r * s - Suc\ 0)$⟧ $\Longrightarrow$
rtos r s $l \leq (r * s - Suc\ 0)$
⟨*proof*⟩

lemma *rtos-hom3*:⟦$(0::nat) < r$; $0 < s$; $i \leq (r * s - Suc\ 0)$ ⟧ $\Longrightarrow$
(*rtos r s i div r*) = i *mod* s
⟨*proof*⟩

lemma *rtos-hom3-1*:⟦$(0::nat) < r$; $(0::nat) < s$; $i \leq (r * s - Suc\ 0)$ ⟧ $\Longrightarrow$

$(rtos\ r\ s\ i\ mod\ \ r) = i\ div\ s$
⟨*proof*⟩

lemma *rtos-hom5*:$[\![(0::nat) < r;\ (0::nat) < s;\ i \le (r *s - Suc\ 0);$
$i\ div\ s = r - Suc\ 0\ ]\!] \Longrightarrow Suc\ (rtos\ r\ s\ i)\ div\ r = Suc\ (i\ mod\ s)$
⟨*proof*⟩

lemma *rtos-hom7*:$[\![(0::nat) < r;\ (0::nat) < s;\ i \le (r * s - Suc\ 0);$
$i\ div\ s = r - Suc\ 0\ ]\!] \Longrightarrow Suc\ (rtos\ r\ s\ i)\ mod\ r = 0$
⟨*proof*⟩

lemma *rtos-inj*:$[\![\ (0::nat) < r;\ (0::nat) < s\ ]\!] \Longrightarrow$
$inj\text{-}on\ (rtos\ r\ s)\ \{i.\ i \le (r * s - Suc\ 0)\}$
⟨*proof*⟩

lemma *rtos-rs-Tr1*:$[\![(0::nat) < r;\ 0 < s\ ]\!] \Longrightarrow rtos\ r\ s\ (r * s) = r * s$
⟨*proof*⟩

lemma *rtos-rs-Tr2*:$[\![(0::nat) < r;\ 0 < s\ ]\!] \Longrightarrow$
$\forall\, l \le (r * s).\ rtos\ r\ s\ l \le (r * s)$
⟨*proof*⟩

lemma *rtos-rs-Tr3*:$[\![(0::nat) < r;\ 0 < s\ ]\!] \Longrightarrow$
$inj\text{-}on\ (rtos\ r\ s)\ \{i.\ i \le (r * s)\}$
⟨*proof*⟩

lemma *Qw-cmpser*:$[\![Group\ G;\ w\text{-}cmpser\ G\ (Suc\ n)\ f\ ]\!] \Longrightarrow$
$Gchain\ n\ (Qw\text{-}cmpser\ G\ f)$
⟨*proof*⟩

definition
$wcsr\text{-}rfns :: [\text{-}\ ,\ nat,\ nat \Rightarrow 'a\ set,\ nat] \Rightarrow (nat \Rightarrow 'a\ set)\ set$ **where**
$wcsr\text{-}rfns\ G\ r\ f\ s = \{h.\ tw\text{-}cmpser\ G\ (s * r)\ h\ \wedge$
$(\forall\, i \le r.\ h\ (i * s) = f\ i)\}$

definition
$trivial\text{-}rfn :: [\text{-}\ ,\ nat,\ nat \Rightarrow 'a\ set,\ nat] \Rightarrow (nat \Rightarrow 'a\ set)$ **where**
$trivial\text{-}rfn\ G\ r\ f\ s\ k == if\ k < (s * r)\ then\ f\ (k\ div\ s)\ else\ f\ r$

lemma (**in** *Group*) *rfn-tool8*:$[\![compseries\ G\ r\ f;\ 0 < r]\!] \Longrightarrow d\text{-}gchain\ G\ r\ f$
⟨*proof*⟩

lemma (**in** *Group*) *rfn-tool16*:$[\![0 < r;\ 0 < s;\ i \le (s * r - Suc\ 0);$
$G \ggg f\ (i\ div\ s);\ (Gp\ G\ (f\ (i\ div\ s))) \triangleright f\ (Suc\ (i\ div\ s));$
$(Gp\ G\ (f\ (i\ div\ s))) \ggg (f\ (i\ div\ s) \cap g\ (s - Suc\ 0))]\!] \Longrightarrow$
$(Gp\ G\ ((f\ (Suc\ (i\ div\ s))\ \diamond_G\ (f\ (i\ div\ s) \cap g\ (s - Suc\ 0))))) \triangleright$
$(f\ (Suc\ (i\ div\ s)))$

⟨*proof*⟩

Show existence of the trivial refinement. This is not necessary to prove JHS

lemma *rfn-tool30*:⟦*0* < *r*; *0* < *s*; *l div s* ∗ *s* + *s* < *s* ∗ *r*⟧
⟹ *Suc* (*l div s*) < *r*
⟨*proof*⟩

lemma (**in** *Group*) *simple-grouptr0*:⟦*G* ≫ *H*; *G* ▷ *K*; *K* ⊆ *H*; *simple-Group* (*G* / *K*)⟧
⟹ *H* = *carrier G* ∨ *H* = *K*
⟨*proof*⟩

lemma (**in** *Group*) *compser-nsg*:⟦*0* < *n*; *compseries G n f*; *i* ≤ (*n* − *1*)⟧
⟹ *Gp G* (*f i*) ▷ (*f* (*Suc i*))
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr5*:⟦*0* < *n*; *compseries G n f*; *i* ≤ (*n* − *Suc 0*)⟧
⟹ (*f* (*Suc i*)) ⊆ (*f i*)
⟨*proof*⟩

lemma (**in** *Group*) *refine-cmpserTr0*:⟦*0* < *n*; *compseries G n f*; *i* ≤ (*n* − *1*);
G ≫ *H*; *f* (*Suc i*) ⊆ *H* ∧ *H* ⊆ *f i*⟧ ⟹ *H* = *f* (*Suc i*) ∨ *H* = *f i*
⟨*proof*⟩

lemma *div-Tr4*:⟦ (*0*::*nat*) < *r*; *0* < *s*; *j* < *s* ∗ *r* ⟧ ⟹ *j div s* ∗ *s* + *s* ≤ *r* ∗ *s*
⟨*proof*⟩

lemma (**in** *Group*) *compseries-is-tW-cmpser*:⟦*0* < *r*; *compseries G r f*⟧ ⟹
tW-cmpser G r f
⟨*proof*⟩

lemma (**in** *Group*) *compseries-is-td-gchain*:⟦*0* < *r*; *compseries G r f*⟧ ⟹
td-gchain G r f
⟨*proof*⟩

lemma (**in** *Group*) *compseries-is-D-gchain*:⟦*0* < *r*; *compseries G r f*⟧ ⟹
D-gchain G r f
⟨*proof*⟩

lemma *divTr5*:⟦*0* < *r*; *0* < *s*; *l* < (*r* ∗ *s*)⟧ ⟹
l div s ∗ *s* ≤ *l* ∧ *l* ≤ (*Suc* (*l div s*)) ∗ *s*
⟨*proof*⟩

lemma (**in** *Group*) *rfn-compseries-iMTr1*:⟦*0* < *r*; *0* < *s*; *compseries G r f*;
h ∈ *wcsr-rfns G r f s*⟧ ⟹ *f* ' {*i*. *i* ≤ *r*} ⊆ *h* ' {*i*. *i* ≤ (*s* ∗ *r*)}
⟨*proof*⟩

lemma *rfn-compseries-iMTr2*:⟦*0* < *r*; *0* < *s*; *xa* < *s* ∗ *r* ⟧ ⟹

$xa\ div\ s * s \leq r * s \wedge Suc\ (xa\ div\ s) * s \leq r * s$
⟨*proof*⟩

lemma (**in** *Group*) *rfn-compseries-iMTr3*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ j \leq r;\ \forall i \leq r.\ h\ (i * s) = f\ i]\!] \Longrightarrow h\ (j * s) = f\ j$
⟨*proof*⟩

lemma (**in** *Group*) *rfn-compseries-iM*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ h \in wcsr\text{-}rfns\ G\ r\ f\ s]\!] \Longrightarrow card\ (h\ `\{i.\ i \leq (s * r)\}) = r + 1$
⟨*proof*⟩

definition
$cmp\text{-}rfn :: [\text{-}\ ,\ nat,\ nat \Rightarrow 'a\ set,\ nat,\ nat \Rightarrow 'a\ set] \Rightarrow (nat \Rightarrow 'a\ set)$ **where**
$cmp\text{-}rfn\ G\ r\ f\ s\ g = (\lambda i.\ (if\ i < s * r\ then$
$f\ (Suc\ (i\ div\ s)) \diamond_G (f\ (i\ div\ s) \cap g\ (i\ mod\ s))\ else\ \{\mathbf{1}_G\}))$

lemma (**in** *Group*) *cmp-rfn0*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;\ i \leq (r - 1);\ j \leq (s - 1)]\!] \Longrightarrow G \gg f\ (Suc\ i) \diamond_G ((f\ i\ ) \cap (g\ j))$
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn1*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g]\!] \Longrightarrow f\ (Suc\ 0) \diamond_G ((f\ 0\ ) \cap (g\ 0)) = carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn2*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;\ l \leq (s * r)]\!] \Longrightarrow G \gg cmp\text{-}rfn\ G\ r\ f\ s\ g\ l$
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn3*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g]\!] \Longrightarrow cmp\text{-}rfn\ G\ r\ f\ s\ g\ 0 = carrier\ G \wedge cmp\text{-}rfn\ G\ r\ f\ s\ g\ (s * r) = \{\mathbf{1}\}$
⟨*proof*⟩

lemma *rfn-tool20*: $[\![(0::nat) < m;\ a = b * m + c;\ c < m\ ]\!] \Longrightarrow a\ mod\ m = c$
⟨*proof*⟩

lemma *Suci-mod-s-2*: $[\![0 < r;\ 0 < s;\ i \leq r * s - Suc\ 0;\ i\ mod\ s < s - Suc\ 0]\!] \Longrightarrow (Suc\ i)\ mod\ s = Suc\ (i\ mod\ s)$
⟨*proof*⟩

lemma (**in** *Group*) *inter-sgsTr1*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;\ i < r * s]\!] \Longrightarrow G \gg f\ (i\ div\ s) \cap g\ (s - Suc\ 0)$
⟨*proof*⟩

lemma (**in** *Group*) *JHS-Tr0-2*: $[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g]\!]$
$\Longrightarrow \forall i \leq (s * r - Suc\ 0).\ Gp\ G\ (cmp\text{-}rfn\ G\ r\ f\ s\ g\ i) \rhd cmp\text{-}rfn\ G\ r\ f\ s\ g\ (Suc\ i)$

⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn4*:$[\![0 < r; 0 < s; compseries\ G\ r\ f;$
$compseries\ G\ s\ g; l \leq (s * r - Suc\ 0)]\!] \Longrightarrow$
$cmp\text{-}rfn\ G\ r\ f\ s\ g\ (Suc\ l) \subseteq cmp\text{-}rfn\ G\ r\ f\ s\ g\ l$
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn5*:$[\![0 < r; 0 < s; compseries\ G\ r\ f; compseries\ G\ s\ g]\!]$
$\Longrightarrow \forall i \leq r.\ cmp\text{-}rfn\ G\ r\ f\ s\ g\ (i * s) = f\ i$
⟨*proof*⟩

lemma (**in** *Group*) *JHS-Tr0*:$[\![(0::nat) < r; 0 < s; compseries\ G\ r\ f;$
$compseries\ G\ s\ g]\!] \Longrightarrow cmp\text{-}rfn\ G\ r\ f\ s\ g \in wcsr\text{-}rfns\ G\ r\ f\ s$
⟨*proof*⟩

lemma *rfn-tool17*:$(a::nat) = b \Longrightarrow a - c = b - c$
⟨*proof*⟩

lemma *isom4b*:$[\![Group\ G; G \triangleright N; G \gg H]\!] \Longrightarrow$
$(Gp\ G\ (N \diamond_G H)\ /\ N) \cong (Gp\ G\ H\ /\ (H \cap N))$
⟨*proof*⟩

lemma *Suc-rtos-div-r-1*:$[\![\ 0 < r; 0 < s; i \leq r * s - Suc\ 0;$
$Suc\ (rtos\ r\ s\ i) < r * s; i\ mod\ s = s - Suc\ 0;$
$i\ div\ s < r - Suc\ 0]\!] \Longrightarrow\ Suc\ (rtos\ r\ s\ i)\ div\ r = i\ mod\ s$
⟨*proof*⟩

lemma *Suc-rtos-mod-r-1*:$[\![0 < r; 0 < s; i \leq r * s - Suc\ 0; Suc\ (rtos\ r\ s\ i) < r$
$* s; i\ mod\ s = s - Suc\ 0; i\ \ div\ s < r - Suc\ 0]\!]$
$\Longrightarrow Suc\ (rtos\ r\ s\ i)\ mod\ r = Suc\ (i\ div\ s)$
⟨*proof*⟩

lemma *i-div-s-less*:$[\![0 < r; 0 < s; i \leq r * s - Suc\ 0; Suc\ (rtos\ r\ s\ i) < r * s;$
$i\ mod\ s = s - Suc\ 0; Suc\ i < s * r\ ]\!] \Longrightarrow i\ div\ s < r - Suc\ 0$
⟨*proof*⟩

lemma *rtos-mod-r-1*:$[\![\ 0 < r; 0 < s; i \leq r * s - Suc\ 0; rtos\ r\ s\ i < r * s;$
$i\ mod\ s = s - Suc\ 0\ ]\!] \Longrightarrow rtos\ r\ s\ i\ mod\ r = i\ div\ s$
⟨*proof*⟩

lemma *Suc-i-mod-s-0-1*:$[\![0 < r; 0 < s; i \leq r * s - Suc\ 0; i\ mod\ s = s - Suc\ 0]\!]$
$\Longrightarrow Suc\ i\ mod\ s = 0$
⟨*proof*⟩

lemma *Suci-div-s-2*:$[\![0 < r; 0 < s; i \leq r * s - Suc\ 0; i\ mod\ s < s - Suc\ 0]\!]$
$\Longrightarrow Suc\ i\ div\ s = i\ div\ s$
⟨*proof*⟩

lemma *rtos-i-mod-r-2*: $[\![0 < r;\ 0 < s;\ i \leq r * s - Suc\ 0]\!] \Longrightarrow rtos\ r\ s\ i\ mod\ r = i\ div\ s$
$\langle proof \rangle$

lemma *Suc-rtos-i-mod-r-2*: $[\![0 < r;\ 0 < s;\ i \leq r * s - Suc\ 0;$
$i\ div\ s = r - Suc\ 0]\!] \Longrightarrow Suc\ (rtos\ r\ s\ i)\ mod\ r = 0$
$\langle proof \rangle$

lemma *Suc-rtos-i-mod-r-3*: $[\![0 < r;\ 0 < s;\ i \leq r * s - Suc\ 0;$
$i\ div\ s < r - Suc\ 0]\!] \Longrightarrow Suc\ (rtos\ r\ s\ i)\ mod\ r = Suc\ (i\ div\ s)$
$\langle proof \rangle$

lemma *Suc-rtos-div-r-3*: $[\![0 < r;\ 0 < s;\ i\ mod\ s < s - Suc\ 0;\ i \leq r * s - Suc\ 0;$
$Suc\ (rtos\ r\ s\ i) < r * s;\ i\ div\ s < r - Suc\ 0]\!] \Longrightarrow$
$Suc\ (rtos\ r\ s\ i)\ div\ r = i\ mod\ s$
$\langle proof \rangle$

lemma *r-s-div-s*: $[\![0 < r;\ 0 < s]\!] \Longrightarrow (r * s - Suc\ 0)\ div\ s = r - Suc\ 0$
$\langle proof \rangle$

lemma *r-s-mod-s*: $[\![0 < r;\ 0 < s]\!] \Longrightarrow (r * s - Suc\ 0)\ mod\ s = s - Suc\ 0$
$\langle proof \rangle$

lemma *rtos-r-s*: $[\![0 < r;\ 0 < s]\!] \Longrightarrow rtos\ r\ s\ (r * s - Suc\ 0) = r * s - Suc\ 0$
$\langle proof \rangle$

lemma *rtos-rs-1*: $[\![\ 0 < r;\ 0 < s;\ rtos\ r\ s\ i < r * s;$
$\neg\ Suc\ (rtos\ r\ s\ i) < r * s]\!] \Longrightarrow rtos\ r\ s\ i = r * s - Suc\ 0$
$\langle proof \rangle$

lemma *rtos-rs-i-rs*: $[\![\ 0 < r;\ 0 < s;\ i \leq r * s - Suc\ 0;$
$rtos\ r\ s\ i = r * s - Suc\ 0]\!] \Longrightarrow\ i = r * s - Suc\ 0$
$\langle proof \rangle$

lemma *JHS-Tr1-1*: $[\![Group\ G;\ 0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g]\!] \Longrightarrow f\ (Suc\ ((r * s - Suc\ 0)\ div\ s)) \diamond_G (f\ ((r * s - Suc\ 0)\ div\ s) \cap g\ ((r * s - Suc\ 0)\ mod\ s)) = f\ (r - Suc\ 0) \cap g\ (s - Suc\ 0)$
$\langle proof \rangle$

lemma *JHS-Tr1-2*: $[\![Group\ G;\ 0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;$
$k < r - Suc\ 0]\!] \Longrightarrow ((Gp\ G\ (f\ (Suc\ k) \diamond_G (f\ k \cap g\ (s - Suc\ 0)))) /$
$(f\ (Suc\ (Suc\ k)) \diamond_G (f\ (Suc\ k) \cap g\ 0))) \cong$
$((Gp\ G\ (g\ s \diamond_G (g\ (s - Suc\ 0) \cap f\ k))) /$
$(g\ s \diamond_G (g\ (s - Suc\ 0) \cap f\ (Suc\ k))))$
$\langle proof \rangle$

lemma *JHS-Tr1-3*: $[\![Group\ G;\ 0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;$
$i \leq s * r - Suc\ 0;\ Suc\ (rtos\ r\ s\ i) < s * r;\ Suc\ i < s * r;$

$i\ mod\ s < s - Suc\ 0;\ Suc\ i\ div\ s \leq r - Suc\ 0;\ i\ div\ s = r - Suc\ 0]\!]$
$\Longrightarrow Group\ (Gp\ G\ (f\ r \diamond_G (f\ (r - Suc\ 0) \cap g\ (i\ mod\ s)))\ /$
$(f\ r \diamond_G (f\ (r - Suc\ 0) \cap g\ (Suc\ (i\ mod\ s)))))$
⟨*proof*⟩

lemma *JHS-Tr1-4*:$[\![Group\ G;\ 0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;$
$i \leq s * r - Suc\ 0;\ Suc\ (rtos\ r\ s\ i) < s * r;\ Suc\ i < s * r;$
$i\ mod\ s < s - Suc\ 0;\ Suc\ i\ div\ s \leq r - Suc\ 0;\ i\ div\ s = r - Suc\ 0]\!] \Longrightarrow$
$Group\ (Gp\ G\ (g\ (Suc\ (i\ mod\ s)) \diamond_G (g\ (i\ mod\ s) \cap f\ (r - Suc\ 0)))\ /$
$(g\ (Suc\ (Suc\ (i\ mod\ s))) \diamond_G (g\ (Suc\ (i\ mod\ s)) \cap f\ 0)))$
⟨*proof*⟩

lemma *JHS-Tr1-5*:$[\![Group\ G;\ 0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;$
$i \leq s * r - Suc\ 0;\ Suc\ (rtos\ r\ s\ i) < s * r;\ Suc\ i < s * r;$
$i\ mod\ s < s - Suc\ 0;\ i\ div\ s < r - Suc\ 0]\!]$
$\Longrightarrow (Gp\ G\ (f\ (Suc\ (i\ div\ s)) \diamond_G (f\ (i\ div\ s) \cap g\ (i\ mod\ s)))\ /$
$(f\ (Suc\ (i\ div\ s)) \diamond_G (f\ (i\ div\ s) \cap g\ (Suc\ (i\ mod\ s))))) \cong$
$(Gp\ G\ (g\ (Suc\ (i\ mod\ s)) \diamond_G (g\ (i\ mod\ s) \cap f\ (i\ div\ s)))\ /$
$(g\ (Suc\ (Suc\ (rtos\ r\ s\ i)\ div\ r)) \diamond_G$
$(g\ (Suc\ (rtos\ r\ s\ i)\ div\ r) \cap f\ (Suc\ (rtos\ r\ s\ i)\ mod\ r))))$
⟨*proof*⟩

lemma *JHS-Tr1-6*: $[\![Group\ G;\ 0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;$
$i \leq r * s - Suc\ 0;\ Suc\ (rtos\ r\ s\ i) < r * s]\!] \Longrightarrow$
$((Gp\ G\ (cmp\text{-}rfn\ G\ r\ f\ s\ g\ i))\ /\ (cmp\text{-}rfn\ G\ r\ f\ s\ g\ (Suc\ i))) \cong$
$((Gp\ G\ (g\ (Suc\ (rtos\ r\ s\ i\ div\ r)) \diamond_G$
$(g\ (rtos\ r\ s\ i\ div\ r) \cap f\ (rtos\ r\ s\ i\ mod\ r))))\ /$
$(g\ (Suc\ (Suc\ (rtos\ r\ s\ i)\ div\ r)) \diamond_G$
$(g\ (Suc\ (rtos\ r\ s\ i)\ div\ r) \cap f\ (Suc\ (rtos\ r\ s\ i)\ mod\ r))))$
⟨*proof*⟩

lemma *JHS-Tr1*:$[\![\ Group\ G;\ 0 < r;\ 0 < s;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g]\!]$
$\Longrightarrow isom\text{-}Gchains\ (r * s - 1)\ (rtos\ r\ s)\ (Qw\text{-}cmpser\ G\ (cmp\text{-}rfn\ G\ r\ f\ s\ g))$
$(Qw\text{-}cmpser\ G\ (cmp\text{-}rfn\ G\ s\ g\ r\ f))$
⟨*proof*⟩

lemma *abc-SucTr0*:$[\![(0::nat) < a;\ c \leq b;\ a - Suc\ 0 = b - c]\!] \Longrightarrow a = (Suc\ b)$
$- c$
⟨*proof*⟩

lemma *length-wcmpser0-0*:$[\![Group\ G;\ Ugp\ E;\ w\text{-}cmpser\ G\ (Suc\ 0)\ f]\!] \Longrightarrow$
$f\ `\ \{i.\ i \leq (Suc\ 0)\} = \{f\ 0,\ f\ (Suc\ 0)\}$
⟨*proof*⟩

lemma *length-wcmpser0-1*:$[\![Group\ G;\ Ugp\ E;\ w\text{-}cmpser\ G\ (Suc\ n)\ f;\ i \in \{i.\ i \leq$
$n\};$
$(Qw\text{-}cmpser\ G\ f)\ i \cong E]\!] \Longrightarrow f\ i = f\ (Suc\ i)$
⟨*proof*⟩

lemma *length-wcmpser0-2*:$[\![Group\ G;\ Ugp\ E;\ w\text{-}cmpser\ G\ (Suc\ n)\ f;\ i \le n;$
$\neg\ (Qw\text{-}cmpser\ G\ f)\ i \cong E]\!] \Longrightarrow f\ i \ne f\ (Suc\ i)$
⟨*proof*⟩

lemma *length-wcmpser0-3*:$[\![Group\ G;\ Ugp\ E;\ w\text{-}cmpser\ G\ (Suc\ (Suc\ n))\ f;$
$f\ (Suc\ n) \ne f\ (Suc\ (Suc\ n))]\!] \Longrightarrow f\ (Suc\ (Suc\ n)) \notin f\ `\ \{i.\ i \le (Suc\ n)\}$
⟨*proof*⟩

lemma *length-wcmpser0-4*:$[\![Group\ G;\ Ugp\ E;\ w\text{-}cmpser\ G\ (Suc\ 0)\ f]\!] \Longrightarrow$
$card\ (f\ `\ \{i.\ i \le Suc\ 0\}) - 1 = Suc\ 0 - card\ \{i.\ i = 0\ \wedge$
$Qw\text{-}cmpser\ G\ f\ i \cong E\}$

⟨*proof*⟩

lemma *length-wcmpser0-5*: $[\![Group\ G;\ Ugp\ E;\ w\text{-}cmpser\ G\ (Suc\ (Suc\ n))\ f;$
$w\text{-}cmpser\ G\ (Suc\ n)\ f;$
$card\ (f\ `\ \{i.\ i \le (Suc\ n)\}) - 1 = Suc\ n -$
$card\ \{i.\ i \le n\ \wedge\ Qw\text{-}cmpser\ G\ f\ i \cong E\};$
$Qw\text{-}cmpser\ G\ f\ (Suc\ n) \cong E]\!] \Longrightarrow$
$card\ (f\ `\ \{i\ .\ i \le (Suc\ (Suc\ n))\}) - 1 =$
$Suc\ (Suc\ n) - card\ \{i.\ i \le (Suc\ n) \wedge Qw\text{-}cmpser\ G\ f\ i \cong E\}$
⟨*proof*⟩

lemma *length-wcmpser0-6*:$[\![Group\ G;\ w\text{-}cmpser\ G\ (Suc\ (Suc\ n))\ f]\!] \Longrightarrow$
$0 < card\ (f\ `\ \{i.\ i \le (Suc\ n)\})$
⟨*proof*⟩

lemma *length-wcmpser0-7*:$[\![Group\ G;\ w\text{-}cmpser\ G\ (Suc\ (Suc\ n))\ f]\!] \Longrightarrow$
$card\ \{i.\ i \le n \wedge Qw\text{-}cmpser\ G\ f\ i \cong E\} \le Suc\ n$
⟨*proof*⟩

lemma *length-wcmpser0*:$[\![Group\ G;\ Ugp\ E]\!] \Longrightarrow \forall f.\ w\text{-}cmpser\ G\ (Suc\ n)\ f \longrightarrow$
$card\ (f\ `\ \{i.\ i \le (Suc\ n)\}) - 1 = (Suc\ \ n) - (card\ \{i.\ i \le n\ \wedge$
$((Qw\text{-}cmpser\ G\ f)\ i \cong E)\})$
⟨*proof*⟩

lemma *length-of-twcmpser*:$[\![Group\ G;\ Ugp\ E;\ tw\text{-}cmpser\ G\ (Suc\ n)\ f\ ]\!] \Longrightarrow$
$length\text{-}twcmpser\ G\ (Suc\ n)\ f =$
$(Suc\ n) - (card\ \{i.\ i \le n \wedge ((Qw\text{-}cmpser\ G\ f)\ i \cong E)\})$
⟨*proof*⟩

lemma *JHS-1*:$[\![Group\ G;\ Ugp\ E;\ compseries\ G\ r\ f;\ compseries\ G\ s\ g;\ 0{<}r;\ 0 <$
$s]\!]$
$\Longrightarrow r = \ r * s - card\ \{i.\ i \le (r * s - Suc\ 0)\ \wedge$
$Qw\text{-}cmpser\ G\ (cmp\text{-}rfn\ G\ r\ f\ s\ g)\ i \cong E\}$
⟨*proof*⟩

lemma *J-H-S*:⟦*Group G*; *Ugp E*; *compseries G r f*; *compseries G s g*; *0<r*;
  (*0*::*nat*)<*s* ⟧ ⟹ *r* = *s*
⟨*proof*⟩

end

theory *Algebra4*
imports *Algebra3*
begin

3.18 Abelian groups

record *'a aGroup* = *'a carrier* +
  pop :: [*'a*, *'a*] ⇒ *'a* (**infixl** $\pm_1$ *62*)
  mop :: *'a* ⇒ *'a* (($-_{a1}$ -) [*64*]*63*)
  zero :: *'a* (**0_1**)

locale *aGroup* =
  fixes *A* (**structure**)
 assumes
  pop-closed: *pop A* ∈ *carrier A* → *carrier A* → *carrier A*
 and *aassoc* : ⟦*a* ∈ *carrier A*; *b* ∈ *carrier A*; *c* ∈ *carrier A*⟧ ⟹
  (*a* ± *b*) ± *c* = *a* ± (*b* ± *c*)
 and *pop-commute*:⟦*a* ∈ *carrier A*; *b* ∈ *carrier A*⟧ ⟹ *a* ± *b* = *b* ± *a*
 and *mop-closed*:*mop A* ∈ *carrier A* → *carrier A*
 and *l-m* :*a* ∈ *carrier A* ⟹ ($-_a$ *a*) ± *a* = **0**
 and *ex-zero*: **0** ∈ *carrier A*
 and *l-zero*:*a* ∈ *carrier A* ⟹ **0** ± *a* = *a*

definition
  b-ag :: - ⇒
  (|*carrier*:: *'a set*, *top*:: [*'a*, *'a*] ⇒ *'a* , *iop*:: *'a* ⇒ *'a*, *one*:: *'a* |) **where**
  b-ag A = (|*carrier* = *carrier A*, *top* = *pop A*, *iop* = *mop A*, *one* = *zero A* |)

definition
  asubGroup :: [- , *'a set*] ⇒ *bool* **where**
  asubGroup A H ⟷ (*b-ag A*) » *H*

definition
  aqgrp :: [- , *'a set*] ⇒
   (| *carrier*::*'a set set*, *pop*::[*'a set*, *'a set*] ⇒ *'a set*,
    mop::*'a set* ⇒ *'a set*, *zero* :: *'a set* |) **where**
  aqgrp A H = (|*carrier* = *set-rcs* (*b-ag A*) *H*,
   pop = λ*X*. λ*Y*. (*c-top* (*b-ag A*) *H X Y*),
   mop = λ*X*. (*c-iop* (*b-ag A*) *H X*), *zero* = *H* |)

definition
  ag-idmap :: - ⇒ (*'a* ⇒ *'a*) ((*aI*$_-$)) **where**

$aI_A = (\lambda x{\in}carrier\ A.\ x)$

abbreviation
$ASubG :: [('a, 'more)\ aGroup\text{-}scheme, 'a\ set] => bool$ (**infixl** +> 58) **where**
$A +> H == asubGroup\ A\ H$

definition
$Ag\text{-}ind :: [\text{-} , 'a \Rightarrow 'd] \Rightarrow 'd\ aGroup$ **where**
$Ag\text{-}ind\ A\ f = (\!|carrier = f`(carrier\ A),$
$pop = \lambda x \in f`(carrier\ A).\ \lambda y \in f`(carrier\ A).$
$f(((invfun\ (carrier\ A)\ (f`(carrier\ A))\ f)\ x) \pm_A$
$((invfun\ (carrier\ A)\ (f`(carrier\ A))\ f)\ y)),$
$mop = \lambda x{\in}(f`(carrier\ A)).\ f\ (-_{a\,A}\ ((invfun\ (carrier\ A)\ (f`(carrier\ A))\ f)\ x)),$
$zero = f\ (\mathbf{0}_A)|\!)$

definition
$Agii :: [\text{-} , 'a \Rightarrow 'd] \Rightarrow ('a \Rightarrow 'd)$ **where**
$Agii\ A\ f = (\lambda x{\in}carrier\ A.\ f\ x)$

lemma (**in** *aGroup*) *ag-carrier-carrier*:$carrier\ (b\text{-}ag\ A) = carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-closed*:$[\![x \in carrier\ A;\ y \in carrier\ A]\!] \Longrightarrow$
$pop\ A\ x\ y \in carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-mOp-closed*:$x \in carrier\ A \Longrightarrow (-_a\ x) \in carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-subset*:$A +> H \Longrightarrow H \subseteq carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-commute*:$[\![x \in carrier\ A;\ y \in carrier\ A]\!] \Longrightarrow$
$pop\ A\ x\ y = pop\ A\ y\ x$
⟨*proof*⟩

lemma (**in** *aGroup*) *b-ag-group*:$Group\ (b\text{-}ag\ A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agop-gop*:$top\ (b\text{-}ag\ A) = pop\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *agiop-giop*:$iop\ (b\text{-}ag\ A) = mop\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *agunit-gone*:$one\ (b\text{-}ag\ A) = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-add-r*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;$

$a = b\rrbracket \Longrightarrow a \pm c = b \pm c$
⟨proof⟩

lemma (**in** *aGroup*) *ag-add-commute*: $\llbracket a \in carrier\ A;\ b \in carrier\ A\rrbracket \Longrightarrow a \pm b = b \pm a$
⟨proof⟩

lemma (**in** *aGroup*) *ag-pOp-add-l*: $\llbracket a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;\ a = b\rrbracket \Longrightarrow c \pm a = c \pm b$
⟨proof⟩

lemma (**in** *aGroup*) *asubg-pOp-closed*: $\llbracket asubGroup\ A\ H;\ x \in H;\ y \in H\rrbracket \Longrightarrow pop\ A\ x\ y \in H$
⟨proof⟩

lemma (**in** *aGroup*) *asubg-mOp-closed*: $\llbracket asubGroup\ A\ H;\ x \in H\rrbracket \Longrightarrow -_a\ x \in H$
⟨proof⟩

lemma (**in** *aGroup*) *asubg-subset1*: $\llbracket asubGroup\ A\ H;\ x \in H\rrbracket \Longrightarrow x \in carrier\ A$
⟨proof⟩

lemma (**in** *aGroup*) *asubg-inc-zero*: $asubGroup\ A\ H \Longrightarrow \mathbf{0} \in H$
⟨proof⟩

lemma (**in** *aGroup*) *ag-inc-zero*: $\mathbf{0} \in carrier\ A$
⟨proof⟩

lemma (**in** *aGroup*) *ag-l-zero*: $x \in carrier\ A \Longrightarrow \mathbf{0} \pm x = x$
⟨proof⟩

lemma (**in** *aGroup*) *ag-r-zero*: $x \in carrier\ A \Longrightarrow x \pm \mathbf{0} = x$
⟨proof⟩

lemma (**in** *aGroup*) *ag-l-inv1*: $x \in carrier\ A \Longrightarrow (-_a\ x) \pm x = \mathbf{0}$
⟨proof⟩

lemma (**in** *aGroup*) *ag-r-inv1*: $x \in carrier\ A \Longrightarrow x \pm (-_a\ x) = \mathbf{0}$
⟨proof⟩

lemma (**in** *aGroup*) *ag-pOp-assoc*: $\llbracket x \in carrier\ A;\ y \in carrier\ A;\ z \in carrier\ A\rrbracket \Longrightarrow (x \pm y) \pm z = x \pm (y \pm z)$
⟨proof⟩

lemma (**in** *aGroup*) *ag-inv-unique*: $\llbracket x \in carrier\ A;\ y \in carrier\ A;\ x \pm y = \mathbf{0}\rrbracket \Longrightarrow y = -_a\ x$
⟨proof⟩

lemma (**in** *aGroup*) *ag-inv-inj*: $\llbracket x \in carrier\ A;\ y \in carrier\ A;\ x \neq y\rrbracket \Longrightarrow (-_a\ x) \neq (-_a\ y)$

⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assocTr41*:$[\![a \in carrier\ A; b \in carrier\ A; c \in carrier\ A; d \in carrier\ A]\!] \Longrightarrow a \pm b \pm c \pm d = a \pm b \pm (c \pm d)$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assocTr42*:$[\![a \in carrier\ A; b \in carrier\ A; c \in carrier\ A; d \in carrier\ A]\!] \Longrightarrow a \pm b \pm c \pm d = a \pm (b \pm c) \pm d$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assocTr43*:$[\![a \in carrier\ A; b \in carrier\ A; c \in carrier\ A; d \in carrier\ A]\!] \Longrightarrow a \pm b \pm (c \pm d) = a \pm (b \pm c) \pm d$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assoc-cancel*:$[\![a \in carrier\ A; b \in carrier\ A; c \in carrier\ A]\!] \Longrightarrow a \pm -_a\ b \pm (b \pm -_a\ c) = a \pm -_a\ c$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-p-inv*:$[\![x \in carrier\ A; y \in carrier\ A]\!] \Longrightarrow (-_a\ (x \pm y)) = (-_a\ x) \pm (-_a\ y)$
⟨*proof*⟩

lemma (**in** *aGroup*) *gEQAddcross*: $[\![l1 \in carrier\ A; l2 \in carrier\ A; r1 \in carrier\ A; r1 \in carrier\ A; l1 = r2; l2 = r1]\!] \Longrightarrow l1 \pm l2 = r1 \pm r2$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-sol1*:$[\![a \in carrier\ A; x\in carrier\ A; b\in carrier\ A; a \pm x = b]\!] \Longrightarrow x = (-_a\ a) \pm b$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-sol2*:$[\![a \in carrier\ A; x\in carrier\ A; b\in carrier\ A; x \pm a = b]\!] \Longrightarrow x = b \pm (-_a\ a)$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-add4-rel*:$[\![a \in carrier\ A; b \in carrier\ A; c \in carrier\ A; d \in carrier\ A\ ]\!] \Longrightarrow a \pm b \pm (c \pm d) = a \pm c \pm (b \pm d)$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-inv-inv*:$x \in carrier\ A \Longrightarrow -_a\ (-_a\ x) = x$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-inv-zero*:$-_a\ \mathbf{0} = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-diff-minus*:$[\![a \in carrier\ A; b \in carrier\ A; c \in carrier\ A; a \pm (-_a\ b) = c]\!] \Longrightarrow b \pm (-_a\ a) = (-_a\ c)$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-cancel-l*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;\ c \pm a = c \pm b\ ]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-cancel-r*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;\ a \pm c = b \pm c\ ]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-diffzero*:$[\![a \in carrier\ A;\ b \in carrier\ A]\!] \Longrightarrow$
$(a = b) = (a \pm (-_a\ b) = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-diffzero1*:$[\![a \in carrier\ A;\ b \in carrier\ A]\!] \Longrightarrow$
$(a = b) = ((-_a\ a) \pm b = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-neq-diffnonzero*:$[\![a \in carrier\ A;\ b \in carrier\ A]\!] \Longrightarrow$
$(a \neq b) = (a \pm (-_a\ b) \neq \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-plus-zero*:$[\![x \in carrier\ A;\ y \in carrier\ A]\!] \Longrightarrow$
$(x = -_a\ y) = (x \pm y = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-nsubg*:$A \mathrel{+>} H \Longrightarrow (b\text{-}ag\ A) \rhd H$
⟨*proof*⟩

lemma (**in** *aGroup*) *subg-asubg*:$b\text{-}ag\ G \gg H \Longrightarrow G \mathrel{+>} H$
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-test*:$[\![H \subseteq carrier\ A;\ H \neq \{\};$
$\forall a{\in}H.\ \forall b{\in}H.\ (a \pm (-_a\ b) \in H)]\!] \Longrightarrow A \mathrel{+>} H$
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-zero*:$A \mathrel{+>} \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-whole*:$A \mathrel{+>} carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *Ag-ind-carrier*:$bij\text{-}to\ f\ (carrier\ A)\ (D{::}'d\ set) \Longrightarrow$
$carrier\ (Ag\text{-}ind\ A\ f) = f\ `\ (carrier\ A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *Ag-ind-aGroup*:$[\![f \in carrier\ A \rightarrow D;$
$bij\text{-}to\ f\ (carrier\ A)\ (D{::}'d\ set)]\!] \Longrightarrow aGroup\ (Ag\text{-}ind\ A\ f)$
⟨*proof*⟩

3.18.1 Homomorphism of abelian groups

definition
aHom :: [('a, 'm) aGroup-scheme, ('b, 'm1) aGroup-scheme] ⇒ ('a ⇒ 'b) set
where
aHom A B = {f. f ∈ carrier A → carrier B ∧ f ∈ extensional (carrier A) ∧
(∀ a∈carrier A. ∀ b∈carrier A. f (a $\pm_A$ b) = (f a) $\pm_B$ (f b))}

definition
compos :: [('a, 'm) aGroup-scheme, 'b ⇒ 'c, 'a ⇒ 'b] ⇒ 'a ⇒ 'c **where**
compos A g f = compose (carrier A) g f

definition
ker :: [('a, 'm) aGroup-scheme, ('b, 'm1) aGroup-scheme] ⇒ ('a ⇒ 'b)
⇒ 'a set ((3ker$_{-,-}$ -) [82,82,83]82) **where**
$ker_{F,G}$ f = {a. a ∈ carrier F ∧ f a = ($\mathbf{0}_G$)}

definition
injec :: [('a, 'm) aGroup-scheme, ('b, 'm1) aGroup-scheme, 'a ⇒ 'b]
⇒ bool ((3injec$_{-,-}$ -) [82,82,83]82) **where**
$injec_{F,G}$ f ⟷ f ∈ aHom F G ∧ $ker_{F,G}$ f = {$\mathbf{0}_F$}

definition
surjec :: [('a, 'm) aGroup-scheme, ('b, 'm1) aGroup-scheme, 'a ⇒ 'b]
⇒ bool ((3surjec$_{-,-}$ -) [82,82,83]82) **where**
$surjec_{F,G}$ f ⟷ f ∈ aHom F G ∧ surj-to f (carrier F) (carrier G)

definition
bijec :: [('a, 'm) aGroup-scheme, ('b, 'm1) aGroup-scheme, 'a ⇒ 'b]
⇒ bool ((3bijec$_{-,-}$ -) [82,82,83]82) **where**
$bijec_{F,G}$ f ⟷ $injec_{F,G}$ f ∧ $surjec_{F,G}$ f

definition
ainvf :: [('a, 'm) aGroup-scheme, ('b, 'm1) aGroup-scheme, 'a ⇒ 'b]
⇒ ('b ⇒ 'a) ((3ainvf$_{-,-}$ -) [82,82,83]82) **where**
$ainvf_{F,G}$ f = invfun (carrier F) (carrier G) f

lemma *aHom-mem*:⟦aGroup F; aGroup G; f ∈ aHom F G; a ∈ carrier F⟧ ⟹
f a ∈ carrier G
⟨proof⟩

lemma *aHom-func*:f ∈ aHom F G ⟹ f ∈ carrier F → carrier G
⟨proof⟩

lemma *aHom-add*:⟦aGroup F; aGroup G; f ∈ aHom F G; a ∈ carrier F;
b ∈ carrier F⟧ ⟹ f (a $\pm_F$ b) = (f a) $\pm_G$ (f b)
⟨proof⟩

lemma *aHom-0-0*:⟦aGroup F; aGroup G; f ∈ aHom F G⟧ ⟹ f ($\mathbf{0}_F$) = $\mathbf{0}_G$
⟨proof⟩

lemma *ker-inc-zero*:$[\![aGroup\ F;\ aGroup\ G;\ f \in aHom\ F\ G]\!] \Longrightarrow \mathbf{0}_F \in ker_{F,G}\ f$
⟨*proof*⟩

lemma *aHom-inv-inv*:$[\![aGroup\ F;\ aGroup\ G;\ f \in aHom\ F\ G;\ a \in carrier\ F]\!] \Longrightarrow$
$f\ (-_{a F}\ a) = -_{a G}\ (f\ a)$
⟨*proof*⟩

lemma *aHom-compos*:$[\![aGroup\ L;\ aGroup\ M;\ aGroup\ N;\ f \in aHom\ L\ M;\ g \in aHom\ M\ N\ ]\!]$
$\Longrightarrow compos\ L\ g\ f \in aHom\ L\ N$
⟨*proof*⟩

lemma *aHom-compos-assoc*:$[\![aGroup\ K;\ aGroup\ L;\ aGroup\ M;\ aGroup\ N;\ f \in aHom\ K\ L;$
$g \in aHom\ L\ M;\ h \in aHom\ M\ N\ ]\!] \Longrightarrow$
$compos\ K\ h\ (compos\ K\ g\ f) = compos\ K\ (compos\ L\ h\ g)\ f$
⟨*proof*⟩

lemma *injec-inj-on*:$[\![aGroup\ F;\ aGroup\ G;\ injec_{F,G}\ f]\!] \Longrightarrow inj\text{-}on\ f\ (carrier\ F)$
⟨*proof*⟩

lemma *surjec-surj-to*:$surjec_{R,S}\ f \Longrightarrow surj\text{-}to\ f\ (carrier\ R)\ (carrier\ S)$
⟨*proof*⟩

lemma *compos-bijec*:$[\![aGroup\ E;\ aGroup\ F;\ aGroup\ G;\ bijec_{E,F}\ f;\ bijec_{F,G}\ g]\!]$
$\Longrightarrow$
$bijec_{E,G}\ (compos\ E\ g\ f)$
⟨*proof*⟩

lemma *ainvf-aHom*:$[\![aGroup\ F;\ aGroup\ G;\ bijec_{F,G}\ f]\!] \Longrightarrow$
$ainvf_{F,G}\ f \in aHom\ G\ F$
⟨*proof*⟩

lemma *ainvf-bijec*:$[\![aGroup\ F;\ aGroup\ G;\ bijec_{F,G}\ f]\!] \Longrightarrow bijec_{G,F}\ (ainvf_{F,G}\ f)$
⟨*proof*⟩

lemma *ainvf-l*:$[\![aGroup\ E;\ aGroup\ F;\ bijec_{E,F}\ f;\ x \in carrier\ E]\!] \Longrightarrow$
$(ainvf_{E,F}\ f)\ (f\ x) = x$
⟨*proof*⟩

lemma (**in** *aGroup*) *aI-aHom*:$aI_A \in aHom\ A\ A$
⟨*proof*⟩

lemma *compos-aI-l*:$[\![aGroup\ A;\ aGroup\ B;\ f \in aHom\ A\ B]\!] \Longrightarrow compos\ A\ aI_B\ f = f$
⟨*proof*⟩

lemma *compos-aI-r*:$[\![aGroup\ A;\ aGroup\ B;\ f \in aHom\ A\ B]\!] \Longrightarrow compos\ A\ f\ aI_A$

$= f$
⟨*proof*⟩

lemma *compos-aI-surj*:⟦*aGroup A*; *aGroup B*; *f* ∈ *aHom A B*; *g* ∈ *aHom B A*;
compos A g f = aI_A⟧ ⟹ $surjec_{B,A}$ *g*
⟨*proof*⟩

lemma *compos-aI-inj*:⟦*aGroup A*; *aGroup B*; *f* ∈ *aHom A B*; *g* ∈ *aHom B A*;
compos A g f = aI_A⟧ ⟹ $injec_{A,B}$ *f*
⟨*proof*⟩

lemma (**in** *aGroup*) *Ag-ind-aHom*:⟦*f* ∈ *carrier A* → *D*;
bij-to f (*carrier A*) (*D*::′*d set*)⟧ ⟹ *Agii A f* ∈ *aHom A* (*Ag-ind A f*)
⟨*proof*⟩

lemma (**in** *aGroup*) *Agii-mem*:⟦*f* ∈ *carrier A* → *D*; *x* ∈ *carrier A*;
bij-to f (*carrier A*) (*D*::′*d set*)⟧ ⟹ *Agii A f x* ∈ *carrier* (*Ag-ind A f*)
⟨*proof*⟩

lemma *Ag-ind-bijec*:⟦*aGroup A*; *f* ∈ *carrier A* → *D*;
bij-to f (*carrier A*) (*D*::′*d set*)⟧ ⟹ $bijec_{A,\ (Ag\text{-}ind\ A\ f)}$ (*Agii A f*)
⟨*proof*⟩

definition
aimg :: [(′*b*, ′*m1*) *aGroup-scheme*, -, ′*b* ⇒ ′*a*]
⇒ ′*a aGroup* ((*3aimg*$_{-,-}$ -) [*82*,*82*,*83*]*82*) **where**
$aimg_{F,A}$ *f* = *A* ⦇ *carrier* := *f* ‘ (*carrier F*), *pop* := *pop A*, *mop* := *mop A*,
zero := *zero A*⦈

lemma *ker-subg*:⟦*aGroup F*; *aGroup G*; *f* ∈ *aHom F G* ⟧ ⟹ *F* +> $ker_{F,G}$ *f*
⟨*proof*⟩

3.18.2 Quotient abelian group

definition
ar-coset :: [′*a*, - , ′*a set*] ⇒ ′*a set*
((*3*- ⊎- -) [*66*,*66*,*67*]*66*) **where**
ar-coset a A H = *H* $\cdot_{(b\text{-}ag\ A)}$ *a*

definition
set-ar-cos :: [- , ′*a set*] ⇒ ′*a set set* **where**
set-ar-cos A I = {*X*. ∃ *a*∈*carrier A*. *X* = *ar-coset a A I*}

definition
aset-sum :: [- , ′*a set*, ′*a set*] ⇒ ′*a set* **where**
aset-sum A H K = *s-top* (*b-ag A*) *H K*

abbreviation
ASBOP1 (**infix** ∓1 *60*) **where**

$H \mp_A K == \textit{aset-sum}\ A\ H\ K$

lemma (in *aGroup*) *ag-a-in-ar-cos*: $[\![A +> H;\ a \in \textit{carrier}\ A]\!] \Longrightarrow a \in a \uplus_A H$
⟨*proof*⟩

lemma (in *aGroup*) *r-cos-subset*: $[\![A +> H;\ X \in \textit{set-rcs}\ (\textit{b-ag}\ A)\ H]\!] \Longrightarrow$
$X \subseteq \textit{carrier}\ A$
⟨*proof*⟩

lemma (in *aGroup*) *asubg-costOp-commute*: $[\![A +> H;\ x \in \textit{set-rcs}\ (\textit{b-ag}\ A)\ H;$
$y \in \textit{set-rcs}\ (\textit{b-ag}\ A)\ H]\!] \Longrightarrow$
$\textit{c-top}\ (\textit{b-ag}\ A)\ H\ x\ y = \textit{c-top}\ (\textit{b-ag}\ A)\ H\ y\ x$
⟨*proof*⟩

lemma (in *aGroup*) *Subg-Qgroup*: $A +> H \Longrightarrow \textit{aGroup}\ (\textit{aqgrp}\ A\ H)$
⟨*proof*⟩

lemma (in *aGroup*) *plus-subgs*: $[\![A +> H1;\ A +> H2]\!] \Longrightarrow A +> H1 \mp H2$
⟨*proof*⟩

lemma (in *aGroup*) *set-sum*: $[\![H \subseteq \textit{carrier}\ A;\ K \subseteq \textit{carrier}\ A]\!] \Longrightarrow$
$H \mp K = \{x.\ \exists h{\in}H.\ \exists k{\in}K.\ x = h \pm k\}$
⟨*proof*⟩

lemma (in *aGroup*) *mem-set-sum*: $[\![H \subseteq \textit{carrier}\ A;\ K \subseteq \textit{carrier}\ A;$
$x \in H \mp K\ ]\!] \Longrightarrow \exists h{\in}H.\ \exists k{\in}K.\ x = h \pm k$
⟨*proof*⟩

lemma (in *aGroup*) *mem-sum-subgs*: $[\![A +> H;\ A +> K;\ h \in H;\ k \in K]\!] \Longrightarrow$
$h \pm k \in H \mp K$
⟨*proof*⟩

lemma (in *aGroup*) *aqgrp-carrier*: $A +> H \Longrightarrow$
$\textit{set-rcs}\ (\textit{b-ag}\ A\ )\ H = \textit{set-ar-cos}\ A\ H$
⟨*proof*⟩

lemma (in *aGroup*) *unit-in-set-ar-cos*: $A +> H \Longrightarrow H \in \textit{set-ar-cos}\ A\ H$
⟨*proof*⟩

lemma (in *aGroup*) *aqgrp-pOp-maps*: $[\![A +> H;\ a \in \textit{carrier}\ A;\ b \in \textit{carrier}\ A]\!] \Longrightarrow$
$\textit{pop}\ (\textit{aqgrp}\ A\ H)\ (a \uplus_A H)\ (b \uplus_A H) = (a \pm b) \uplus_A H$
⟨*proof*⟩

lemma (in *aGroup*) *aqgrp-mOp-maps*: $[\![A +> H;\ a \in \textit{carrier}\ A]\!] \Longrightarrow$
$\textit{mop}\ (\textit{aqgrp}\ A\ H)\ (a \uplus_A H) = (-_a\ a) \uplus_A H$
⟨*proof*⟩

lemma (in *aGroup*) *aqgrp-zero*: $A +> H \Longrightarrow \textit{zero}\ (\textit{aqgrp}\ A\ H) = H$
⟨*proof*⟩

lemma (**in** *aGroup*) *arcos-fixed*:$[\![A +> H;\ a \in carrier\ A;\ h \in H\ ]\!] \Longrightarrow$
$a \uplus_A H = (h \pm a) \uplus_A H$
⟨*proof*⟩

definition
rind-hom :: $[('a, 'more)\ aGroup\text{-}scheme, ('b, 'more1)\ aGroup\text{-}scheme, ('a \Rightarrow 'b)] \Rightarrow ('a\ set \Rightarrow 'b\ )$ **where**
$rind\text{-}hom\ A\ B\ f = (\lambda X{\in}(set\text{-}ar\text{-}cos\ A\ (ker_{A,B}\ f)).\ f\ (SOME\ x.\ x \in X))$

abbreviation
RIND-HOM ((*3*-°-,-) [*82*,*82*,*83*]*82*) **where**
$f^{\circ}{}_{F,G} == rind\text{-}hom\ F\ G\ f$

3.19 Direct product and direct sum of abelian groups, in general case

definition
Un-carrier :: $['i\ set, 'i \Rightarrow ('a, 'more)\ aGroup\text{-}scheme] \Rightarrow 'a\ set$ **where**
$Un\text{-}carrier\ I\ A = \bigcup\{X.\ \exists\, i{\in}I.\ X = carrier\ (A\ i)\}$

definition
carr-prodag :: $['i\ set, 'i \Rightarrow ('a, 'more)\ aGroup\text{-}scheme] \Rightarrow ('i \Rightarrow 'a\ )\ set$ **where**
$carr\text{-}prodag\ I\ A = \{f.\ f \in extensional\ I \wedge f \in I \rightarrow (Un\text{-}carrier\ I\ A) \wedge$
$(\forall\, i{\in}I.\ f\ i \in carrier\ (A\ i))\}$

definition
prod-pOp :: $['i\ set, 'i \Rightarrow ('a, 'more)\ aGroup\text{-}scheme] \Rightarrow$
$('i \Rightarrow 'a) \Rightarrow ('i \Rightarrow 'a) \Rightarrow ('i \Rightarrow 'a)$ **where**
$prod\text{-}pOp\ I\ A = (\lambda f{\in}carr\text{-}prodag\ I\ A.\ \lambda g{\in}carr\text{-}prodag\ I\ A.$
$\lambda x{\in}I.\ (f\ x) \pm_{(A\ x)} (g\ x))$

definition
prod-mOp :: $['i\ set, 'i \Rightarrow ('a, 'more)\ aGroup\text{-}scheme] \Rightarrow$
$('i \Rightarrow 'a) \Rightarrow ('i \Rightarrow 'a)$ **where**
$prod\text{-}mOp\ I\ A = (\lambda f{\in}carr\text{-}prodag\ I\ A.\ \lambda x{\in}I.\ (-_{a(A\ x)}\ (f\ x)))$

definition
prod-zero :: $['i\ set, 'i \Rightarrow ('a, 'more)\ aGroup\text{-}scheme] \Rightarrow ('i \Rightarrow 'a)$ **where**
$prod\text{-}zero\ I\ A = (\lambda x{\in}I.\ \mathbf{0}_{(A\ x)})$

definition
prodag :: $['i\ set, 'i \Rightarrow ('a, 'more)\ aGroup\text{-}scheme] \Rightarrow ('i \Rightarrow 'a)\ aGroup$ **where**
$prodag\ I\ A = (\!|\ carrier = carr\text{-}prodag\ I\ A,$
$pop = prod\text{-}pOp\ I\ A,\ mop = prod\text{-}mOp\ I\ A,$
$zero = prod\text{-}zero\ I\ A|\!)$

definition

PRoject :: $['i\ set,\ 'i \Rightarrow ('a,\ 'more)\ aGroup\text{-}scheme,\ 'i]$
$\Rightarrow ('i \Rightarrow 'a) \Rightarrow 'a\ ((3\pi_{-,-,-})\ [82,82,83]82)$ **where**
PRoject I A x $= (\lambda f \in carr\text{-}prodag\ I\ A.\ f\ x)$

abbreviation
PRODag $((a\Pi_{-}\ \text{-})\ [72,73]72)$ **where**
$a\Pi_I\ A == prodag\ I\ A$

lemma *prodag-comp-i*:$[\![a \in carr\text{-}prodag\ I\ A;\ i \in I]\!] \Longrightarrow (a\ i) \in carrier\ (A\ i)$
⟨*proof*⟩

lemma *prod-pOp-func*:$\forall k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$prod\text{-}pOp\ I\ A \in carr\text{-}prodag\ I\ A \to carr\text{-}prodag\ I\ A \to carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-pOp-mem*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ X \in carr\text{-}prodag\ I\ A;$
$Y \in carr\text{-}prodag\ I\ A]\!] \Longrightarrow prod\text{-}pOp\ I\ A\ X\ Y \in carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-pOp-mem-i*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ X \in carr\text{-}prodag\ I\ A;$
$Y \in carr\text{-}prodag\ I\ A;\ i \in I]\!] \Longrightarrow prod\text{-}pOp\ I\ A\ X\ Y\ i = (X\ i) \pm_{(A\ i)} (Y\ i)$
⟨*proof*⟩

lemma *prod-mOp-func*:$\forall k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$prod\text{-}mOp\ I\ A \in carr\text{-}prodag\ I\ A \to carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-mOp-mem*:$[\![\forall j{\in}I.\ aGroup\ (A\ j);\ X \in carr\text{-}prodag\ I\ A]\!] \Longrightarrow$
$prod\text{-}mOp\ I\ A\ X \in carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-mOp-mem-i*:$[\![\forall j{\in}I.\ aGroup\ (A\ j);\ X \in carr\text{-}prodag\ I\ A;\ i \in I]\!] \Longrightarrow$
$prod\text{-}mOp\ I\ A\ X\ i = -_{a(A\ i)}\ (X\ i)$
⟨*proof*⟩

lemma *prod-zero-func*:$\forall k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$prod\text{-}zero\ I\ A \in carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-zero-i*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ i \in I]\!] \Longrightarrow$
$prod\text{-}zero\ I\ A\ i = \mathbf{0}_{(A\ i)}$
⟨*proof*⟩

lemma *carr-prodag-mem-eq*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ X \in carr\text{-}prodag\ I\ A;$
$Y \in carr\text{-}prodag\ I\ A;\ \forall l{\in}I.\ (X\ l) = (Y\ l)\ ]\!] \Longrightarrow X = Y$
⟨*proof*⟩

lemma *prod-pOp-assoc*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ a \in carr\text{-}prodag\ I\ A;$
$b \in carr\text{-}prodag\ I\ A;\ c \in carr\text{-}prodag\ I\ A]\!] \Longrightarrow$

prod-pOp I A (*prod-pOp I A a b*) *c* =
prod-pOp I A a (*prod-pOp I A b c*)
⟨*proof*⟩

lemma *prod-pOp-commute*:⟦∀ *k*∈*I*. *aGroup* (*A k*); *a* ∈ *carr-prodag I A*;
b ∈ *carr-prodag I A*⟧ ⟹
prod-pOp I A a b = *prod-pOp I A b a*
⟨*proof*⟩

lemma *prodag-aGroup*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹ *aGroup* (*prodag I A*)
⟨*proof*⟩

lemma *prodag-carrier*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
carrier (*prodag I A*) = *carr-prodag I A*
⟨*proof*⟩

lemma *prodag-elemfun*:⟦∀ *k*∈*I*. *aGroup* (*A k*); *f* ∈ *carrier* (*prodag I A*)⟧ ⟹
f ∈ *extensional I*
⟨*proof*⟩

lemma *prodag-component*:⟦*f* ∈ *carrier* (*prodag I A*); *i* ∈ *I* ⟧ ⟹
f i ∈ *carrier* (*A i*)
⟨*proof*⟩

lemma *prodag-pOp*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
pop (*prodag I A*) = *prod-pOp I A*
⟨*proof*⟩

lemma *prodag-iOp*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
mop (*prodag I A*) = *prod-mOp I A*
⟨*proof*⟩

lemma *prodag-zero*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
zero (*prodag I A*) = *prod-zero I A*
⟨*proof*⟩

lemma *prodag-sameTr0*:⟦∀ *k*∈*I*. *aGroup* (*A k*); ∀ *k*∈*I*. *A k* = *B k*⟧
⟹ *Un-carrier I A* = *Un-carrier I B*
⟨*proof*⟩

lemma *prodag-sameTr1*:⟦∀ *k*∈*I*. *aGroup* (*A k*); ∀ *k*∈*I*. *A k* = *B k*⟧
⟹ *carr-prodag I A* = *carr-prodag I B*
⟨*proof*⟩

lemma *prodag-sameTr2*:⟦∀ *k*∈*I*. *aGroup* (*A k*); ∀ *k*∈*I*. *A k* = *B k*⟧
⟹ *prod-pOp I A* = *prod-pOp I B*
⟨*proof*⟩

lemma *prodag-sameTr3*:⟦∀ *k*∈*I*. *aGroup* (*A k*); ∀ *k*∈*I*. *A k* = *B k*⟧

$\Longrightarrow$ *prod-mOp I A = prod-mOp I B*
⟨*proof*⟩

lemma *prodag-sameTr4*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ \forall k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow$ *prod-zero I A = prod-zero I B*
⟨*proof*⟩

lemma *prodag-same*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ \forall k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow$ *prodag I A = prodag I B*
⟨*proof*⟩

lemma *project-mem*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ j \in I;\ x \in carrier\ (prodag\ I\ A)]\!] \Longrightarrow$
$(PRoject\ I\ A\ j)\ x\ \in carrier\ (A\ j)$
⟨*proof*⟩

lemma *project-aHom*:$[\![\forall k{\in}I.\ aGroup\ (A\ k);\ j \in I]\!] \Longrightarrow$
$PRoject\ I\ A\ j \in aHom\ (prodag\ I\ A)\ (A\ j)$
⟨*proof*⟩

lemma *project-aHom1*:$\forall k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$\forall j \in I.\ PRoject\ I\ A\ j \in aHom\ (prodag\ I\ A)\ (A\ j)$
⟨*proof*⟩

definition
A-to-prodag :: [('a, 'm) *aGroup-scheme*, 'i *set*, 'i ⇒('a ⇒ 'b),
'i ⇒ ('b, 'm1) *aGroup-scheme*] ⇒ ('a ⇒ ('i ⇒'b)) **where**
A-to-prodag A I S B = ($\lambda a{\in}carrier\ A.\ \lambda k{\in}I.\ S\ k\ a$)

lemma *A-to-prodag-mem*:$[\![aGroup\ A;\ \forall k{\in}I.\ aGroup\ (B\ k);\ \forall k{\in}I.\ (S\ k) \in$
$aHom\ A\ (B\ k);\ x \in carrier\ A\ ]\!] \Longrightarrow A\text{-}to\text{-}prodag\ A\ I\ S\ B\ x \in carr\text{-}prodag\ I\ B$
⟨*proof*⟩

lemma *A-to-prodag-aHom*:$[\![aGroup\ A;\ \forall k{\in}I.\ aGroup\ (B\ k);\ \forall k{\in}I.\ (S\ k) \in$
$aHom\ A\ (B\ k)\ ]\!] \Longrightarrow A\text{-}to\text{-}prodag\ A\ I\ S\ B \in aHom\ A\ (a\Pi_I\ B)$
⟨*proof*⟩

definition
finiteHom :: ['i *set*, 'i ⇒ ('a, 'more) *aGroup-scheme*, 'i ⇒ 'a] ⇒ *bool* **where**
$finiteHom\ I\ A\ f \longleftrightarrow f \in carr\text{-}prodag\ I\ A \wedge (\exists H.\ H \subseteq I \wedge finite\ H \wedge ($
$\forall j \in (I - H).\ (f\ j) = \mathbf{0}_{(A\ j)}))$

definition
carr-dsumag :: ['i *set*, 'i ⇒ ('a, 'more) *aGroup-scheme*] ⇒ ('i ⇒ 'a) *set* **where**
carr-dsumag I A = {*f*. *finiteHom I A f*}

definition
dsumag :: ['i *set*, 'i ⇒ ('a, 'more) *aGroup-scheme*] ⇒ ('i ⇒ 'a) *aGroup* **where**

dsumag I A = (| *carrier* = *carr-dsumag I A*,
pop = *prod-pOp I A*, *mop* = *prod-mOp I A*,
zero = *prod-zero I A*|)

definition
dProj :: ['i set, 'i ⇒ ('a, 'more) aGroup-scheme, 'i]
⇒ ('i ⇒ 'a) ⇒ 'a **where**
dProj I A x = (λf∈*carr-dsumag I A*. *f x*)

abbreviation
DSUMag ((a⨁ _ -) [72,73]72) **where**
a⨁ $_I$ *A* == *dsumag I A*

lemma *dsum-pOp-func*:∀ k∈I. *aGroup* (*A k*) ⟹
prod-pOp I A ∈ *carr-dsumag I A* → *carr-dsumag I A* → *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-pOp-mem*:⟦∀ k∈I. *aGroup* (*A k*); *X* ∈ *carr-dsumag I A*;
Y ∈ *carr-dsumag I A*⟧ ⟹ *prod-pOp I A X Y* ∈ *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-iOp-func*:∀ k∈I. *aGroup* (*A k*) ⟹
prod-mOp I A ∈ *carr-dsumag I A* → *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-iOp-mem*:⟦∀ j∈I. *aGroup* (*A j*); *X* ∈ *carr-dsumag I A*⟧ ⟹
prod-mOp I A X ∈ *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-zero-func*:∀ k∈I. *aGroup* (*A k*) ⟹
prod-zero I A ∈ *carr-dsumag I A*
⟨*proof*⟩

lemma *dsumag-sub-prodag*:∀ k∈I. *aGroup* (*A k*) ⟹
carr-dsumag I A ⊆ *carr-prodag I A*
⟨*proof*⟩

lemma *carrier-dsumag*:∀ k∈I. *aGroup* (*A k*) ⟹
carrier (*dsumag I A*) = *carr-dsumag I A*
⟨*proof*⟩

lemma *dsumag-elemfun*:⟦∀ k∈I. *aGroup* (*A k*); *f* ∈ *carrier* (*dsumag I A*)⟧ ⟹
f ∈ *extensional I*
⟨*proof*⟩

lemma *dsumag-aGroup*:∀ k∈I. *aGroup* (*A k*) ⟹ *aGroup* (*dsumag I A*)
⟨*proof*⟩

lemma *dsumag-pOp*:∀ k∈I. *aGroup* (*A k*) ⟹

pop (*dsumag I A*) = *prod-pOp I A*
⟨*proof*⟩

lemma *dsumag-mOp*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
mop (*dsumag I A*) = *prod-mOp I A*
⟨*proof*⟩

lemma *dsumag-zero*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
zero (*dsumag I A*) = *prod-zero I A*
⟨*proof*⟩

3.19.1 Characterization of a direct product

lemma *direct-prod-mem-eq*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *f* ∈ *carrier* ($a\Pi_I$ *A*);
g ∈ *carrier* ($a\Pi_I$ *A*); ∀ *j*∈*I*. (*PRoject I A j*) *f* = (*PRoject I A j*) *g*⟧ ⟹
f = *g*
⟨*proof*⟩

lemma *map-family-fun*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *aGroup S*;
∀ *j*∈*I*. ((*g j*) ∈ *aHom S* (*A j*)); *x* ∈ *carrier S*⟧ ⟹
(λ*y* ∈ *carrier S*. (λ*j*∈*I*. (*g j*) *y*)) *x* ∈ *carrier* ($a\Pi_I$ *A*)
⟨*proof*⟩

lemma *map-family-aHom*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *aGroup S*;
∀ *j*∈*I*. ((*g j*) ∈ *aHom S* (*A j*))⟧ ⟹
(λ*y* ∈ *carrier S*. (λ*j*∈*I*. (*g j*) *y*)) ∈ *aHom S* ($a\Pi_I$ *A*)
⟨*proof*⟩

lemma *map-family-triangle*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *aGroup S*;
∀ *j*∈*I*. ((*g j*) ∈ *aHom S* (*A j*))⟧ ⟹ ∃!*f*. *f* ∈ *aHom S* ($a\Pi_I$ *A*) ∧
(∀ *j*∈*I*. *compos S* (*PRoject I A j*) *f* = (*g j*))
⟨*proof*⟩

lemma *Ag-ind-triangle*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *j* ∈ *I*; *f* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to f (*carrier* ($a\Pi_I$ *A*)) (*B*::'*d set*); *j* ∈ *I*⟧ ⟹
compos ($a\Pi_I$ *A*) (*compos* (*Ag-ind* ($a\Pi_I$ *A*) *f*)(*PRoject I A j*) ($ainvf_{(a\Pi_I A)}$, (*Ag-ind* ($a\Pi_I$ *A*) *f*) (*Agii* ($a\Pi_I$ *A*) *f*))) (*Agii* ($a\Pi_I$ *A*) *f*) =
PRoject I A j
⟨*proof*⟩

definition
ProjInd :: ['*i set*, '*i* ⇒ ('*a*, '*m*) *aGroup-scheme*, ('*i* ⇒ '*a*) ⇒ '*d*, '*i*] ⇒
('*d* ⇒ '*a*) **where**
ProjInd I A f j = *compos* (*Ag-ind* ($a\Pi_I$ *A*) *f*)(*PRoject I A j*) ($ainvf_{(a\Pi_I A)}$, (*Ag-ind* ($a\Pi_I$ *A*) *f*) (*Agii* ($a\Pi_I$ *A*) *f*))

lemma *ProjInd-aHom*:⟦∀ *j*∈ *I*. *aGroup* (*A j*); *j* ∈ *I*; *f* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to f (*carrier* ($a\Pi_I$ *A*)) (*B*::′*d set*); *j* ∈ *I*⟧ ⟹
(*ProjInd I A f j*) ∈ *aHom* (*Ag-ind* ($a\Pi_I$ *A*) *f*) (*A j*)
⟨*proof*⟩

lemma *ProjInd-aHom1*:⟦∀ *j*∈ *I*. *aGroup* (*A j*); *f* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to f (*carrier* ($a\Pi_I$ *A*)) (*B*::′*d set*)⟧ ⟹
∀ *j*∈*I*. (*ProjInd I A f j*) ∈ *aHom* (*Ag-ind* ($a\Pi_I$ *A*) *f*) (*A j*)
⟨*proof*⟩

lemma *ProjInd-mem-eq*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *f* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to f (*carrier* ($a\Pi_I$ *A*)) *B*; *aGroup S*; *x* ∈ *carrier* (*Ag-ind* ($a\Pi_I$ *A*) *f*);
y ∈ *carrier* (*Ag-ind* ($a\Pi_I$ *A*) *f*);
∀ *j*∈*I*. (*ProjInd I A f j x* = *ProjInd I A f j y*)⟧ ⟹ *x* = *y*
⟨*proof*⟩

lemma *ProjInd-mem-eq1*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *f* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to f (*carrier* ($a\Pi_I$ *A*)) *B*; *aGroup S*;
h ∈ *aHom* (*Ag-ind* ($a\Pi_I$ *A*) *f*) (*Ag-ind* ($a\Pi_I$ *A*) *f*);
∀ *j*∈*I*. *compos* (*Ag-ind* ($a\Pi_I$ *A*) *f*) (*ProjInd I A f j*) *h* = *ProjInd I A f j*⟧
⟹ *h* = *ag-idmap* (*Ag-ind* ($a\Pi_I$ *A*) *f*)
⟨*proof*⟩

lemma *Ag-ind-triangle1*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *f* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to f (*carrier* ($a\Pi_I$ *A*)) (*B*::′*d set*); *j* ∈ *I*⟧ ⟹
compos ($a\Pi_I$ *A*) (*ProjInd I A f j*) (*Agii* ($a\Pi_I$ *A*) *f*) = *PRoject I A j*
⟨*proof*⟩

lemma *map-family-triangle1*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *f* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to f (*carrier* ($a\Pi_I$ *A*)) (*B*::′*d set*); *aGroup S*;
∀ *j*∈*I*. ((*g j*) ∈ *aHom S* (*A j*))⟧ ⟹ ∃!*h*. *h* ∈ *aHom S* (*Ag-ind* ($a\Pi_I$ *A*) *f*) ∧
(∀ *j*∈*I*. *compos S* (*ProjInd I A f j*) *h* = (*g j*))
⟨*proof*⟩

lemma *map-family-triangle2*:⟦*I* ≠ {}; ∀ *j*∈*I*. *aGroup* (*A j*); *aGroup S*;
∀ *j*∈*I*. *g j* ∈ *aHom S* (*A j*); *ff* ∈ *carrier* ($a\Pi_I$ *A*) → *B*;
bij-to ff (*carrier* ($a\Pi_I$ *A*)) *B*;
h1 ∈ *aHom* (*Ag-ind* ($a\Pi_I$ *A*) *ff*) *S*;
∀ *j*∈*I*. *compos* (*Ag-ind* ($a\Pi_I$ *A*) *ff*) (*g j*) *h1* = *ProjInd I A ff j*;
h2 ∈ *aHom S* (*Ag-ind* ($a\Pi_I$ *A*) *ff*);
∀ *j*∈*I*. *compos S* (*ProjInd I A ff j*) *h2* = *g j*⟧
⟹ ∀ *j*∈*I*. *compos* (*Ag-ind* ($a\Pi_I$ *A*) *ff*) (*ProjInd I A ff j*)
(*compos* (*Ag-ind* ($a\Pi_I$ *A*) *ff*) *h2 h1*) =
ProjInd I A ff j
⟨*proof*⟩

lemma *map-family-triangle3*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *aGroup S*; *aGroup S1*;
∀ *j*∈*I*. *f j* ∈ *aHom S* (*A j*); ∀ *j*∈*I*. *g j* ∈ *aHom S1* (*A j*);

$h1 \in aHom\ S1\ S$; $h2 \in aHom\ S\ S1$;
$\forall j \in I.\ compos\ S\ (g\ j)\ h2 = f\ j$;
$\forall j \in I.\ compos\ S1\ (f\ j)\ h1 = g\ j$⟧
$\Longrightarrow \forall j \in I.\ compos\ S\ (f\ j)\ (compos\ S\ h1\ h2) = f\ j$
⟨*proof*⟩

lemma *map-family-triangle4*:⟦$\forall j \in I.\ aGroup\ (A\ j)$; $aGroup\ S$;
$\forall j \in I.\ f\ j \in aHom\ S\ (A\ j)$⟧ $\Longrightarrow$
$\forall j \in I.\ compos\ S\ (f\ j)\ (ag\text{-}idmap\ S) = f\ j$
⟨*proof*⟩

lemma *prod-triangle*:⟦$I \neq \{\}$; $\forall j \in I.\ aGroup\ (A\ j)$; $aGroup\ S$;
$\forall j \in I.\ g\ j \in aHom\ S\ (A\ j)$; $ff \in carrier\ (a\Pi_I\ A) \to B$;
$bij\text{-}to\ ff\ (carrier\ (a\Pi_I\ A))\ B$;
$h1 \in aHom\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ S$;
$\forall j \in I.\ compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ (g\ j)\ h1 = ProjInd\ I\ A\ ff\ j$;
$h2 \in aHom\ S\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)$;
$\forall j \in I.\ compos\ S\ (ProjInd\ I\ A\ ff\ j)\ h2 = g\ j$⟧
$\Longrightarrow (compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ h2\ h1) = ag\text{-}idmap\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)$
⟨*proof*⟩

lemma *characterization-prodag*:⟦$I \neq \{\}$; $\forall j \in (I::'i\ set).\ aGroup\ ((A\ j)::$
$('a,\ 'm)\ aGroup\text{-}scheme)$; $aGroup\ (S::'d\ aGroup)$;
$\forall j \in I.\ ((g\ j) \in aHom\ S\ (A\ j))$; $\exists ff.\ ff \in carrier\ (a\Pi_I\ A) \to (B::'d\ set) \wedge$
$bij\text{-}to\ ff\ (carrier\ (a\Pi_I\ A))\ B$;
$\forall (S'::\ 'd\ aGroup).\ aGroup\ S' \longrightarrow$
$(\forall g'.\ (\forall j \in I.\ (g'\ j) \in aHom\ S'\ (A\ j) \longrightarrow$
$(\exists!\ f.\ f \in aHom\ S'\ S \wedge (\forall j \in I.\ compos\ S'\ (g\ j)\ f = (g'\ j)))))$⟧ $\Longrightarrow$
$\exists h.\ bijec_{(prodag\ I\ A),S}\ h$
⟨*proof*⟩

Chapter 4

Ring theory

4.1 Definition of a ring and an ideal

record $'a$ *Ring* = $'a$ *aGroup* +
  tp :: $[\,'a,\ 'a\,] \Rightarrow\ 'a$ (**infixl** $\cdot_{r1}$ *70*)
  un :: $'a$ (1_{r1})

locale *Ring* =
 fixes R (**structure**)

 assumes
  pop-closed: $pop\ R \in carrier\ R \rightarrow carrier\ R \rightarrow carrier\ R$
 and *pop-aassoc* : $[\![a \in carrier\ R;\ b \in carrier\ R;\ c \in carrier\ R]\!] \Longrightarrow$
  $(a \pm b) \pm c = a \pm (b \pm c)$
 and *pop-commute*:$[\![a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow a \pm b = b \pm a$
 and *mop-closed*:$mop\ R \in carrier\ R \rightarrow carrier\ R$
 and *l-m* :$a \in carrier\ R \Longrightarrow (-_a\ a) \pm a = \mathbf{0}$
 and *ex-zero*: $\mathbf{0} \in carrier\ R$
 and *l-zero*:$a \in carrier\ R \Longrightarrow \mathbf{0} \pm a = a$
 and *tp-closed*: $tp\ R \in carrier\ R \rightarrow carrier\ R \rightarrow carrier\ R$
 and *tp-assoc* : $[\![a \in carrier\ R;\ b \in carrier\ R;\ c \in carrier\ R]\!] \Longrightarrow$
   $(a \cdot_r b) \cdot_r c = a \cdot_r (b \cdot_r c)$
 and *tp-commute*: $[\![a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow a \cdot_r b = b \cdot_r a$
 and *un-closed*: $(1_r) \in carrier\ R$
 and *rg-distrib*: $[\![a \in carrier\ R;\ b \in carrier\ R;\ c \in carrier\ R]\!] \Longrightarrow$
   $a \cdot_r (b \pm c) = a \cdot_r b \pm a \cdot_r c$
 and *rg-l-unit*: $a \in carrier\ R \Longrightarrow (1_r) \cdot_r a = a$

definition
  zeroring :: $('a,\ 'more)$ *Ring-scheme* $\Rightarrow$ *bool* **where**
  zeroring $R \longleftrightarrow Ring\ R \wedge carrier\ R = \{\mathbf{0}_R\}$

primrec *nscal* :: $('a,\ 'more)$ *Ring-scheme* => $'a$ => *nat* => $'a$
where
  nscal-0: $nscal\ R\ x\ 0 = \mathbf{0}_R$

| *nscal-suc*: *nscal R x* (*Suc n*) = (*nscal R x n*) $\pm_R$ *x*

primrec *npow* :: (*'a*, *'more*) *Ring-scheme* => *'a* => *nat* => *'a*
where
npow-0: *npow R x 0* = 1_{rR}
| *npow-suc*: *npow R x* (*Suc n*) = (*npow R x n*) $\cdot_{rR}$ *x*

primrec *nprod* :: (*'a*, *'more*) *Ring-scheme* => (*nat* => *'a*) => *nat* => *'a*
where
nprod-0: *nprod R f 0* = *f 0*
| *nprod-suc*: *nprod R f* (*Suc n*) = (*nprod R f n*) $\cdot_{rR}$ (*f* (*Suc n*))

primrec *nsum* :: (*'a*, *'more*) *aGroup-scheme* => (*nat* => *'a*) => *nat* => *'a*
where
nsum-0: *nsum R f 0* = *f 0*
| *nsum-suc*: *nsum R f* (*Suc n*) = (*nsum R f n*) $\pm_R$ (*f* (*Suc n*))

abbreviation
NSCAL :: [*nat*, (*'a*, *'more*) *Ring-scheme*, *'a*] ⇒ *'a*
((*3* - ×_ -) [*75*,*75*,*76*]*75*) **where**
n $\times_R$ *x* == *nscal R x n*

abbreviation
NPOW :: [*'a*, (*'a*, *'more*) *Ring-scheme*, *nat*] ⇒ *'a*
((*3*- ^- -) [*77*,*77*,*78*]*77*) **where**
$a^{\hat{}R\ n}$ == *npow R a n*

abbreviation
SUM :: (*'a*, *'more*) *aGroup-scheme* => (*nat* => *'a*) => *nat* => *'a*
((*3*Σ_e - - -) [*85*,*85*,*86*]*85*) **where**
Σ_e *G f n* == *nsum G f n*

abbreviation
NPROD :: [(*'a*, *'m*) *Ring-scheme*, *nat*, *nat* ⇒ *'a*] ⇒ *'a*
((*3e*$\Pi_{\_,\_}$ -) [*98*,*98*,*99*]*98*) **where**
e$\Pi_{R,n}$ *f* == *nprod R f n*

definition
fSum :: [-, (*nat* => *'a*), *nat*, *nat*] ⇒ *'a* **where**
fSum A f n m = (*if n* ≤ *m then nsum A* (*cmp f* (*slide n*))(*m* − *n*)
else $\mathbf{0}_A$)

abbreviation
FSUM :: [(*'a*, *'more*) *aGroup-scheme*, (*nat* ⇒ *'a*), *nat*, *nat*] ⇒ *'a*
((*4*Σ_f - - - -) [*85*,*85*,*85*,*86*]*85*) **where**
Σ_f *G f n m* == *fSum G f n m*

lemma (**in** *aGroup*) *nsum-zeroGTr*:(∀ *j* ≤ *n*. *f j* = **0**) ⟶ *nsum A f n* = **0**
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-zeroA*:$\forall j \le n.\ f\ j = \mathbf{0} \Longrightarrow \quad nsum\ A\ f\ n = \mathbf{0}$
⟨*proof*⟩

definition
sr :: [- , *'a set*] ⇒ *bool* **where**
$sr\ R\ S == S \subseteq carrier\ R \wedge 1_{r\,R} \in S \wedge (\forall x{\in}S.\ \forall y \in S.\ x\ \pm_R (-_{a\,R}\ y) \in S \wedge$
$x \cdot_{r\,R} y \in S)$

definition
Sr :: [- , *'a set*] ⇒ - **where**
$Sr\ R\ S = R\ (\!|carrier := S,\ pop := \lambda x{\in}S.\ \lambda y{\in}S.\ x \pm_R y,\ mop := \lambda x{\in}S.\ (-_{a\,R}$
$x),$
$zero := \mathbf{0}_R,\ tp := \lambda x{\in}S.\ \lambda y{\in}S.\ x \cdot_{r\,R} y,\ un := 1_{r\,R}\ |\!)$

lemma (**in** *Ring*) *Ring*: *Ring R* ⟨*proof*⟩

lemma (**in** *Ring*) *ring-is-ag*:*aGroup R*
⟨*proof*⟩

lemma (**in** *Ring*) *ring-zero*:$\mathbf{0} \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-one*:$1_r \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-tOp-closed*:$[\![\ x \in carrier\ R;\ y \in carrier\ R]\!] \Longrightarrow$
$x \cdot_r y \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-tOp-commute*:$[\![x \in carrier\ R;\ y \in carrier\ R]\!] \Longrightarrow$
$x \cdot_r y = y \cdot_r x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib1*:$[\![x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R\ ]\!]$
$\Longrightarrow x \cdot_r (y \pm z) = x \cdot_r y \pm x \cdot_r z$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib2*:$[\![x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R\ ]\!]$
$\Longrightarrow (y \pm z) \cdot_r x = y \cdot_r x \pm\ z \cdot_r x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib3*:$[\![a \in carrier\ R;\ b \in carrier\ R;\ x \in carrier\ R;$
$y \in carrier\ R\ ]\!] \Longrightarrow (a \pm b) \cdot_r (x \pm y) =$
$a \cdot_r x \pm a \cdot_r y \pm b \cdot_r x \pm b \cdot_r y$
⟨*proof*⟩

lemma (**in** *Ring*) *rEQMulR*:
$[\![x \in \mathit{carrier}\ R;\ y \in \mathit{carrier}\ R;\ z \in \mathit{carrier}\ R;\ x = y\ ]\!]$
$\Longrightarrow x \cdot_r z = y \cdot_r z$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-tOp-assoc*:$[\![x \in \mathit{carrier}\ R;\ y \in \mathit{carrier}\ R;\ z \in \mathit{carrier}\ R\ ]\!]$
$\Longrightarrow (x \cdot_r y) \cdot_r z = x \cdot_r (y \cdot_r z)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-l-one*:$x \in \mathit{carrier}\ R \Longrightarrow 1_r \cdot_r x = x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-r-one*:$x \in \mathit{carrier}\ R \Longrightarrow x \cdot_r 1_r = x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-0-x*:$x \in \mathit{carrier}\ R \Longrightarrow \mathbf{0} \cdot_r x = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-x-0*:$x \in \mathit{carrier}\ R \Longrightarrow x \cdot_r \mathbf{0} = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *rMulZeroDiv*:
$[\![\ x \in \mathit{carrier}\ R;\ y \in \mathit{carrier}\ R;\ x = \mathbf{0} \vee y = \mathbf{0}\ ]\!] \Longrightarrow x \cdot_r y = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1*:$[\![\ a \in \mathit{carrier}\ R;\ b \in \mathit{carrier}\ R\ ]\!] \Longrightarrow$
$-_a (a \cdot_r b) = (-_a a) \cdot_r b \wedge -_a (a \cdot_r b) = a \cdot_r (-_a b)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1-1*:$[\![a \in \mathit{carrier}\ R;\ b \in \mathit{carrier}\ R\ ]\!] \Longrightarrow$
$-_a (a \cdot_r b) = (-_a a) \cdot_r b$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1-2*:$[\![\ a \in \mathit{carrier}\ R;\ b \in \mathit{carrier}\ R\ ]\!] \Longrightarrow$
$-_a (a \cdot_r b) = a \cdot_r (-_a b)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-minusl*:$a \in \mathit{carrier}\ R \Longrightarrow -_a a = (-_a 1_r) \cdot_r a$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-minusr*:$a \in \mathit{carrier}\ R \Longrightarrow -_a a = a \cdot_r (-_a 1_r)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1-3*:$[\![a \in \mathit{carrier}\ R;\ b \in \mathit{carrier}\ R]\!] \Longrightarrow$
$a \cdot_r b = (-_a a) \cdot_r (-_a b)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib4*:$[\![a \in \mathit{carrier}\ R;\ b \in \mathit{carrier}\ R;$

$x \in carrier\ R;\ y \in carrier\ R\ \rrbracket \Longrightarrow$
$a \cdot_r b \pm (-_a x \cdot_r y) = a \cdot_r (b \pm (-_a y)) \pm (a \pm (-_a x)) \cdot_r y$
$\langle proof \rangle$

lemma (**in** *Ring*) *rMulLC*:
$\llbracket x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R \rrbracket$
$\Longrightarrow x \cdot_r (y \cdot_r z) = y \cdot_r (x \cdot_r z)$
$\langle proof \rangle$

lemma (**in** *Ring*) *Zero-ring*:$1_r = \mathbf{0} \Longrightarrow zeroring\ R$
$\langle proof \rangle$

lemma (**in** *Ring*) *Zero-ring1*:$\neg\ (zeroring\ R) \Longrightarrow\ 1_r \neq \mathbf{0}$
$\langle proof \rangle$

lemma (**in** *Ring*) *Sr-one*:$sr\ R\ S \Longrightarrow 1_r \in S$
$\langle proof \rangle$

lemma (**in** *Ring*) *Sr-zero*:$sr\ R\ S \Longrightarrow \mathbf{0} \in S$
$\langle proof \rangle$

lemma (**in** *Ring*) *Sr-mOp-closed*:$\llbracket sr\ R\ S;\ x \in S \rrbracket \Longrightarrow -_a x \in S$
$\langle proof \rangle$

lemma (**in** *Ring*) *Sr-pOp-closed*:$\llbracket sr\ R\ S;\ x \in S;\ y \in S \rrbracket \Longrightarrow x \pm y \in S$
$\langle proof \rangle$

lemma (**in** *Ring*) *Sr-tOp-closed*:$\llbracket sr\ R\ S;\ x \in S;\ y \in S \rrbracket \Longrightarrow x \cdot_r y \in S$
$\langle proof \rangle$

lemma (**in** *Ring*) *Sr-ring*:$sr\ R\ S \Longrightarrow Ring\ (Sr\ R\ S)$
$\langle proof \rangle$

4.2 Calculation of elements

4.2.1 nscale

lemma (**in** *Ring*) *ring-tOp-rel*:$\llbracket x \in carrier\ R;\ xa \in carrier\ R;\ y \in carrier\ R;$
$ya \in carrier\ R\ \rrbracket \Longrightarrow (x \cdot_r xa) \cdot_r (y \cdot_r ya) = (x \cdot_r y) \cdot_r (xa \cdot_r ya)$
$\langle proof \rangle$

lemma (**in** *Ring*) *nsClose*:
$\bigwedge n.\ \llbracket\ x \in carrier\ R\ \rrbracket \Longrightarrow nscal\ R\ x\ n \in carrier\ R$
$\langle proof \rangle$

lemma (**in** *Ring*) *nsZero*:
$nscal\ R\ \mathbf{0}\ n = \mathbf{0}$
$\langle proof \rangle$

lemma (**in** *Ring*) *nsZeroI*: $\bigwedge n.\ \ x = \mathbf{0} \implies nscal\ R\ x\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *nsEqElm*: $[\![\ x \in carrier\ R;\ y \in carrier\ R;\ x = y\ ]\!]$
$\implies (nscal\ R\ x\ n) = (nscal\ R\ y\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *nsDistr*: $x \in carrier\ R$
$\implies (nscal\ R\ x\ n) \pm (nscal\ R\ x\ m) = nscal\ R\ x\ (n + m)$
⟨*proof*⟩

lemma (**in** *Ring*) *nsDistrL*: $[\![x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\implies (nscal\ R\ x\ n) \pm (nscal\ R\ y\ n) = nscal\ R\ (x \pm y)\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *nsMulDistrL*: $[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\implies x \cdot_r (nscal\ R\ y\ n) = nscal\ R\ (x \cdot_r y)\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *nsMulDistrR*: $[\![\ x \in carrier\ R;\ y \in carrier\ R]\!]$
$\implies (nscal\ R\ y\ n) \cdot_r x = nscal\ R\ (y \cdot_r x)\ n$
⟨*proof*⟩

4.2.2 npow

lemma (**in** *Ring*) *npClose*: $x \in carrier\ R \implies npow\ R\ x\ n \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *npMulDistr*: $\bigwedge n\ m.\ x \in carrier\ R \implies$
$(npow\ R\ x\ n) \cdot_r (npow\ R\ x\ m) = npow\ R\ x\ (n + m)$
⟨*proof*⟩

lemma (**in** *Ring*) *npMulExp*: $\bigwedge n\ m.\ x \in carrier\ R$
$\implies\ npow\ R\ (npow\ R\ x\ n)\ m = npow\ R\ x\ (n * m)$
⟨*proof*⟩

lemma (**in** *Ring*) *npGTPowZero-sub*:
$\bigwedge n.\ [\![\ x \in carrier\ R;\ npow\ R\ x\ m = \mathbf{0}\ ]\!]$
$\implies (m \le n) \longrightarrow (npow\ R\ x\ n = \mathbf{0}\ )$
⟨*proof*⟩

lemma (**in** *Ring*) *npGTPowZero*:
$\bigwedge n.\ [\![\ x \in carrier\ R;\ npow\ R\ x\ m = \mathbf{0};\ m \le n\ ]\!]$
$\implies npow\ R\ x\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *npOne*: $npow\ R\ (1_r)\ n = 1_r$

$\langle proof \rangle$

lemma (**in** *Ring*) *npZero-sub*: $0 < n \longrightarrow npow\ R\ \mathbf{0}\ n = \mathbf{0}$
$\langle proof \rangle$

lemma (**in** *Ring*) *npZero*: $0 < n \implies npow\ R\ \mathbf{0}\ n = \mathbf{0}$
$\langle proof \rangle$

lemma (**in** *Ring*) *npMulElmL*: $\bigwedge n.\ [\![\ x \in carrier\ R;\ 0 \leq n]\!]$
$\implies x \cdot_r (npow\ R\ x\ n) = npow\ R\ x\ (Suc\ n)$
$\langle proof \rangle$

lemma (**in** *Ring*) *npMulEleL*: $\bigwedge n.\ x \in carrier\ R$
$\implies (npow\ R\ x\ n) \cdot_r x = npow\ R\ x\ (Suc\ n)$
$\langle proof \rangle$

lemma (**in** *Ring*) *npMulElmR*: $\bigwedge n.\ x \in carrier\ R$
$\implies (npow\ R\ x\ n) \cdot_r x = npow\ R\ x\ (Suc\ n)$
$\langle proof \rangle$

lemma (**in** *Ring*) *np-1*: $a \in carrier\ R \implies npow\ R\ a\ (Suc\ 0) = a$
$\langle proof \rangle$

4.2.3 nsum and fSum

lemma (**in** *aGroup*) *nsum-memTr*: $(\forall j \leq n.\ f\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ f\ n \in carrier\ A$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nsum-mem*: $\forall j \leq n.\ f\ j \in carrier\ A \implies$
$nsum\ A\ f\ n \in carrier\ A$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nsum-eqTr*: $(\forall j \leq n.\ f\ j \in carrier\ A\ \wedge$
$g\ j \in carrier\ A\ \wedge$
$f\ j = g\ j)$
$\longrightarrow\ nsum\ A\ f\ n = nsum\ A\ g\ n$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nsum-eq*: $[\![\forall j \leq n.\ f\ j \in carrier\ A;\ \forall j \leq n.\ g\ j \in carrier\ A;$
$\forall j \leq n.\ f\ j = g\ j]\!] \implies nsum\ A\ f\ n = nsum\ A\ g\ n$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nsum-cmp-assoc*: $[\![\forall j \leq n.\ f\ j \in carrier\ A;$
$g \in \{j.\ j \leq n\} \rightarrow \{j.\ j \leq n\};\ h \in \{j.\ j \leq n\} \rightarrow \{j.\ j \leq n\}]\!] \implies$
$nsum\ A\ (cmp\ (cmp\ f\ h)\ g)\ n = nsum\ A\ (cmp\ f\ (cmp\ h\ g))\ n$
$\langle proof \rangle$

lemma (**in** *aGroup*) *fSum-Suc*: $\forall j \in nset\ n\ (n + Suc\ m).\ f\ j \in carrier\ A \implies$

$fSum\ A\ f\ n\ (n + Suc\ m) = fSum\ A\ f\ n\ (n + m) \pm f\ (n + Suc\ m)$
⟨proof⟩

lemma (**in** *aGroup*) *fSum-eqTr*:$(\forall j \in nset\ n\ (n + m).\ f\ j \in carrier\ A \wedge g\ j \in carrier\ A \wedge f\ j = g\ j) \longrightarrow fSum\ A\ f\ n\ (n + m) = fSum\ A\ g\ n\ (n + m)$
⟨proof⟩

lemma (**in** *aGroup*) *fSum-eq*:$[\![\forall j \in nset\ n\ (n + m).\ f\ j \in carrier\ A;\ \forall j \in nset\ n\ (n + m).\ g\ j \in carrier\ A;\ (\forall j\in nset\ n\ (n + m).\ f\ j = g\ j)]\!] \Longrightarrow fSum\ A\ f\ n\ (n + m) = fSum\ A\ g\ n\ (n + m)$
⟨proof⟩

lemma (**in** *aGroup*) *fSum-eq1*:$[\![n \leq m;\ \forall j\in nset\ n\ m.\ f\ j \in carrier\ A;\ \forall j\in nset\ n\ m.\ g\ j \in carrier\ A;\ \forall j\in nset\ n\ m.\ f\ j = g\ j]\!] \Longrightarrow fSum\ A\ f\ n\ m = fSum\ A\ g\ n\ m$
⟨proof⟩

lemma (**in** *aGroup*) *fSum-zeroTr*:$(\forall j \in nset\ n\ (n + m).\ f\ j = \mathbf{0}) \longrightarrow fSum\ A\ f\ n\ (n + m) = \mathbf{0}$
⟨proof⟩

lemma (**in** *aGroup*) *fSum-zero*:$\forall j \in nset\ n\ (n + m).\ f\ j = \mathbf{0} \Longrightarrow fSum\ A\ f\ n\ (n + m) = \mathbf{0}$
⟨proof⟩

lemma (**in** *aGroup*) *fSum-zero1*:$[\![n < m;\ \forall j \in nset\ (Suc\ n)\ m.\ f\ j = \mathbf{0}]\!] \Longrightarrow fSum\ A\ f\ (Suc\ n)\ m = \mathbf{0}$
⟨proof⟩

lemma (**in** *Ring*) *nsumMulEleL*: $\bigwedge n.\ [\![\forall\ i.\ f\ i \in carrier\ R;\ x \in carrier\ R\ ]\!] \Longrightarrow x \cdot_r (nsum\ R\ f\ n) = nsum\ R\ (\lambda\ i.\ x \cdot_r (f\ i))\ n$
⟨proof⟩

lemma (**in** *Ring*) *nsumMulElmL*:
$\bigwedge n.\ [\![\forall\ i.\ f\ i \in carrier\ R;\ x \in carrier\ R\ ]\!] \Longrightarrow x \cdot_r (nsum\ R\ f\ n) = nsum\ R\ (\lambda\ i.\ x \cdot_r (f\ i))\ n$
⟨proof⟩

lemma (**in** *aGroup*) *nsumTailTr*:
$(\forall j\leq(Suc\ n).\ f\ j \in carrier\ A) \longrightarrow nsum\ A\ f\ (Suc\ n) = (nsum\ A\ (\lambda\ i.\ (f\ (Suc\ i)))\ n) \pm (f\ 0)$
⟨proof⟩

lemma (**in** *aGroup*) *nsumTail*:
$\forall j \leq (Suc\ n).\ f\ j \in carrier\ A \Longrightarrow nsum\ A\ f\ (Suc\ n) = (nsum\ A\ (\lambda\ i.\ (f\ (Suc\ i)))\ n) \pm (f\ 0)$
⟨proof⟩

lemma (**in** *aGroup*) *nsumElmTail*:
$\forall i.\ f\ i \in carrier\ A$
$\Longrightarrow nsum\ A\ f\ (Suc\ n) = (nsum\ A\ (\lambda\ i.\ (f\ (Suc\ i)))\ n) \pm (f\ 0)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-addTr*:
$(\forall j \le n.\ f\ j \in carrier\ A \wedge g\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ (\lambda\ i.\ (f\ i) \pm (g\ i))\ n = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-add*:
$[\![\ \forall j \le n.\ f\ j \in carrier\ A;\ \forall j \le n.\ g\ j \in carrier\ A]\!] \Longrightarrow$
$nsum\ A\ (\lambda\ i.\ (f\ i) \pm (g\ i))\ n = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsumElmAdd*:
$[\![\ \forall\ i.\ f\ i \in carrier\ A;\ \forall\ i.\ g\ i \in carrier\ A]\!]$
$\Longrightarrow nsum\ A\ (\lambda\ i.\ (f\ i) \pm (g\ i))\ n = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-add-nmTr*:
$(\forall j \le n.\ f\ j \in carrier\ A) \wedge (\forall j \le m.\ g\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ (jointfun\ n\ f\ m\ g)\ (Suc\ (n + m)) = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ m)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-add-nm*:
$[\![\forall j \le n.\ f\ j \in carrier\ A;\ \forall j \le m.\ g\ j \in carrier\ A]\!] \Longrightarrow$
$nsum\ A\ (jointfun\ n\ f\ m\ g)\ (Suc\ (n + m)) = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ m)$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2-sub-muly*:
$[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!] \Longrightarrow$
$y \cdot_r (nsum\ R\ (\lambda i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(n\ choose\ i))\ n)$
$= nsum\ R\ (\lambda i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ (i+1)))$
$(n\ choose\ i))\ n$
⟨*proof*⟩

lemma *binomial-n0*: $(Suc\ n\ choose\ 0) = (n\ choose\ 0)$
⟨*proof*⟩

lemma *binomial-ngt-diff*:
$(n\ choose\ Suc\ n) = (Suc\ n\ choose\ Suc\ n) - (n\ choose\ n)$
⟨*proof*⟩

lemma *binomial-ngt-0*: $(n\ choose\ Suc\ n) = 0$

⟨*proof*⟩

lemma *diffLessSuc*: $m \leq n \Longrightarrow Suc\ (n-m) = Suc\ n\ -\ m$
⟨*proof*⟩

lemma (**in** *Ring*) *npow-suc-i*:
$[\![\ x \in carrier\ R;\ i \leq n\ ]\!]$
$\Longrightarrow npow\ R\ x\ (Suc\ n\ -\ i) =\ x \cdot_r (npow\ R\ x\ (n-i))$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2-sub-mulx*: $[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!] \Longrightarrow$
$x \cdot_r (nsum\ R\ (\lambda\ i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(n\ choose\ i))\ n)$
$= (nsum\ R\ (\lambda i.\ nscal\ R$
$((npow\ R\ x\ (Suc\ n\ -\ Suc\ i)) \cdot_r (npow\ R\ y\ (Suc\ i)))$
$(n\ choose\ Suc\ i))\ n) \pm$
$(nscal\ R\ ((npow\ R\ x\ (Suc\ n\ -\ 0)) \cdot_r (npow\ R\ y\ 0))$
$(Suc\ n\ choose\ 0))$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2-sub-mulx2*:
$[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!] \Longrightarrow$
$x \cdot_r (nsum\ R\ (\lambda\ i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(n\ choose\ i))\ n)$
$= (nsum\ R\ \ (\lambda i.\ nscal\ R$
$((npow\ R\ x\ (n\ -\ i)) \cdot_r ((npow\ R\ y\ i) \cdot_r\ y\ ))$
$(n\ choose\ Suc\ i))\ n) \pm$
$(\mathbf{0} \pm ((x \cdot_r (npow\ R\ x\ n)) \cdot_r (1_r)))$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2*:
$\bigwedge n.\ [\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\Longrightarrow npow\ R\ (x \pm y)\ n =$
$nsum\ R\ (\lambda\ i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(\ n\ choose\ i)\ )\ n$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-zeroTr*:
$\bigwedge n.\ (\forall\ i.\ i \leq n \longrightarrow\ f\ i = \mathbf{0}) \longrightarrow (nsum\ A\ f\ n = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *Ring*) *npAdd*:
$[\![\ x \in carrier\ R;\ y \in carrier\ R;$
$npow\ R\ x\ m = \mathbf{0};\ npow\ R\ y\ n = \mathbf{0}\ ]\!]$
$\Longrightarrow npow\ R\ (x \pm y)\ (m + n) = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *npInverse*:
$\bigwedge n.\ x \in carrier\ R$
$\Longrightarrow npow\ R\ (-_a\ x)\ n = npow\ R\ x\ n$
$\vee\ npow\ R\ (-_a\ x)\ n = -_a\ (npow\ R\ x\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *npMul*:
$\bigwedge\ n.\ [\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\Longrightarrow npow\ R\ (x \cdot_r\ y)\ n = (npow\ R\ x\ n) \cdot_r\ (npow\ R\ y\ n)$
⟨*proof*⟩

4.3 Ring homomorphisms

definition
rHom :: [('a, 'm) *Ring-scheme*, ('b, 'm1) *Ring-scheme*]
⇒ ('a ⇒ 'b) *set* **where**
$rHom\ A\ R = \{f.\ f \in aHom\ A\ R\ \wedge$
$(\forall x{\in}carrier\ A.\ \forall y{\in}carrier\ A.\ f\ (\ x \cdot_{r\,A}\ y) = (f\ x) \cdot_{r\,R}\ (f\ y))$
$\wedge\ f\ (1_{r\,A}) = (1_{r\,R})\}$

definition
rInvim :: [('a, 'm) *Ring-scheme*, ('b, 'm1) *Ring-scheme*, 'a ⇒ 'b, 'b *set*]
⇒ 'a *set* **where**
$rInvim\ A\ R\ f\ K = \{a.\ a \in carrier\ A \wedge f\ a \in K\}$

definition
rimg :: [('a, 'm) *Ring-scheme*, ('b, 'm1) *Ring-scheme*, 'a ⇒ 'b] ⇒
'b *Ring* **where**
rimg A R f = (|*carrier*= *f* ‘(*carrier A*), *pop* = *pop R*, *mop* = *mop R*,
zero = *zero R*, *tp* = *tp R*, *un* = *un R* |)

definition
ridmap :: ('a, 'm) *Ring-scheme* ⇒ ('a ⇒ 'a) **where**
$ridmap\ R = (\lambda x{\in}carrier\ R.\ x)$

definition
r-isom :: [('a, 'm) *Ring-scheme*, ('b, 'm1) *Ring-scheme*] ⇒ *bool*
(**infixr** $\cong_r$ *100*) **where**
$r\text{-}isom\ R\ R' \longleftrightarrow (\exists f{\in}rHom\ R\ R'.\ bijec_{R,R'}\ f)$

definition
Subring :: [('a, 'm) *Ring-scheme*, ('a, 'm1) *Ring-scheme*] ⇒ *bool* **where**
$Subring\ R\ S == Ring\ S \wedge (carrier\ S \subseteq carrier\ R) \wedge (ridmap\ S) \in rHom\ S\ R$

lemma *ridmap-surjec*:$Ring\ A \Longrightarrow surjec_{A,A}\ (ridmap\ A)$
⟨*proof*⟩

lemma *rHom-aHom*:$f \in rHom\ A\ R \Longrightarrow f \in aHom\ A\ R$

⟨*proof*⟩

lemma *rimg-carrier*:$f \in rHom\ A\ R \Longrightarrow carrier\ (rimg\ A\ R\ f) = f\ `\ (carrier\ A)$
⟨*proof*⟩

lemma *rHom-mem*:$[\![\ f \in rHom\ A\ R;\ a \in carrier\ A\ ]\!] \Longrightarrow f\ a \in carrier\ R$
⟨*proof*⟩

lemma *rHom-func*:$f \in rHom\ A\ R \Longrightarrow f \in carrier\ A \rightarrow carrier\ R$
⟨*proof*⟩

lemma *ringhom1*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ y \in carrier\ A;$
$f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (x \pm_A\ y) = (f\ x) \pm_R (f\ y)$
⟨*proof*⟩

lemma *rHom-inv-inv*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ f \in rHom\ A\ R\ ]\!]$
$\Longrightarrow f\ (-_{a A}\ x) = -_{a R}\ (f\ x)$
⟨*proof*⟩

lemma *rHom-0-0*:$[\![\ Ring\ A;\ Ring\ R;\ f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (\mathbf{0}_A) = \mathbf{0}_R$
⟨*proof*⟩

lemma *rHom-tOp*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ y \in carrier\ A;$
$f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (x \cdot_{r A}\ y) = (f\ x) \cdot_{r R} (f\ y)$
⟨*proof*⟩

lemma *rHom-add*:$[\![f \in rHom\ A\ R;\ x \in carrier\ A;\ y \in carrier\ A]\!] \Longrightarrow$
$f\ (x \pm_A\ y) = (f\ x) \pm_R (f\ y)$
⟨*proof*⟩

lemma *rHom-one*:$[\![\ Ring\ A;\ Ring\ R;f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (1_{r A}) = (1_{r R})$
⟨*proof*⟩

lemma *rHom-npow*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ f \in rHom\ A\ R\ ]\!] \Longrightarrow$
$f\ (x^{\hat{}A\ n}) = (f\ x)^{\hat{}R\ n}$
⟨*proof*⟩

lemma *rHom-compos*:$[\![Ring\ A;\ Ring\ B;\ Ring\ C;\ f \in rHom\ A\ B;\ g \in rHom\ B\ C]\!]$
$\Longrightarrow$
$compos\ A\ g\ f \in rHom\ A\ C$
⟨*proof*⟩

lemma *rimg-ag*:$[\![Ring\ A;\ Ring\ R;\ f \in rHom\ A\ R]\!] \Longrightarrow aGroup\ (rimg\ A\ R\ f)$
⟨*proof*⟩

lemma *rimg-ring*:$[\![Ring\ A;\ Ring\ R;\ f \in rHom\ A\ R\ ]\!] \Longrightarrow Ring\ (rimg\ A\ R\ f)$
⟨*proof*⟩

definition

$ideal :: [\text{-} ,\ 'a\ set] \Rightarrow bool$ **where**
$ideal\ R\ I \longleftrightarrow (R +> I) \wedge (\forall r \in carrier\ R.\ \forall x \in I.\ (r \cdot_{r R}\ x \in I))$

lemma (**in** *Ring*) $ideal\text{-}asubg{:}ideal\ R\ I \Longrightarrow R +> I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}pOp\text{-}closed{:}[\![ideal\ R\ I;\ x \in I;\ y \in I\ ]\!] \Longrightarrow x \pm y \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}nsum\text{-}closedTr{:}ideal\ R\ I \Longrightarrow (\forall j \le n.\ f\ j \in I) \longrightarrow\ nsum\ R\ f\ n \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}nsum\text{-}closed{:}[\![ideal\ R\ I;\ \forall j \le n.\ f\ j \in I]\!] \Longrightarrow nsum\ R\ f\ n \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}subset1{:}ideal\ R\ I \Longrightarrow I \subseteq carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}subset{:}[\![ideal\ R\ I;\ h \in I]\!] \Longrightarrow h \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}ring\text{-}multiple{:}[\![ideal\ R\ I;\ x \in I;\ r \in carrier\ R]\!] \Longrightarrow r \cdot_r x \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}ring\text{-}multiple1{:}[\![ideal\ R\ I;\ x \in I;\ r \in carrier\ R\ ]\!] \Longrightarrow x \cdot_r r \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}npow\text{-}closedTr{:}[\![ideal\ R\ I;\ x \in I]\!] \Longrightarrow 0 < n \longrightarrow x^{\hat{}R\ n} \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}npow\text{-}closed{:}[\![ideal\ R\ I;\ x \in I;\ 0 < n]\!] \Longrightarrow x^{\hat{}R\ n} \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $times\text{-}modTr{:}[\![a \in carrier\ R;\ a' \in carrier\ R;\ b \in carrier\ R;\ b' \in carrier\ R;\ ideal\ R\ I;\ a \pm (-_a\ b) \in I;\ a' \pm (-_a\ b') \in I]\!] \Longrightarrow a \cdot_r a' \pm (-_a\ (b \cdot_r b')) \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}inv1\text{-}closed{:}[\![\ ideal\ R\ I;\ x \in I\ ]\!] \Longrightarrow -_a\ x \in I$
⟨*proof*⟩

lemma (**in** *Ring*) $ideal\text{-}zero{:}ideal\ R\ I\ \Longrightarrow \mathbf{0} \in I$

⟨*proof*⟩

lemma (**in** *Ring*) *ideal-zero-forall*:$\forall I.\ ideal\ R\ I \longrightarrow\ \mathbf{0} \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-ele-sumTr1*:$[\![$ *ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*;
$a \pm b \in I;\ a \in I\ ]\!] \Longrightarrow b \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-ele-sumTr2*:$[\![$*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*;
$a \pm b \in I;\ b \in I]\!] \Longrightarrow a \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-condition*:$[\![I \subseteq$ *carrier R*; $I \neq \{\}$;
$\forall x \in I.\ \forall y \in I.\ x \pm (-_a\ y) \in I;\ \forall r \in carrier\ R.\ \forall x \in I.\ r \cdot_r x \in I\ ]\!] \Longrightarrow$
ideal R I
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-condition1*:$[\![I \subseteq$ *carrier R*; $I \neq \{\}$;
$\forall x \in I.\ \forall y \in I.\ x \pm y \in I;\ \forall r \in carrier\ R.\ \forall x \in I.\ r \cdot_r x \in I\ ]\!] \Longrightarrow$ *ideal R I*
⟨*proof*⟩

lemma (**in** *Ring*) *zero-ideal*:*ideal R* $\{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Ring*) *whole-ideal*:*ideal R* (*carrier R*)
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-one*:$[\![$*ideal R I*; $1_r \in I\ ]\!] \Longrightarrow I =$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-one1*:*ideal R I* $\Longrightarrow$
$(1_r \in I) = (I = carrier\ R)$
⟨*proof*⟩

definition
Unit :: - $\Rightarrow$ *'a* $\Rightarrow$ *bool* **where**
Unit R a $\longleftrightarrow a \in$ *carrier R* $\wedge$ $(\exists b \in carrier\ R.\ a \cdot_{r R}\ b = 1_{r R})$

lemma (**in** *Ring*) *ideal-inc-unit*:$[\![$*ideal R I*; $a \in I$; *Unit R a*$]\!] \Longrightarrow 1_r \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *proper-ideal*:$[\![$*ideal R I*; $1_r \notin I]\!] \Longrightarrow I \neq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-unit1*:$[\![a \in$ *carrier R*; *Unit R a*; *ideal R I*; $a \in I]\!]$
$\Longrightarrow I =$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *int-ideal*:$[\![$*ideal R I*; *ideal R J*$]\!] \Longrightarrow$ *ideal R* $(I \cap J)$
$\langle proof \rangle$

definition
ideal-prod::[-, *'a set*, *'a set*] $\Rightarrow$ *'a set* (**infix** $\diamondsuit_{r1}$ *90*) **where**
ideal-prod R I J $== \bigcap$ {*L*. *ideal R L* $\wedge$
$\{x.(\exists i{\in}I.\ \exists j{\in}J.\ x = i \cdot_{r R} j)\} \subseteq L\}$

lemma (**in** *Ring*) *set-sum-mem*:$[\![a \in I;\ b \in J;\ I \subseteq$ *carrier R*; $J \subseteq$ *carrier R*$]\!] \Longrightarrow$
$a \pm b \in I \mp J$
$\langle proof \rangle$

lemma (**in** *Ring*) *sum-ideals*:$[\![$*ideal R I1*; *ideal R I2*$]\!] \Longrightarrow$ *ideal R* $(I1 \mp I2)$
$\langle proof \rangle$

lemma (**in** *Ring*) *sum-ideals-la1*:$[\![$*ideal R I1*; *ideal R I2*$]\!] \Longrightarrow I1 \subseteq (I1 \mp I2)$
$\langle proof \rangle$

lemma (**in** *Ring*) *sum-ideals-la2*:$[\![$*ideal R I1*; *ideal R I2* $]\!] \Longrightarrow I2 \subseteq (I1 \mp I2)$
$\langle proof \rangle$

lemma (**in** *Ring*) *sum-ideals-cont*:$[\![$*ideal R I*; $A \subseteq I$; $B \subseteq I$ $]\!] \Longrightarrow A \mp B \subseteq I$
$\langle proof \rangle$

lemma (**in** *Ring*) *ideals-set-sum*:$[\![$*ideal R A*; *ideal R B*; $x \in A \mp B]\!] \Longrightarrow$
$\exists h{\in}A.\ \exists k{\in}B.\ x = h \pm k$
$\langle proof \rangle$

definition
Rxa :: [-, *'a*] $\Rightarrow$ *'a set* (**infixl** $\diamondsuit_p$ *200*) **where**
Rxa R a = $\{x.\ \exists r{\in}$*carrier R*. $x = (r \cdot_{r R} a)\}$

lemma (**in** *Ring*) *a-in-principal*:$a \in$ *carrier R* $\Longrightarrow a \in$ *Rxa R a*
$\langle proof \rangle$

lemma (**in** *Ring*) *principal-ideal*:$a \in$ *carrier R* $\Longrightarrow$ *ideal R* (*Rxa R a*)
$\langle proof \rangle$

lemma (**in** *Ring*) *rxa-in-Rxa*:$[\![a \in$ *carrier R*; $r \in$ *carrier R*$]\!] \Longrightarrow$
$r \cdot_r a \in$ *Rxa R a*
$\langle proof \rangle$

lemma (**in** *Ring*) *Rxa-one*:*Rxa R* 1_r = *carrier R*
$\langle proof \rangle$

lemma (**in** *Ring*) *Rxa-zero*:*Rxa R* $\mathbf{0} = \{\mathbf{0}\}$
$\langle proof \rangle$

lemma (**in** *Ring*) *Rxa-nonzero*:$[\![a \in carrier\ R;\ a \neq \mathbf{0}]\!] \Longrightarrow Rxa\ R\ a \neq \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-cont-Rxa*:$[\![ideal\ R\ I;\ a \in I]\!] \Longrightarrow Rxa\ R\ a \subseteq I$
⟨*proof*⟩

lemma (**in** *Ring*) *Rxa-mult-smaller*:$[\![\ a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow$
$Rxa\ R\ (a \cdot_r\ b) \subseteq Rxa\ R\ b$
⟨*proof*⟩

lemma (**in** *Ring*) *id-ideal-psub-sum*:$[\![ideal\ R\ I;\ a \in carrier\ R;\ a \notin I]\!] \Longrightarrow$
$I \subset I \mp Rxa\ R\ a$
⟨*proof*⟩

lemma (**in** *Ring*) *mul-two-principal-idealsTr*:$[\![a \in carrier\ R;\ b \in carrier\ R;$
$x \in Rxa\ R\ a;\ y \in Rxa\ R\ b]\!] \Longrightarrow \exists\, r{\in}carrier\ R.\ x \cdot_r\ y = r \cdot_r\ (a \cdot_r\ b)$
⟨*proof*⟩

primrec *sum-pr-ideals*::$[('a,\ 'm)\ Ring\text{-}scheme,\ nat \Rightarrow 'a,\ nat] \Rightarrow 'a\ set$
where
sum-pr0: $sum\text{-}pr\text{-}ideals\ R\ f\ 0 = Rxa\ R\ (f\ 0)$
| *sum-prn*: $sum\text{-}pr\text{-}ideals\ R\ f\ (Suc\ n) =$
$(Rxa\ R\ (f\ (Suc\ n))) \mp_R (sum\text{-}pr\text{-}ideals\ R\ f\ n)$

lemma (**in** *Ring*) *sum-of-prideals0*:
$\forall f.\ (\forall\, l \leq n.\ f\ l \in carrier\ R) \longrightarrow ideal\ R\ (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals*:$[\![\forall\, l \leq n.\ f\ l \in carrier\ R]\!] \Longrightarrow$
$ideal\ R\ (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

later, we show *sum-pr-ideals* is the least ideal containing $\{f\ 0,\ f\ 1, \ldots,\ f\ n\}$

lemma (**in** *Ring*) *sum-of-prideals1*:$\forall f.\ (\forall\, l \leq n.\ f\ l \in carrier\ R) \longrightarrow$
$f\ `\ \{i.\ i \leq n\} \subseteq (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals2*:$\forall\, l \leq n.\ f\ l \in carrier\ R$
$\Longrightarrow\ f\ `\ \{i.\ i \leq n\} \subseteq (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals3*:$ideal\ R\ I \Longrightarrow$
$\forall f.\ (\forall\, l \leq n.\ f\ l \in carrier\ R) \wedge (f\ `\ \{i.\ i \leq n\} \subseteq I) \longrightarrow$
$(sum\text{-}pr\text{-}ideals\ R\ f\ n \subseteq I)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals4*:$[\![ideal\ R\ I;\ \forall\, l \leq n.\ f\ l \in carrier\ R;$

$(f \text{ ` } \{i.\ i \leq n\} \subseteq I)\rrbracket \Longrightarrow \textit{sum-pr-ideals } R\ f\ n \subseteq I$
⟨proof⟩

lemma *ker-ideal*:$\llbracket Ring\ A;\ Ring\ R;\ f \in rHom\ A\ R\rrbracket \Longrightarrow ideal\ A\ (ker_{A,R}\ f)$
⟨proof⟩

4.3.1 Ring of integers

definition
$Zr :: int\ Ring$ **where**
$Zr = (\!|\ carrier = Zset,\ pop = \lambda n{\in}Zset.\ \lambda m{\in}Zset.\ (m + n),$
$mop = \lambda l{\in}Zset.\ -l,\ zero = 0,\ tp = \lambda m{\in}Zset.\ \lambda n{\in}Zset.\ m * n,\ un = 1\ |\!)$

lemma *ring-of-integers*:*Ring Zr*
⟨proof⟩

lemma *Zr-zero*:$\mathbf{0}_{Zr} = 0$
⟨proof⟩

lemma *Zr-one*:$1_{r\,Zr} = 1$
⟨proof⟩

lemma *Zr-minus*:$-_{a\,Zr}\ n = -\ n$
⟨proof⟩

lemma *Zr-add*:$n \pm_{Zr} m = n + m$
⟨proof⟩

lemma *Zr-times*:$n \cdot_{r\,Zr} m = n * m$
⟨proof⟩

definition
$lev :: int\ set \Rightarrow int$ **where**
$lev\ I = Zleast\ \{n.\ n \in I \wedge 0 < n\}$

lemma *Zr-gen-Zleast*:$\llbracket ideal\ Zr\ I;\ I \neq \{0{::}int\}\rrbracket \Longrightarrow$
$Rxa\ Zr\ (lev\ I) = I$
⟨proof⟩

lemma *Zr-pir*:$ideal\ Zr\ I \Longrightarrow \exists\, n.\ Rxa\ Zr\ n = I$
⟨proof⟩

4.4 Quotient rings

lemma (**in** *Ring*) *mem-set-ar-cos*:$\llbracket ideal\ R\ I;\ a \in carrier\ R\rrbracket \Longrightarrow$
$a \uplus_R I \in set\text{-}ar\text{-}cos\ R\ I$
⟨proof⟩

lemma (**in** *Ring*) *I-in-set-ar-cos*:$ideal\ R\ I \Longrightarrow I \in set\text{-}ar\text{-}cos\ R\ I$

⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same1*:$[\![$*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*; $b \pm (-_a\ a) \in I\ ]\!] \Longrightarrow a \uplus_R I = b \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same2*:$[\![$*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*; $a \uplus_R I = b \uplus_R I]\!] \Longrightarrow\ b \pm (-_a\ a) \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same3*:$[\![$*ideal R I*; $a \in$ *carrier R*; $a \uplus_R I = I]\!] \Longrightarrow$ $a{\in}I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same3-1*:$[\![$*ideal R I*; $a \in$ *carrier R*; $a \notin I]\!] \Longrightarrow$ $a \uplus_R I \neq I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same4*:$[\![$*ideal R I*; $a \in I]\!] \Longrightarrow$ $a \uplus_R I = I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same4-1*:$[\![$*ideal R I*; $a \uplus_R I \neq I]\!] \Longrightarrow a \notin I$
⟨*proof*⟩

lemma (**in** *Ring*) *belong-ar-coset1*:$[\![$*ideal R I*; $a \in$ *carrier R*; $x \in$ *carrier R*; $x \pm (-_a\ a) \in I]\!] \Longrightarrow\ x \in a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *a-in-ar-coset*:$[\![$*ideal R I*; $a \in$ *carrier R*$]\!] \Longrightarrow a \in a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-subsetD*:$[\![$*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I\ ]\!] \Longrightarrow$ $x \in$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *ar-cos-mem*:$[\![$*ideal R I*; $a \in$ *carrier R*$]\!] \Longrightarrow$ $a \uplus_R I \in$ *set-rcs* (*b-ag R*) *I*
⟨*proof*⟩

lemma (**in** *Ring*) *mem-ar-coset1*:$[\![$*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I]\!] \Longrightarrow$ $\exists\, h{\in}I.\ h \pm a = x$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-mem2*:$[\![$*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I]\!] \Longrightarrow$ $\exists\, h{\in}I.\ x = a \pm h$
⟨*proof*⟩

lemma (**in** *Ring*) *belong-ar-coset2*:$[\![$*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I\ ]\!]$

$\Longrightarrow x \pm (-_a\ a) \in I$
⟨proof⟩

lemma (**in** *Ring*) *ar-c-top*: $[\![ideal\ R\ I;\ a \in carrier\ R;\ b \in carrier\ R]\!]$
$\Longrightarrow (c\text{-}top\ (b\text{-}ag\ R)\ I\ (a \uplus_R I)\ (b \uplus_R I)) = (a \pm b) \uplus_R I$
⟨proof⟩

Following lemma is not necessary to define a quotient ring. But it makes clear that the binary operation2 of the quotient ring is well defined.

lemma (**in** *Ring*) *quotient-ring-tr1*: $[\![ideal\ R\ I;\ a1 \in carrier\ R;\ a2 \in carrier\ R;$
$b1 \in carrier\ R;\ b2 \in carrier\ R;$
$a1 \uplus_R I = a2 \uplus_R I;\ b1 \uplus_R I = b2 \uplus_R I]\!] \Longrightarrow$
$(a1 \cdot_r b1) \uplus_R I = (a2 \cdot_r b2) \uplus_R I$
⟨proof⟩

definition
rcostOp :: [-, 'a set] ⇒ (['a set, 'a set] ⇒ 'a set) **where**
$rcostOp\ R\ I = (\lambda X{\in}(set\text{-}rcs\ (b\text{-}ag\ R)\ I).\ \lambda Y{\in}(set\text{-}rcs\ (b\text{-}ag\ R)\ I).$
$\{z.\ \exists\ x \in X.\ \exists\ y \in Y.\ \exists h{\in}I.\ (x \cdot_{rR}\ y) \pm_R h = z\})$

lemma (**in** *Ring*) *rcostOp*: $[\![ideal\ R\ I;\ a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow$
$rcostOp\ R\ I\ (a \uplus_R I)\ (b \uplus_R I) = (a \cdot_r b) \uplus_R I$
⟨proof⟩

definition
qring :: [('a, 'm) Ring-scheme, 'a set] ⇒ (| carrier :: 'a set set,
pop :: ['a set, 'a set] ⇒ 'a set, mop :: 'a set ⇒ 'a set,
zero :: 'a set, tp :: ['a set, 'a set] ⇒ 'a set, un :: 'a set |) **where**
qring R I = (| carrier = set-rcs (b-ag R) I,
pop = c-top (b-ag R) I,
mop = c-iop (b-ag R) I,
zero = I,
tp = rcostOp R I,
$un = 1_{rR} \uplus_R I$|)

abbreviation
QRING (**infixl** $'/_r$ *200*) **where**
$R\ /_r\ I == qring\ R\ I$

lemma (**in** *Ring*) *carrier-qring*: *ideal R I* ⟹
carrier (*qring R I*) = *set-rcs* (*b-ag R*) *I*
⟨proof⟩

lemma (**in** *Ring*) *carrier-qring1*: *ideal R I* ⟹
carrier (*qring R I*) = *set-ar-cos R I*
⟨proof⟩

lemma (**in** *Ring*) *qring-ring*: *ideal R I* ⟹ *Ring* (*qring R I*)
⟨proof⟩

lemma (**in** *Ring*) *qring-carrier*:*ideal R I* $\Longrightarrow$
carrier (*qring R I*) $= \{X.\ \exists\, a \in$ *carrier R*. $a \uplus_R I = X\}$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-mem*:⟦*ideal R I*; $a \in$ *carrier R*⟧ $\Longrightarrow$
$a \uplus_R I \in$ *carrier* (*qring R I*)
⟨*proof*⟩

lemma (**in** *Ring*) *qring-pOp*:⟦*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R* ⟧
$\Longrightarrow$ *pop* (*qring R I*) $(a \uplus_R I)$ $(b \uplus_R I) = (a \pm b) \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-zero*:*ideal R I* $\Longrightarrow$ *zero* (*qring R I*) $= I$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-zero-1*:⟦$a \in$ *carrier R*; *ideal R I*; $a \uplus_R I = I$⟧ $\Longrightarrow$
$a \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *Qring-fix1*:⟦$a \in$ *carrier R*; *ideal R I*; $a \in I$⟧ $\Longrightarrow$ $a \uplus_R I = I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-cos-same*:⟦$a \in$ *carrier R*; *ideal R I*; $x \in a \uplus_R I$⟧ $\Longrightarrow$
$x \uplus_R I = a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-tOp*:⟦*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*⟧ $\Longrightarrow$
tp (*qring R I*) $(a \uplus_R I)$ $(b \uplus_R I) = (a \cdot_r b) \uplus_R I$
⟨*proof*⟩

lemma *rind-hom-well-def*:⟦*Ring A*; *Ring R*; $f \in$ *rHom A R*; $a \in$ *carrier A* ⟧ $\Longrightarrow$
$f\ a = (f^{\circ}{}_{A,R})$ $(a \uplus_A (ker_{A,R}\ f))$
⟨*proof*⟩

lemma (**in** *Ring*) *set-r-ar-cos*:*ideal R I* $\Longrightarrow$
set-rcs (*b-ag R*) $I =$ *set-ar-cos R I*
⟨*proof*⟩

lemma *set-r-ar-cos-ker*:⟦*Ring A*; *Ring R*; $f \in$ *rHom A R* ⟧ $\Longrightarrow$
set-rcs (*b-ag A*) $(ker_{A,R}\ f) =$ *set-ar-cos A* $(ker_{A,R}\ f)$
⟨*proof*⟩

lemma *ind-hom-rhom*:⟦*Ring A*; *Ring R*; $f \in$ *rHom A R*⟧ $\Longrightarrow$
$(f^{\circ}{}_{A,R}) \in$ *rHom* (*qring A* $(ker_{A,R}\ f)$) *R*
⟨*proof*⟩

lemma *ind-hom-injec*:⟦*Ring A*; *Ring R*; $f \in$ *rHom A R*⟧ $\Longrightarrow$
$injec_{(qring\ A\ (ker_{A,R}\ f)),R}\ (f^{\circ}{}_{A,R})$

⟨*proof*⟩

lemma *rhom-to-rimg*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$⟧ ⟹
$f \in rHom\ A\ (rimg\ A\ R\ f)$
⟨*proof*⟩

lemma *ker-to-rimg*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$ ⟧ ⟹
$ker_{A,R}\ f = ker_{A,(rimg\ A\ R\ f)}\ f$
⟨*proof*⟩

lemma *indhom-eq*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$⟧ ⟹ $f^{\circ}{}_{A,(rimg\ A\ R\ f)} = f^{\circ}{}_{A,R}$
⟨*proof*⟩

lemma *indhom-bijec2-rimg*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$⟧ ⟹
$bijec_{(qring\ A\ (ker_{A,R}\ f)),(rimg\ A\ R\ f)}\ (f^{\circ}{}_{A,R})$
⟨*proof*⟩

lemma *surjec-ind-bijec*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$; $surjec_{A,R}\ f$⟧ ⟹
$bijec_{(qring\ A\ (ker_{A,R}\ f)),R}\ (f^{\circ}{}_{A,R})$
⟨*proof*⟩

lemma *ridmap-ind-bijec*:*Ring A* ⟹
$bijec_{(qring\ A\ (ker_{A,A}\ (ridmap\ A))),A}\ ((ridmap\ A)^{\circ}{}_{A,A})$
⟨*proof*⟩

lemma *ker-of-idmap*:*Ring A* ⟹ $ker_{A,A}\ (ridmap\ A) = \{\mathbf{0}_A\}$
⟨*proof*⟩

lemma *ring-natural-isom*:*Ring A* ⟹
$bijec_{(qring\ A\ \{\mathbf{0}_A\}),A}\ ((ridmap\ A)^{\circ}{}_{A,A})$
⟨*proof*⟩

definition
pj :: [(*'a*, *'m*) *Ring-scheme*, *'a set*] ⇒ (*'a* => *'a set*) **where**
pj R I = (λ*x*. *Pj* (*b-ag R*) *I x*)

lemma *pj-Hom*:⟦*Ring R*; *ideal R I*⟧ ⟹ (*pj R I*) ∈ *rHom R* (*qring R I*)
⟨*proof*⟩

lemma *pj-mem*:⟦*Ring R*; *ideal R I*; *x* ∈ *carrier R*⟧ ⟹ $pj\ R\ I\ x = x \uplus_R I$
⟨*proof*⟩

lemma *pj-zero*:⟦*Ring R*; *ideal R I*; *x* ∈ *carrier R*⟧ ⟹
$(pj\ R\ I\ x = \mathbf{0}_{(R\ /_r\ I)}) = (x \in I)$

⟨proof⟩

lemma *pj-surj-to*:⟦*Ring R*; *ideal R J*; $X \in$ *carrier* $(R /_r J)$⟧ $\Longrightarrow$
$\exists r\in$ *carrier R*. *pj R J r* $= X$
⟨proof⟩

lemma *invim-of-ideal*:⟦*Ring R*; *ideal R I*; *ideal* (*qring R I*) *J* ⟧ $\Longrightarrow$
ideal R (*rInvim R* (*qring R I*) (*pj R I*) *J*)
⟨proof⟩

lemma *pj-invim-cont-I*:⟦*Ring R*; *ideal R I*; *ideal* (*qring R I*) *J*⟧ $\Longrightarrow$
$I \subseteq$ (*rInvim R* (*qring R I*) (*pj R I*) *J*)
⟨proof⟩

lemma *pj-invim-mono1*:⟦*Ring R*; *ideal R I*; *ideal* (*qring R I*) *J1*;
ideal (*qring R I*) *J2*; $J1 \subseteq J2$ ⟧ $\Longrightarrow$
(*rInvim R* (*qring R I*) (*pj R I*) *J1*) $\subseteq$ (*rInvim R* (*qring R I*) (*pj R I*) *J2*)
⟨proof⟩

lemma *pj-img-ideal*:⟦*Ring R*; *ideal R I*; *ideal R J*; $I \subseteq J$⟧ $\Longrightarrow$
ideal (*qring R I*) ((*pj R I*)‘*J*)
⟨proof⟩

lemma *npQring*:⟦*Ring R*; *ideal R I*; $a \in$ *carrier R*⟧ $\Longrightarrow$
npow (*qring R I*) ($a \uplus_R I$) *n* = (*npow R a n*) $\uplus_R I$
⟨proof⟩

4.5 Primary ideals, Prime ideals

definition
maximal-set :: [′*a set set*, ′*a set*] ⇒ *bool* **where**
maximal-set S mx ⟷ $mx \in S \wedge (\forall s\in S.\ mx \subseteq s \longrightarrow s = mx)$

definition
nilpotent :: [-, ′*a*] ⇒ *bool* **where**
nilpotent R a ⟷ $(\exists (n{::}nat).\ a\,\hat{}^{R\ n} = \mathbf{0}_R)$

definition
zero-divisor :: [-, ′*a*] ⇒ *bool* **where**
zero-divisor R a ⟷ $(\exists x\in$ *carrier R*. $x \neq \mathbf{0}_R \wedge x \cdot_{rR} a = \mathbf{0}_R)$

definition
primary-ideal :: [-, ′*a set*] ⇒ *bool* **where**
primary-ideal R q ⟷ *ideal R q* $\wedge$ $(1_{rR}) \notin q \wedge$
($\forall x\in$ *carrier R*. $\forall y\in$ *carrier R*.
$x \cdot_{rR} y \in q \longrightarrow (\exists n.$ (*npow R x n*) $\in q \vee y \in q$))

definition
prime-ideal :: [-, ′*a set*] ⇒ *bool* **where**

prime-ideal R p $\longleftrightarrow$ *ideal R p* $\wedge$ $(1_{r R}) \notin p \wedge (\forall x \in$ *carrier R*. $\forall y \in$ *carrier R*. $(x \cdot_{r R} y \in p \longrightarrow x \in p \vee y \in p))$

definition
maximal-ideal :: [-, *'a set*] $\Rightarrow$ *bool* **where**
maximal-ideal R mx $\longleftrightarrow$ *ideal R mx* $\wedge$ $1_{r R} \notin mx \wedge$
$\{J.$ (*ideal R J* $\wedge$ $mx \subseteq J)\} = \{mx,$ *carrier R*$\}$

lemma (**in** *Ring*) *maximal-ideal-ideal*:⟦*maximal-ideal R mx*⟧ $\Longrightarrow$ *ideal R mx*
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-ideal-proper*:*maximal-ideal R mx* $\Longrightarrow$ $1_r \notin mx$
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-ideal*:*prime-ideal R I* $\Longrightarrow$ *ideal R I*
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-proper*:*prime-ideal R I* $\Longrightarrow$ $I \neq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-proper1*:*prime-ideal R p* $\Longrightarrow$ $1_r \notin p$
⟨*proof*⟩

lemma (**in** *Ring*) *primary-ideal-ideal*:*primary-ideal R q* $\Longrightarrow$ *ideal R q*
⟨*proof*⟩

lemma (**in** *Ring*) *primary-ideal-proper1*:*primary-ideal R q* $\Longrightarrow$ $1_r \notin q$
⟨*proof*⟩

lemma (**in** *Ring*) *prime-elems-mult-not*:⟦*prime-ideal R P*; $x \in$ *carrier R*;
$y \in$ *carrier R*; $x \notin P$; $y \notin P$ ⟧ $\Longrightarrow$ $x \cdot_r y \notin P$
⟨*proof*⟩

lemma (**in** *Ring*) *prime-is-primary*:*prime-ideal R p* $\Longrightarrow$ *primary-ideal R p*
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-prime-Tr0*:⟦*maximal-ideal R mx*; $x \in$ *carrier R*; $x \notin$ *mx*⟧
$\Longrightarrow$ $mx \mp$ (*Rxa R x*) = *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-prime*:*maximal-ideal R mx* $\Longrightarrow$ *prime-ideal R mx*
⟨*proof*⟩

lemma (**in** *Ring*) *chains-un*:⟦$c \in$ *chains* $\{I.$ *ideal R I* $\wedge$ $I \subset$ *carrier R*$\}$; $c \neq \{\}$⟧
$\Longrightarrow$ *ideal R* $(\bigcup c)$
⟨*proof*⟩

lemma (**in** *Ring*) *zeroring-no-maximal*:*zeroring R* $\Longrightarrow \neg (\exists I.$ *maximal-ideal R I*)
⟨*proof*⟩

lemma (**in** *Ring*) *id-maximal-Exist*:¬(*zeroring R*) $\Longrightarrow \exists I.$ *maximal-ideal R I*
⟨*proof*⟩

definition
ideal-Int :: [-, ′*a set set*] ⇒ ′*a set* **where**
ideal-Int R S == $\bigcap$ *S*

lemma (**in** *Ring*) *ideal-Int-ideal*:⟦*S* ⊆ {*I*. *ideal R I*}; *S*≠{}⟧ ⟹
ideal R ($\bigcap$ *S*)
⟨*proof*⟩

lemma (**in** *Ring*) *sum-prideals-Int*:⟦∀ *l* ≤ *n*. *f l* ∈ *carrier R*;
S = {*I*. *ideal R I* ∧ *f* ‘ {*i*. *i* ≤ *n*} ⊆ *I*}⟧ ⟹
(*sum-pr-ideals R f n*) = $\bigcap$ *S*
⟨*proof*⟩

This proves that (*sum-pr-ideals R f n*) is the smallest ideal containing *f* ‘ (*Nset n*)

primrec *ideal-n-prod*::[(′*a*, ′*m*) *Ring-scheme*, *nat*, *nat* ⇒ ′*a set*] ⇒ ′*a set*
where
ideal-n-prod0: *ideal-n-prod R 0 J* = *J 0*
| *ideal-n-prodSn*: *ideal-n-prod R* (*Suc n*) *J* =
(*ideal-n-prod R n J*) $\diamondsuit_{r\,R}$ (*J* (*Suc n*))

abbreviation
IDNPROD ((*3i*Π-,- -) [*98*,*98*,*99*]*98*) **where**
$i\Pi_{R,n}$ *J* == *ideal-n-prod R n J*

primrec
ideal-pow :: [′*a set*, (′*a*, ′*more*) *Ring-scheme*, *nat*] ⇒ ′*a set*
((*3*-/ $^{\diamondsuit - -}$) [*120*,*120*,*121*]*120*)
where
ip0: $I^{\diamondsuit R\ 0}$ = *carrier R*
| *ipSuc*: $I^{\diamondsuit R\ (Suc\ n)}$ = $I \diamondsuit_{r\,R} (I^{\diamondsuit R\ n})$

lemma (**in** *Ring*) *prod-mem-prod-ideals*:⟦*ideal R I*; *ideal R J*; *i* ∈ *I*; *j* ∈ *J*⟧ ⟹
$i \cdot_r j \in (I \diamondsuit_r J)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-ideal*:⟦*ideal R I*; *ideal R J* ⟧ ⟹
ideal R ($I \diamondsuit_r J$)
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-commute*:⟦*ideal R I*; *ideal R J*⟧ ⟹
$I \diamondsuit_r J = J \diamondsuit_r I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-subTr*:$[\![$*ideal R I*; *ideal R J*; *ideal R C*;
$\forall i{\in}I.\ \forall j{\in}J.\ i \cdot_r j \in C]\!] \Longrightarrow I \diamondsuit_r J \subseteq C$
⟨*proof*⟩

lemma (**in** *Ring*) *n-prod-idealTr*:
$(\forall k \le n.\ ideal\ R\ (J\ k)) \longrightarrow ideal\ R\ (ideal\text{-}n\text{-}prod\ R\ n\ J)$
⟨*proof*⟩

lemma (**in** *Ring*) *n-prod-ideal*:$[\![\forall k \le n.\ ideal\ R\ (J\ k)]\!]$
$\Longrightarrow\ ideal\ R\ (ideal\text{-}n\text{-}prod\ R\ n\ J)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-la1*:$[\![ideal\ R\ I;\ ideal\ R\ J]\!] \Longrightarrow (I \diamondsuit_r J) \subseteq I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-el1*:$[\![ideal\ R\ I;\ ideal\ R\ J;\ a \in (I \diamondsuit_r J)]\!] \Longrightarrow$
$a \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-la2*:$[\![ideal\ R\ I;\ ideal\ R\ J\ ]\!] \Longrightarrow (I \diamondsuit_r J) \subseteq J$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-sub-Int*:$[\![ideal\ R\ I;\ ideal\ R\ J\ ]\!] \Longrightarrow$
$(I \diamondsuit_r J) \subseteq I \cap J$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-el2*:$[\![ideal\ R\ I;\ ideal\ R\ J;\ a \in (I \diamondsuit_r J)]\!] \Longrightarrow$
$a \in J$
⟨*proof*⟩

$i\Pi_{R,n}\ J$ is the product of ideals

lemma (**in** *Ring*) *ele-n-prodTr0*:$[\![\forall k \le (Suc\ n).\ ideal\ R\ (J\ k);$
$a \in i\Pi_{R,(Suc\ n)}\ J\ ]\!] \Longrightarrow a \in (i\Pi_{R,n}\ J) \wedge a \in (J\ (Suc\ n))$
⟨*proof*⟩

lemma (**in** *Ring*) *ele-n-prodTr1*:
$(\forall k \le n.\ ideal\ R\ (J\ k)) \wedge a \in ideal\text{-}n\text{-}prod\ R\ n\ J \longrightarrow$
$(\forall k \le n.\ a \in (J\ k))$
⟨*proof*⟩

lemma (**in** *Ring*) *ele-n-prod*:$[\![\forall k \le n.\ ideal\ R\ (J\ k);$
$a \in ideal\text{-}n\text{-}prod\ R\ n\ J\ ]\!] \Longrightarrow\ \forall k \le n.\ a \in (J\ k)$
⟨*proof*⟩

lemma (**in** *Ring*) *idealprod-whole-l*:$ideal\ R\ I \Longrightarrow (carrier\ R) \diamondsuit_{r_R} I = I$
⟨*proof*⟩

lemma (**in** *Ring*) *idealprod-whole-r*:$ideal\ R\ I \Longrightarrow I \diamondsuit_r (carrier\ R) = I$

⟨*proof*⟩

lemma (**in** *Ring*) *idealpow-1-self*:*ideal R I* $\Longrightarrow$ $I^{\diamondsuit R\ (Suc\ 0)} = I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-pow-ideal*:*ideal R I* $\Longrightarrow$ *ideal R* $(I^{\diamondsuit R\ n})$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-prime*:$[\![$*ideal R I*; *ideal R J*; *prime-ideal R P*;
$I \diamondsuit_r J \subseteq P$ $]\!] \Longrightarrow I \subseteq P \vee J \subseteq P$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-n-prod-primeTr*:*prime-ideal R P* $\Longrightarrow$
$(\forall k \leq n.$ *ideal R* $(J\ k)) \longrightarrow$ (*ideal-n-prod R n J* $\subseteq P) \longrightarrow$
$(\exists i \leq n.\ (J\ i) \subseteq P)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-n-prod-prime*:$[\![$*prime-ideal R P*;
$\forall k \leq n.$ *ideal R* $(J\ k)$; *ideal-n-prod R n J* $\subseteq P]\!] \Longrightarrow$
$\exists i \leq n.\ (J\ i) \subseteq P$
⟨*proof*⟩

definition
ppa::[-, *nat* ⇒ *'a set*, *'a set*, *nat*] ⇒ (*nat* ⇒ *'a*) **where**
ppa R P A i l = (*SOME x*. $x \in A \wedge x \in$ (*P* (*skip i l*)) $\wedge\ x \notin P\ i$)

lemma (**in** *Ring*) *prod-primeTr*:$[\![$*prime-ideal R P*; *ideal R A*; $\neg\ A \subseteq P$;
ideal R B; $\neg\ B \subseteq P$ $]\!] \Longrightarrow \exists x.\ x \in A \wedge x \in B \wedge x \notin P$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-primeTr1*:$[\![\forall k \leq$ (*Suc n*). *prime-ideal R* (*P k*);
ideal R A; $\forall l \leq$ (*Suc n*). $\neg$ ($A \subseteq P\ l$);
$\forall k \leq$ (*Suc n*). $\forall l \leq$ (*Suc n*). $k = l \vee \neg$ (*P k*) $\subseteq$ (*P l*); $i \leq$ (*Suc n*)$]\!] \Longrightarrow$
$\forall l \leq n.$ *ppa R P A i l* $\in A\ \wedge$
ppa R P A i l $\in$ (*P* (*skip i l*)) $\wedge$ *ppa R P A i l* $\notin$ (*P i*)
⟨*proof*⟩

lemma (**in** *Ring*) *ppa-mem*:$[\![\forall k \leq$ (*Suc n*). *prime-ideal R* (*P k*); *ideal R A*;
$\forall l \leq$ (*Suc n*). $\neg$ ($A \subseteq P\ l$);
$\forall k \leq$ (*Suc n*). $\forall l \leq$ (*Suc n*). $k = l \vee \neg$ (*P k*) $\subseteq$ (*P l*);
$i \leq$ (*Suc n*); $l \leq n]\!] \Longrightarrow$ *ppa R P A i l* $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *nsum-memrTr*:($\forall i \leq n.$ *f i* $\in$ *carrier R*) $\longrightarrow$
($\forall l \leq n.$ *nsum R f l* $\in$ *carrier R*)
⟨*proof*⟩

lemma (**in** *Ring*) *nsum-memr*:$\forall i \leq n.$ *f i* $\in$ *carrier R* $\Longrightarrow$

$\forall l \leq n.\ nsum\ R\ f\ l \in carrier\ R$
⟨proof⟩

lemma (**in** *Ring*) *nsum-ideal-incTr*:$ideal\ R\ A \Longrightarrow$
$(\forall i \leq n.\ f\ i \in A) \longrightarrow\ nsum\ R\ f\ n \in A$
⟨proof⟩

lemma (**in** *Ring*) *nsum-ideal-inc*:$[\![ideal\ R\ A;\ \forall i \leq n.\ f\ i \in A]\!] \Longrightarrow$
$nsum\ R\ f\ n \in A$
⟨proof⟩

lemma (**in** *Ring*) *nsum-ideal-excTr*:$ideal\ R\ A \Longrightarrow$
$(\forall i \leq n.\ f\ i \in carrier\ R) \wedge (\exists j \leq n.\ (\forall l \in \{i.\ i \leq n\} - \{j\}.\ f\ l \in A)$
$\wedge (f\ j \notin A)) \longrightarrow nsum\ R\ f\ n \notin A$
⟨proof⟩

lemma (**in** *Ring*) *nsum-ideal-exc*:$[\![ideal\ R\ A;\ \forall i \leq n.\ f\ i \in carrier\ R;$
$\exists j \leq n.\ (\forall l\in\{i.\ i \leq n\} - \{j\}.\ f\ l \in A) \wedge (f\ j \notin A)\ ]\!] \Longrightarrow nsum\ R\ f\ n \notin A$
⟨proof⟩

lemma (**in** *Ring*) *nprod-memTr*:$(\forall i \leq n.\ f\ i \in carrier\ R) \longrightarrow$
$(\forall l.\ l \leq n \longrightarrow\ nprod\ R\ f\ l \in carrier\ R)$
⟨proof⟩

lemma (**in** *Ring*) *nprod-mem*:$[\![\forall i \leq n.\ f\ i \in carrier\ R;\ l \leq n]\!] \Longrightarrow$
$nprod\ R\ f\ l \in carrier\ R$
⟨proof⟩

lemma (**in** *Ring*) *ideal-nprod-incTr*:$ideal\ R\ A \Longrightarrow$
$(\forall i \leq n.\ f\ i \in carrier\ R) \wedge$
$(\exists l \leq n.\ f\ l \in A) \longrightarrow nprod\ R\ f\ n \in A$
⟨proof⟩

lemma (**in** *Ring*) *ideal-nprod-inc*:$[\![ideal\ R\ A;\ \forall i \leq n.\ f\ i \in carrier\ R;$
$\exists l \leq n.\ f\ l \in A]\!] \Longrightarrow nprod\ R\ f\ n \in A$
⟨proof⟩

lemma (**in** *Ring*) *nprod-excTr*:$prime\text{-}ideal\ R\ P \Longrightarrow$
$(\forall i \leq n.\ f\ i \in carrier\ R) \wedge (\forall l \leq n.\ f\ l \notin P) \longrightarrow$
$nprod\ R\ f\ n \notin P$
⟨proof⟩

lemma (**in** *Ring*) *prime-nprod-exc*:$[\![prime\text{-}ideal\ R\ P;\ \forall i \leq n.\ f\ i \in carrier\ R;$
$\forall l \leq n.\ f\ l \notin P]\!] \Longrightarrow nprod\ R\ f\ n \notin P$
⟨proof⟩

definition
nilrad :: - $\Rightarrow$ *'a set* **where**
$nilrad\ R = \{x.\ x \in carrier\ R \wedge nilpotent\ R\ x\}$

lemma (**in** *Ring*) *id-nilrad-ideal*:*ideal R* (*nilrad R*)
⟨*proof*⟩

definition
rad-ideal :: [-, *'a set*] ⇒ *'a set* **where**
rad-ideal R I = {*a*. *a* ∈ *carrier R* ∧ *nilpotent* (*qring R I*) ((*pj R I*) *a*)}

lemma (**in** *Ring*) *id-rad-invim*:*ideal R I* ⟹
rad-ideal R I = (*rInvim R* (*qring R I*) (*pj R I*) (*nilrad* (*qring R I*)))
⟨*proof*⟩

lemma (**in** *Ring*) *id-rad-ideal*:*ideal R I* ⟹ *ideal R* (*rad-ideal R I*)

⟨*proof*⟩

lemma (**in** *Ring*) *id-rad-cont-I*:*ideal R I* ⟹ *I* ⊆ (*rad-ideal R I*)
⟨*proof*⟩

lemma (**in** *Ring*) *id-rad-set*:*ideal R I* ⟹
rad-ideal R I = {*x*. *x* ∈ *carrier R* ∧ (∃ *n*. *npow R x n* ∈ *I*)}
⟨*proof*⟩

lemma (**in** *Ring*) *rad-primary-prime*:*primary-ideal R q* ⟹
prime-ideal R (*rad-ideal R q*)
⟨*proof*⟩

lemma (**in** *Ring*) *npow-notin-prime*:⟦*prime-ideal R P*; *x* ∈ *carrier R*; *x* ∉ *P*⟧
⟹ ∀ *n*. *npow R x n* ∉ *P*
⟨*proof*⟩

lemma (**in** *Ring*) *npow-in-prime*:⟦*prime-ideal R P*; *x* ∈ *carrier R*;
∃ *n*. *npow R x n* ∈ *P* ⟧ ⟹ *x* ∈ *P*
⟨*proof*⟩

definition
mul-closed-set::[-, *'a set*] ⇒ *bool* **where**
mul-closed-set R S ⟷ *S* ⊆ *carrier R* ∧ (∀ *s*∈*S*. ∀ *t*∈*S*. $s \cdot_{rR} t$ ∈ *S*)

locale *Idomain* = *Ring* +
assumes *idom*:
⟦*a* ∈ *carrier R*; *b* ∈ *carrier R*; $a \cdot_r b$ = **0**⟧ ⟹ *a* = **0** ∨ *b* = **0**

locale *Corps* =
fixes *K* (**structure**)
assumes *f-is-ring*: *Ring K*
and *f-inv*: ∀ *x*∈*carrier K* − {**0**}. ∃ *x'* ∈ *carrier K*. $x' \cdot_r x = 1_r$

lemma (**in** *Ring*) *mul-closed-set-sub*:*mul-closed-set R S* $\Longrightarrow$ *S* $\subseteq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *mul-closed-set-tOp-closed*:⟦*mul-closed-set R S*; *s* ∈ *S*;
t ∈ *S*⟧ $\Longrightarrow$ *s* $\cdot_r$ *t* ∈ *S*
⟨*proof*⟩

lemma (**in** *Corps*) *f-inv-unique*:⟦ *x* ∈ *carrier K* − {**0**}; *x'* ∈ *carrier K*;
x'' ∈ *carrier K*; *x'* $\cdot_r$ *x* = 1_r; *x''* $\cdot_r$ *x* = 1_r ⟧ $\Longrightarrow$ *x'* = *x''*
⟨*proof*⟩

definition
invf :: [-, *'a*] ⇒ *'a* **where**
invf K x = (*THE y*. *y* ∈ *carrier K* ∧ *y* $\cdot_{r K}$ *x* = $1_{r K}$)

lemma (**in** *Corps*) *invf-inv*:*x* ∈ *carrier K* − {**0**} $\Longrightarrow$
(*invf K x*) ∈ *carrier K* ∧ (*invf K x*) $\cdot_r$ *x* = 1_r
⟨*proof*⟩

definition
npowf :: - ⇒ *'a* ⇒ *int* ⇒ *'a* **where**
npowf K x n =
(*if 0* ≤ *n then npow K x* (*nat n*) *else npow K* (*invf K x*) (*nat* (− *n*)))

abbreviation
NPOWF :: [*'a*, -, *int*] ⇒ *'a* ((*3*-$_-$$^-$) [*77*,*77*,*78*]*77*) **where**
$a_K{}^n$ == *npowf K a n*

abbreviation
IOP :: [*'a*, -] ⇒ *'a* ((-$^{- -}$) [*87*,*88*]*87*) **where**
a^{-K} == *invf K a*

lemma (**in** *Idomain*) *idom-is-ring*: *Ring R* ⟨*proof*⟩

lemma (**in** *Idomain*) *idom-tOp-nonzeros*:⟦*x* ∈ *carrier R*;
y ∈ *carrier R*; *x* ≠ **0**; *y* ≠ **0**⟧ $\Longrightarrow$ *x* $\cdot_r$ *y* ≠ **0**
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-potent-nonzero*:
⟦*x* ∈ *carrier R*; *x* ≠ **0**⟧ $\Longrightarrow$ *npow R x n* ≠ **0**
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-potent-unit*:⟦*a* ∈ *carrier R*; *0* < *n*⟧
$\Longrightarrow$ (*Unit R a*) = (*Unit R* (*npow R a n*))
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-mult-cancel-r*:$[\![a \in carrier\ R;$
$b \in carrier\ R;\ c \in carrier\ R;\ c \neq \mathbf{0};\ a \cdot_r c = b \cdot_r c]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-mult-cancel-l*:$[\![a \in carrier\ R;$
$b \in carrier\ R;\ c \in carrier\ R;\ c \neq \mathbf{0};\ c \cdot_r a = c \cdot_r b]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *Corps*) *invf-closed1*:$x \in carrier\ K - \{\mathbf{0}\} \Longrightarrow$
$invf\ K\ x \in (carrier\ K) - \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Corps*) *linvf*:$x \in carrier\ K - \{\mathbf{0}\} \Longrightarrow (invf\ K\ x) \cdot_r x = 1_r$
⟨*proof*⟩

lemma (**in** *Corps*) *field-is-ring*:*Ring K*
⟨*proof*⟩

lemma (**in** *Corps*) *invf-one*:$1_r \neq \mathbf{0} \Longrightarrow invf\ K\ (1_r) = 1_r$
⟨*proof*⟩

lemma (**in** *Corps*) *field-tOp-assoc*:$[\![x \in carrier\ K;\ y \in carrier\ K;\ z \in carrier\ K]\!]$
$\Longrightarrow x \cdot_r y \cdot_r z = x \cdot_r (y \cdot_r z)$
⟨*proof*⟩

lemma (**in** *Corps*) *field-tOp-commute*:$[\![x \in carrier\ K;\ y \in carrier\ K]\!]$
$\Longrightarrow x \cdot_r y = y \cdot_r x$
⟨*proof*⟩

lemma (**in** *Corps*) *field-inv-inv*:$[\![x \in carrier\ K;\ x \neq \mathbf{0}]\!] \Longrightarrow (x^{-K})^{-K} = x$
⟨*proof*⟩

lemma (**in** *Corps*) *field-is-idom*:*Idomain K*
⟨*proof*⟩

lemma (**in** *Corps*) *field-potent-nonzero*:$[\![x \in carrier\ K;\ x \neq \mathbf{0}]\!] \Longrightarrow$
$x^{\hat{}K\ n} \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Corps*) *field-potent-nonzero1*:$[\![x \in carrier\ K;\ x \neq \mathbf{0}]\!] \Longrightarrow {x_K}^n \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Corps*) *field-nilp-zero*:$[\![x \in carrier\ K;\ x^{\hat{}K\ n} = \mathbf{0}]\!] \Longrightarrow x = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-mem*:$[\![a \in carrier\ K;\ a \neq \mathbf{0}]\!] \Longrightarrow$
$npowf\ K\ a\ n \in carrier\ K$
⟨*proof*⟩

lemma (**in** *Corps*) *field-npowf-exp-zero*:$[\![a \in carrier\ K;\ a \neq \mathbf{0}]\!] \Longrightarrow$
$npowf\ K\ a\ 0 = 1_r$
⟨*proof*⟩

lemma (**in** *Corps*) *npow-exp-minusTr1*:$[\![x \in carrier\ K;\ x \neq \mathbf{0};\ 0 \leq i]\!] \Longrightarrow$
$0 \leq i - (int\ j) \longrightarrow x_K{}^{(i - (int\ j))} = x^{\hat{}K\ (nat\ i)} \cdot_r (x^{-K})^{\hat{}K\ j}$
⟨*proof*⟩

lemma (**in** *Corps*) *npow-exp-minusTr2*:$[\![x \in carrier\ K;\ x \neq \mathbf{0};\ 0 \leq i;\ 0 \leq j;$
$0 \leq i - j]\!] \Longrightarrow x_K{}^{(i - j)} = x^{\hat{}K\ (nat\ i)} \cdot_r (x^{-K})^{\hat{}K\ (nat\ j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-inv*:$[\![x \in carrier\ K;\ x \neq \mathbf{0};\ 0 \leq j]\!] \Longrightarrow x_K{}^j = (x^{-K})_K{}^{(-j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-inv1*:$[\![x \in carrier\ K;\ x \neq \mathbf{0};\ \neg\ 0 \leq j]\!] \Longrightarrow$
$x_K{}^j = (x^{-K})_K{}^{(-j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-inverse*:$[\![x \in carrier\ K;\ x \neq \mathbf{0}]\!] \Longrightarrow x_K{}^j = (x^{-K})_K{}^{(-j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-expTr1*:$[\![x \in carrier\ K;\ x \neq \mathbf{0};\ 0 \leq i;\ 0 \leq j;$
$0 \leq i - j]\!] \Longrightarrow x_K{}^{(i - j)} = x_K{}^i \cdot_r x_K{}^{(-\ j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-expTr2*:$[\![x \in carrier\ K;\ x \neq \mathbf{0};\ 0 \leq i + j]\!] \Longrightarrow$
$x_K{}^{(i + j)} = x_K{}^i \cdot_r x_K{}^j$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-exp-add*:$[\![x \in carrier\ K;\ x \neq \mathbf{0}]\!] \Longrightarrow$
$x_K{}^{(i + j)} = x_K{}^i \cdot_r x_K{}^j$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-exp-1-add*:$[\![x \in carrier\ K;\ x \neq \mathbf{0}]\!] \Longrightarrow$
$x_K{}^{(1 + j)} = x \cdot_r x_K{}^j$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-minus*:$[\![x \in carrier\ K;\ x \neq \mathbf{0}]\!] \Longrightarrow (x_K{}^j)^{-K} = x_K{}^{(-\ j)}$
⟨*proof*⟩

lemma (**in** *Ring*) *residue-fieldTr*:$[\![maximal\text{-}ideal\ R\ mx;\ x \in carrier(qring\ R\ mx);$
$x \neq \mathbf{0}_{(qring\ R\ mx)}]\!] \Longrightarrow \exists\, y{\in}carrier\ (qring\ R\ mx).\ y \cdot_{r(qring\ R\ mx)} x = 1_{r(qring\ R\ mx)}$
⟨*proof*⟩

lemma (**in** *Ring*) *residue-field-cd*: $maximal\text{-}ideal\ R\ mx \Longrightarrow$
$Corps\ (qring\ R\ mx)$
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-set-idealTr*:
$maximal\text{-}set\ \{I.\ ideal\ R\ I \wedge S \cap I = \{\}\}\ mx \Longrightarrow ideal\ R\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-setTr*: $[\![maximal\text{-}set\ \{I.\ ideal\ R\ I \wedge S \cap I = \{\}\}\ mx;$
$ideal\ R\ J;\ mx \subset J\ ]\!] \Longrightarrow S \cap J \neq \{\}$
⟨*proof*⟩

lemma (**in** *Ring*) *mulDisj*: $[\![mul\text{-}closed\text{-}set\ R\ S;\ 1_r \in S;\ \mathbf{0} \notin S;$
$T = \{I.\ ideal\ R\ I \wedge S \cap I = \{\}\};\ maximal\text{-}set\ T\ mx\ ]\!] \Longrightarrow prime\text{-}ideal\ R\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *ex-mulDisj-maximal*: $[\![mul\text{-}closed\text{-}set\ R\ S;\ \mathbf{0} \notin S;\ 1_r \in S;$
$T = \{I.\ ideal\ R\ I \wedge S \cap I = \{\}\}]\!] \Longrightarrow\ \exists\, mx.\ maximal\text{-}set\ T\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *ex-mulDisj-prime*: $[\![mul\text{-}closed\text{-}set\ R\ S;\ \mathbf{0} \notin S;\ 1_r \in S]\!] \Longrightarrow$
$\exists\, mx.\ prime\text{-}ideal\ R\ mx \wedge S \cap mx = \{\}$
⟨*proof*⟩

lemma (**in** *Ring*) *nilradTr1*: $\neg\ zeroring\ R \Longrightarrow nilrad\ R = \bigcap\ \{p.\ prime\text{-}ideal\ R\ p\}$
⟨*proof*⟩

lemma (**in** *Ring*) *nonilp-residue-nilrad*: $[\![\neg\ zeroring\ R;\ x \in carrier\ R;$
$nilpotent\ (qring\ R\ (nilrad\ R))\ (x \uplus_R (nilrad\ R))]\!] \Longrightarrow$
$x \uplus_R (nilrad\ R) = \mathbf{0}_{(qring\ R\ (nilrad\ R))}$
⟨*proof*⟩

lemma (**in** *Ring*) *ex-contid-maximal*: $[\![\ S = \{1_r\};\ \mathbf{0} \notin S;\ ideal\ R\ I;\ I \cap S = \{\};$
$T = \{J.\ ideal\ R\ J \wedge S \cap J = \{\} \wedge I \subseteq J\}]\!] \Longrightarrow \exists\, mx.\ maximal\text{-}set\ T\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *contid-maximal*: $[\![S = \{1_r\};\ \mathbf{0} \notin S;\ ideal\ R\ I;\ I \cap S = \{\};$
$T = \{J.\ ideal\ R\ J \wedge S \cap J = \{\} \wedge I \subseteq J\};\ maximal\text{-}set\ T\ mx]\!] \Longrightarrow$
$maximal\text{-}ideal\ R\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-contained-maxid*: $[\![\neg(zeroring\ R);\ ideal\ R\ I;\ 1_r \notin I]\!] \Longrightarrow$
$\exists\, mx.\ maximal\text{-}ideal\ R\ mx \wedge I \subseteq mx$
⟨*proof*⟩

lemma (**in** *Ring*) *nonunit-principal-id*:$[\![a \in carrier\ R; \neg\ (Unit\ R\ a)]\!] \Longrightarrow$
$(R \diamondsuit_p\ a) \neq (carrier\ R)$
⟨*proof*⟩

lemma (**in** *Ring*) *nonunit-contained-maxid*:$[\![\neg(zeroring\ R); a \in carrier\ R;$
$\neg\ Unit\ R\ a\ ]\!] \Longrightarrow \exists\, mx.\ maximal\text{-}ideal\ R\ mx \wedge a \in\ mx$
⟨*proof*⟩

definition
local-ring :: - ⇒ *bool* **where**
$local\text{-}ring\ R == Ring\ R \wedge \neg\ zeroring\ R \wedge card\ \{mx.\ maximal\text{-}ideal\ R\ mx\} = 1$

lemma (**in** *Ring*) *local-ring-diff*:$[\![\neg\ zeroring\ R; ideal\ R\ mx; mx \neq carrier\ R;$
$\forall\, a\in (carrier\ R - mx).\ Unit\ R\ a\ ]\!] \Longrightarrow local\text{-}ring\ R \wedge maximal\text{-}ideal\ R\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *localring-unit*:$[\![\neg\ zeroring\ R; maximal\text{-}ideal\ R\ mx;$
$\forall\, x.\ x \in mx \longrightarrow Unit\ R\ (x \pm 1_r)\ ]\!] \Longrightarrow local\text{-}ring\ R$
⟨*proof*⟩

definition
J-rad ::- ⇒ *'a set* **where**
$J\text{-}rad\ R = (if\ (zeroring\ R)\ then\ (carrier\ R)\ else$
$\bigcap\ \{mx.\ maximal\text{-}ideal\ R\ mx\})$

lemma (**in** *Ring*) *zeroring-J-rad-empty*:$zeroring\ R \Longrightarrow J\text{-}rad\ R = carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *J-rad-mem*:$x \in J\text{-}rad\ R \Longrightarrow x \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *J-rad-unit*:$[\![\neg\ zeroring\ R; x \in J\text{-}rad\ R]\!] \Longrightarrow$
$\forall\, y.\ (y\in carrier\ R \longrightarrow Unit\ R\ (1_r \pm (-_a\ x) \cdot_r\ y))$
⟨*proof*⟩

end

theory *Algebra5* **imports** *Algebra4* **begin**

4.6 Operation of ideals

lemma (**in** *Ring*) *ideal-sumTr1*:$[\![ideal\ R\ A; ideal\ R\ B]\!] \Longrightarrow$
$A \mp B = \bigcap\ \{J.\ ideal\ R\ J \wedge (A \cup B) \subseteq J\}$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-ideals-commute*:$[\![ideal\ R\ A; ideal\ R\ B]\!] \Longrightarrow$

$A \mp B = B \mp A$
⟨proof⟩

lemma (**in** *Ring*) *ideal-prod-mono1*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
$A \subseteq B$ ⟧ $\Longrightarrow A \diamondsuit_r C \subseteq B \diamondsuit_r C$
⟨proof⟩

lemma (**in** *Ring*) *ideal-prod-mono2*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
$A \subseteq B$ ⟧ $\Longrightarrow C \diamondsuit_r A \subseteq C \diamondsuit_r B$
⟨proof⟩

lemma (**in** *Ring*) *cont-ideal-prod*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
$A \subseteq C$; $B \subseteq C$ ⟧ $\Longrightarrow A \diamondsuit_r B \subseteq C$
⟨proof⟩

lemma (**in** *Ring*) *ideal-distrib*:⟦*ideal R A*; *ideal R B*; *ideal R C*⟧ $\Longrightarrow$
$A \diamondsuit_r (B \mp C) = A \diamondsuit_r B \mp A \diamondsuit_r C$
⟨proof⟩

definition
coprime-ideals::[-, *'a set*, *'a set*] ⇒ *bool* **where**
coprime-ideals R A B ⟷ $A \mp_R B$ = *carrier R*

lemma (**in** *Ring*) *coprimeTr*:⟦*ideal R A*; *ideal R B*⟧ $\Longrightarrow$
coprime-ideals R A B = ($\exists\, a \in A.\ \exists\, b \in B.\ a \pm b = 1_r$)
⟨proof⟩

lemma (**in** *Ring*) *coprime-int-prod*:⟦*ideal R A*; *ideal R B*; *coprime-ideals R A B*⟧
$\Longrightarrow A \cap B = A \diamondsuit_r B$
⟨proof⟩

lemma (**in** *Ring*) *coprime-elems*:⟦*ideal R A*; *ideal R B*; *coprime-ideals R A B*⟧ $\Longrightarrow$
$\exists\, a{\in}A.\ \exists\, b{\in}B.\ a \pm b = 1_r$
⟨proof⟩

lemma (**in** *Ring*) *coprime-elemsTr*:⟦*ideal R A*; *ideal R B*; $a{\in}A$; $b{\in}B$; $a \pm b =$
1_r⟧
$\Longrightarrow$ *pj R A b* $= 1_{r(qring\ R\ A)} \wedge$ *pj R B a* $= 1_{r(qring\ R\ B)}$
⟨proof⟩

lemma (**in** *Ring*) *partition-of-unity*:⟦*ideal R A*; $a \in A$; $b \in$ *carrier R*;
$a \pm b = 1_r$; $u \in$ *carrier R*; $v \in$ *carrier R*⟧ $\Longrightarrow$
pj R A $(a \cdot_r v \pm b \cdot_r u)$ = *pj R A u*
⟨proof⟩

lemma (**in** *Ring*) *coprimes-commute*:⟦*ideal R A*; *ideal R B*; *coprime-ideals R A B*
⟧
$\Longrightarrow$ *coprime-ideals R B A*
⟨proof⟩

lemma (**in** *Ring*) *coprime-surjTr*:$[\![$*ideal R A*; *ideal R B*; *coprime-ideals R A B*;
X ∈ *carrier* (*qring R A*); *Y* ∈ *carrier* (*qring R B*) $]\!] \Longrightarrow$
∃ *r*∈*carrier R*. *pj R A r* = *X* ∧ *pj R B r* = *Y*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr0*:$[\![$*ideal R A*; *ideal R B*; *ideal R C*;
coprime-ideals R A C; *coprime-ideals R B C* $]\!] \Longrightarrow$
coprime-ideals R (*A* $\diamondsuit_r$ *B*) *C*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr1*:*ideal R C* $\Longrightarrow$
(∀ *k* ≤ *n*. *ideal R* (*J k*)) ∧ (∀ *i* ≤ *n*. *coprime-ideals R* (*J i*) *C*) ⟶
coprime-ideals R (*i*$\Pi_{R,n}$ *J*) *C*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr2*:$[\![$*ideal R C*; (∀ *k* ≤ *n*. *ideal R* (*J k*));
(∀ *i* ≤ *n*. *coprime-ideals R* (*J i*) *C*) $]\!] \Longrightarrow$
coprime-ideals R (*i*$\Pi_{R,n}$ *J*) *C*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr3*:(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*)) ∧
(∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). *i* ≠ *j* ⟶
coprime-ideals R (*J i*) (*J j*)) ⟶ *coprime-ideals R* (*i*$\Pi_{R,n}$ *J*) (*J* (*Suc n*))
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr4*:$[\![$(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*)) ∧
(∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). *i* ≠ *j* ⟶
coprime-ideals R (*J i*) (*J j*))$]\!] \Longrightarrow$ *coprime-ideals R* (*i*$\Pi_{R,n}$ *J*) (*J* (*Suc n*))
⟨*proof*⟩

4.7 Direct product1, general case

definition
prod-tOp :: [′*i set*, ′*i* ⇒ (′*a*, ′*m*) *Ring-scheme*] ⇒
(′*i* ⇒ ′*a*) ⇒ (′*i* ⇒ ′*a*) ⇒ (′*i* ⇒ ′*a*) **where**
prod-tOp I A = (λ*f*∈*carr-prodag I A*. λ*g*∈*carr-prodag I A*.
λ*x*∈*I*. (*f x*) $\cdot_{r(A\ x)}$ (*g x*))

definition
prod-one::[′*i set*, ′*i* ⇒ (′*a*, ′*m*) *Ring-scheme*] ⇒ (′*i* ⇒ ′*a*) **where**
prod-one I A == λ*x*∈*I*. $1_{r(A\ x)}$

definition
prodrg :: [′*i set*, ′*i* ⇒ (′*a*, ′*more*) *Ring-scheme*] ⇒ (′*i* ⇒ ′*a*) *Ring* **where**
prodrg I A = (|*carrier* = *carr-prodag I A*, *pop* = *prod-pOp I A*, *mop* =
prod-mOp I A, *zero* = *prod-zero I A*, *tp* = *prod-tOp I A*,

un = *prod-one I A* |)

abbreviation
PRODRING ((*r*Π_/ -) [*72*,*73*]*72*) **where**
$r\Pi_I$ *A* == *prodrg I A*

definition
augm-func :: [*nat*, *nat* ⇒ *'a*,*'a set*, *nat*, *nat* ⇒ *'a*, *'a set*] ⇒ *nat* ⇒ *'a* **where**
augm-func n f A m g B = (λ*i*∈{*j*. *j* ≤ (*n* + *m*)}. *if i* ≤ *n then f i else*
if (*Suc n*) ≤ *i* ∧ *i* ≤ *n* + *m then g* ((*sliden* (*Suc n*)) *i*) *else undefined*)

definition
ag-setfunc :: [*nat*, *nat* ⇒ (*'a*, *'more*) *Ring-scheme*, *nat*,
nat ⇒ (*'a*, *'more*) *Ring-scheme*] ⇒ (*nat* ⇒ *'a*) *set* ⇒ (*nat* ⇒ *'a*) *set*
⇒ (*nat* ⇒ *'a*) *set* **where**
ag-setfunc n B1 m B2 X Y =
{*f*. ∃ *g*. ∃ *h*. (*g*∈*X*) ∧(*h*∈*Y*) ∧(*f* = (*augm-func n g* (*Un-carrier* {*j*. *j* ≤ *n*} *B1*)

m h (*Un-carrier* {*j*. *j* ≤ (*m* − *1*)} *B2*)))}

primrec
ac-fProd-Rg :: [*nat*, *nat* ⇒ (*'a*, *'more*) *Ring-scheme*] ⇒
(*nat* ⇒ *'a*) *set*
where
fprod-0: *ac-fProd-Rg 0 B* = *carr-prodag* {*0*::*nat*} *B*
| *frpod-n*: *ac-fProd-Rg* (*Suc n*) *B* = *ag-setfunc n B* (*Suc 0*) (*compose* {*0*::*nat*}
B (*slide* (*Suc n*))) (*carr-prodag* {*j*. *j* ≤ *n*} *B*) (*carr-prodag* {*0*} (*compose* {*0*} *B*
(*slide* (*Suc n*))))

definition
prodB1 :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m*) *Ring-scheme*] ⇒
(*nat* ⇒ (*'a*, *'m*) *Ring-scheme*) **where**
prodB1 R S = (λ*k*. *if k*=*0 then R else if k*=*Suc 0 then S else*
undefined)

definition
Prod2Rg :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m*) *Ring-scheme*]
⇒ (*nat* ⇒ *'a*) *Ring* (**infixl** $\bigoplus_r$ *80*) **where**
A1 $\bigoplus_r$ *A2* = *prodrg* {*0*, *Suc 0*} (*prodB1 A1 A2*)

Don't try (*Prod-ring* (*Nset n*) *B*) $\bigoplus_r$ (*B* (*Suc n*))

lemma *carr-prodrg-mem-eq*:⟦*f* ∈ *carrier* ($r\Pi_I$ *A*); *g* ∈ *carrier* ($r\Pi_I$ *A*);
∀ *i*∈*I*. *f i* = *g i* ⟧ ⟹ *f* = *g*
⟨*proof*⟩

lemma *prod-tOp-mem*:⟦∀ *k*∈*I*. *Ring* (*A k*); *X* ∈ *carr-prodag I A*;

$Y \in$ *carr-prodag I A*$\rrbracket \Longrightarrow$ *prod-tOp I A X Y* $\in$ *carr-prodag I A*
⟨*proof*⟩

lemma *prod-tOp-func*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
prod-tOp I A $\in$ *carr-prodag I A* $\rightarrow$ *carr-prodag I A* $\rightarrow$ *carr-prodag I A*
⟨*proof*⟩

lemma *prod-one-func*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
prod-one I A $\in$ *carr-prodag I A*
⟨*proof*⟩

lemma *prodrg-carrier*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
carrier (*prodrg I A*) = *carrier* (*prodag I A*)
⟨*proof*⟩

lemma *prodrg-ring*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$ *Ring* (*prodrg I A*)
⟨*proof*⟩

lemma *prodrg-elem-extensional*:$\llbracket\forall k\in I.$ *Ring* $(A\ k)$; $f \in$ *carrier* (*prodrg I A*)$\rrbracket$
$\Longrightarrow$ $f \in$ *extensional I*
⟨*proof*⟩

lemma *prodrg-pOp*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
pop (*prodrg I A*) = *prod-pOp I A*
⟨*proof*⟩

lemma *prodrg-mOp*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
mop (*prodrg I A*) = *prod-mOp I A*
⟨*proof*⟩

lemma *prodrg-zero*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
zero (*prodrg I A*) = *prod-zero I A*
⟨*proof*⟩

lemma *prodrg-tOp*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
tp (*prodrg I A*) = *prod-tOp I A*
⟨*proof*⟩

lemma *prodrg-one*:$\forall k\in I.$ *Ring* $(A\ k) \Longrightarrow$
un (*prodrg I A*) = *prod-one I A*
⟨*proof*⟩

lemma *prodrg-sameTr5*:$\llbracket\forall k\in I.$ *Ring* $(A\ k)$; $\forall k\in I.\ A\ k = B\ k\rrbracket$
$\Longrightarrow$ *prod-tOp I A* = *prod-tOp I B*
⟨*proof*⟩

lemma *prodrg-sameTr6*:$\llbracket\forall k\in I.$ *Ring* $(A\ k)$; $\forall k\in I.\ A\ k = B\ k\rrbracket$
$\Longrightarrow$ *prod-one I A* = *prod-one I B*

⟨*proof*⟩

lemma *prodrg-same*:⟦∀ *k*∈*I*. *Ring* (*A k*); ∀ *k*∈*I*. *A k* = *B k*⟧
⟹ *prodrg I A* = *prodrg I B*
⟨*proof*⟩

lemma *prodrg-component*:⟦*f* ∈ *carrier* (*prodrg I A*); *i* ∈ *I*⟧ ⟹
f i ∈ *carrier* (*A i*)
⟨*proof*⟩

lemma *project-rhom*:⟦∀ *k*∈*I*. *Ring* (*A k*); *j* ∈ *I*⟧ ⟹
PRoject I A j ∈ *rHom* (*prodrg I A*) (*A j*)
⟨*proof*⟩

lemma *augm-funcTr*:⟦∀ *k* ≤(*Suc n*). *Ring* (*B k*);
f ∈ *carr-prodag* {*i*. *i* ≤ (*Suc n*)} *B*⟧ ⟹
f = *augm-func n* (*restrict f* {*i*. *i* ≤ *n*}) (*Un-carrier* {*i*. *i* ≤ *n*} *B*) (*Suc 0*)
(λ*x*∈{*0*::*nat*}. *f* (*x* + *Suc n*))
(*Un-carrier* {*0*} (*compose* {*0*} *B* (*slide* (*Suc n*))))
⟨*proof*⟩

lemma *A-to-prodag-mem*:⟦*Ring A*; ∀ *k*∈*I*. *Ring* (*B k*); ∀ *k*∈*I*. (*S k*) ∈
rHom A (*B k*); *x* ∈ *carrier A* ⟧ ⟹ *A-to-prodag A I S B x* ∈ *carr-prodag I B*
⟨*proof*⟩

lemma *A-to-prodag-rHom*:⟦*Ring A*; ∀ *k*∈*I*. *Ring* (*B k*); ∀ *k*∈*I*. (*S k*) ∈
rHom A (*B k*) ⟧ ⟹ *A-to-prodag A I S B* ∈ *rHom A* ($r\Pi_I$ *B*)
⟨*proof*⟩

lemma *ac-fProd-ProdTr1*:∀ *k* ≤ (*Suc n*). *Ring* (*B k*) ⟹
ag-setfunc n B (*Suc 0*) (*compose* {*0*::*nat*} *B* (*slide* (*Suc n*)))
(*carr-prodag* {*i*. *i* ≤ *n*} *B*) (*carr-prodag* {*0*}
(*compose* {*0*} *B* (*slide* (*Suc n*)))) ⊆ *carr-prodag* {*i*. *i* ≤ (*Suc n*)} *B*
⟨*proof*⟩

lemma *ac-fProd-Prod*:∀ *k* ≤ *n*. *Ring* (*B k*) ⟹
ac-fProd-Rg n B = *carr-prodag* {*j*. *j* ≤ *n*} *B*
⟨*proof*⟩

A direct product of a finite number of rings defined with *ac-fProd-Rg* is equal to that defined by using *carr-prodag*.

definition
fprodrg :: [*nat*, *nat* ⇒ ('*a*, '*more*) *Ring-scheme*] ⇒
(|*carrier*:: (*nat* ⇒ '*a*) *set*, *pop*::[(*nat* ⇒ '*a*), (*nat* ⇒ '*a*)]
⇒ (*nat* ⇒ '*a*), *mop*:: (*nat* ⇒ '*a*) ⇒ (*nat* ⇒ '*a*), *zero*::(*nat* ⇒ '*a*),
tp :: [(*nat* ⇒ '*a*), (*nat* ⇒ '*a*)] ⇒ (*nat* ⇒ '*a*), *un* :: (*nat* ⇒ '*a*) |) **where**

fprodrg n B = (| *carrier* = *ac-fProd-Rg n B*,
pop = λ*f*. λ*g*. *prod-pOp* {*i*. *i* ≤ *n*} *B f g*, *mop* = λ*f*. *prod-mOp* {*i*. *i* ≤ *n*} *B*

f,
$zero = prod\text{-}zero\ \{i.\ i \leq n\}\ B,\ tp = \ \lambda f.\ \lambda g.\ prod\text{-}tOp\ \{i.\ i \leq n\}\ B\ f\ g,$
$un = prod\text{-}one\ \{i.\ i \leq n\}\ B\ |)$

definition
$fPRoject :: [nat,\ nat \Rightarrow ('a,\ 'more)\ Ring\text{-}scheme,\ nat]$
$\Rightarrow (nat \Rightarrow 'a) \Rightarrow 'a$ **where**
$fPRoject\ n\ B\ x = (\lambda f{\in}ac\text{-}fProd\text{-}Rg\ n\ B.\ f\ x)$

lemma $fprodrg\text{-}ring{:}\forall k \leq n.\ Ring\ (B\ k) \Longrightarrow Ring\ (fprodrg\ n\ B)$
$\langle proof \rangle$

4.8 Chinese remainder theorem

lemma $Chinese\text{-}remTr1{:}[\![Ring\ A;\ \forall k \leq (n{::}nat).\ ideal\ A\ (J\ k);$
$\forall k \leq n.\ B\ k = qring\ A\ (J\ k);\ \forall k \leq n.\ S\ k = pj\ A\ (J\ k)\ ]\!] \Longrightarrow$
$ker_{A,(r\Pi_{\{j.\ j\ \leq\ n\}}\ B)}\ (A\text{-}to\text{-}prodag\ A\ \{j.\ j \leq n\}\ S\ B) =$
$\bigcap\ \{I.\ \exists k{\in}\{j.\ j \leq n\}.\ I = (J\ k)\}$
$\langle proof \rangle$

lemma (**in** $Ring$) $coprime\text{-}prod\text{-}int2Tr$:
$((\forall k \leq (Suc\ n).\ ideal\ R\ (J\ k)) \wedge$
$(\forall i \leq (Suc\ n).\ \forall j \leq (Suc\ n).\ (i \neq j \longrightarrow coprime\text{-}ideals\ R\ (J\ i)\ (J\ j))))$
$\longrightarrow (\bigcap\ \{I.\ \exists k \leq (Suc\ n).\ I = (J\ k)\} = ideal\text{-}n\text{-}prod\ R\ (Suc\ n)\ J)$
$\langle proof \rangle$

lemma (**in** $Ring$) $coprime\text{-}prod\text{-}int2{:}[\![\ \forall k \leq (Suc\ n).\ ideal\ R\ (J\ k);$
$\forall i \leq (Suc\ n).\ \forall j \leq (Suc\ n).\ (i \neq j \longrightarrow coprime\text{-}ideals\ R\ (J\ i)\ (J\ j))]\!]$
$\Longrightarrow (\bigcap\ \{I.\ \exists k \leq (Suc\ n).\ I = (J\ k)\} = ideal\text{-}n\text{-}prod\ R\ (Suc\ n)\ J)$
$\langle proof \rangle$

lemma (**in** $Ring$) $coprime\text{-}2\text{-}n{:}[\![ideal\ R\ A;\ ideal\ R\ B]\!] \Longrightarrow$
$(qring\ R\ A) \bigoplus_r (qring\ R\ B) = r\Pi_{\{j.\ j\ \leq\ (Suc\ 0)\}}\ (prodB1\ (qring\ R\ A)\ (qring\ R$
$B))$
$\langle proof \rangle$

In this and following lemmata, ideals A and B are of type $('a,\ 'more)$ $RingType\text{-}scheme$. Don't try $(r\Pi_(Nset\ n)\ B) \bigoplus_r B\ (Suc\ n)$

lemma (**in** $Ring$) $A\text{-}to\text{-}prodag2\text{-}hom{:}[\![ideal\ R\ A;\ ideal\ R\ B;\ S\ 0 = pj\ R\ A;$
$S\ (Suc\ 0) = pj\ R\ B]\!] \Longrightarrow$
$A\text{-}to\text{-}prodag\ R\ \{j.\ j \leq (Suc\ 0)\}\ S\ (prodB1\ (qring\ R\ A)\ (qring\ R\ B)) \in$
$rHom\ R\ (qring\ R\ A \bigoplus_r qring\ R\ B)$
$\langle proof \rangle$

lemma (**in** $Ring$) $A2coprime\text{-}rsurjecTr{:}[\![ideal\ R\ A;\ ideal\ R\ B;\ S\ 0 = pj\ R\ A;$
$S\ (Suc\ 0) = pj\ R\ B]\!] \Longrightarrow$
$(carrier\ (qring\ R\ A \bigoplus_r qring\ R\ B)) =$
$carr\text{-}prodag\ \{j.\ j \leq (Suc\ 0)\}\ (prodB1\ (qring\ R\ A)\ (qring\ R\ B))$

⟨proof⟩

lemma (**in** *Ring*) *A2coprime-rsurjec*:⟦*ideal R A*; *ideal R B*; *S 0* = *pj R A*;
S (*Suc 0*) = *pj R B*; *coprime-ideals R A B*⟧ ⟹
$surjec_{R,((qring\ R\ A)\ \bigoplus_r\ (qring\ R\ B))}$
(*A-to-prodag R* {*j*. *j*≤(*Suc 0*)} *S* (*prodB1* (*qring R A*) (*qring R B*)))
⟨proof⟩

lemma (**in** *Ring*) *prod2-n-Tr1*:⟦∀ *k* ≤ (*Suc 0*). *ideal R* (*J k*);
∀ *k* ≤ (*Suc 0*). *B k* = *qring R* (*J k*);
∀ *k* ≤ (*Suc 0*). *S k* = *pj R* (*J k*) ⟧ ⟹
A-to-prodag R {*j*. *j* ≤ (*Suc 0*)} *S*
(*prodB1* (*qring R* (*J 0*)) (*qring R* (*J* (*Suc 0*)))) =
A-to-prodag R {*j*. *j* ≤ (*Suc 0*)} *S B*
⟨proof⟩

lemma (**in** *aGroup*) *restrict-prod-Suc*:⟦∀ *k* ≤ (*Suc* (*Suc n*)). *ideal R* (*J k*);
∀ *k* ≤ (*Suc* (*Suc n*)). *B k* = *R* $/_r$ *J k*;
∀ *k* ≤ (*Suc* (*Suc n*)). *S k* = *pj R* (*J k*);
f ∈ *carrier* ($r\Pi_{\{j.\ j\ \leq\ (Suc\ (Suc\ n))\}}$ *B*)⟧ ⟹
restrict f {*j*. *j* ≤ (*Suc n*)} ∈ *carrier* ($r\Pi_{\{j.\ j\ \leq\ (Suc\ n)\}}$ *B*)
⟨proof⟩

lemma (**in** *Ring*) *Chinese-remTr2*:(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*)) ∧
(∀ *k*≤(*Suc n*). *B k* = *qring R* (*J k*)) ∧
(∀ *k*≤(*Suc n*). *S k* = *pj R* (*J k*)) ∧
(∀ *i*≤(*Suc n*). ∀ *j*≤ (*Suc n*). (*i* ≠*j* ⟶
coprime-ideals R (*J i*) (*J j*))) ⟶
$surjec_{R,(r\Pi_{\{j.\ j\leq\ (Suc\ n)\}}\ B)}$
(*A-to-prodag R* {*j*. *j*≤(*Suc n*)} *S B*)
⟨proof⟩

lemma (**in** *Ring*) *Chinese-remTr3*:⟦∀ *k* ≤ (*Suc n*). *ideal R* (*J k*);
∀ *k* ≤ (*Suc n*). *B k* = *qring R* (*J k*); ∀ *k*≤ (*Suc n*). *S k* = *pj R* (*J k*);
∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). (*i* ≠*j* ⟶ *coprime-ideals R* (*J i*) (*J j*))⟧ ⟹
$surjec_{R,(r\Pi_{\{j.\ j\ \leq\ (Suc\ n)\}}\ B)}$
(*A-to-prodag R* {*j*. *j* ≤ (*Suc n*)} *S B*)
⟨proof⟩

lemma (**in** *Ring*) *imset*:⟦∀ *k*≤ (*Suc n*). *ideal R* (*J k*)⟧
⟹ {*I*. ∃ *k*≤ (*Suc n*). *I* = *J k*} = {*J k*| *k*. *k* ∈ {*j*. *j* ≤ (*Suc n*)}}
⟨proof⟩

theorem (**in** *Ring*) *Chinese-remThm*:⟦(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*));
∀ *k*≤(*Suc n*). *B k* = *qring R* (*J k*); ∀ *k* ≤ (*Suc n*). *S k* = *pj R* (*J k*);
∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). (*i* ≠*j* ⟶ *coprime-ideals R* (*J i*) (*J j*))⟧
⟹ $bijec_{(qring\ R\ (\bigcap\ \{J\ k\ \mid\ k.\ k\in\{j.\ j\ \leq\ (Suc\ n)\}\})),(r\Pi_{\{j.\ j\ \leq\ (Suc\ n)\}}\ B)}$

$((A\text{-}to\text{-}prodag\ R\ \{j.\ j \le (Suc\ n)\}\ S\ B)^{\circ}{}_{R,(prodrg\ \{j.\ j \le (Suc\ n)\}\ B)})$
⟨proof⟩

lemma (**in** *Ring*) *prod-prime*: $[\![ideal\ R\ A;\ \forall k{\le}(Suc\ n).\ prime\text{-}ideal\ R\ (P\ k);$
$\forall l{\le}(Suc\ n).\ \neg\ (A \subseteq P\ l);$
$\forall k{\le}\ (Suc\ n).\ \forall l{\le}\ (Suc\ n).\ k = l \vee \neg\ (P\ k) \subseteq (P\ l)]\!] \Longrightarrow$
$\forall i \le (Suc\ n).\ (nprod\ R\ (ppa\ R\ P\ A\ i)\ n \in A\ \wedge$
$(\forall l \in \{j.\ j{\le}(Suc\ n)\} - \{i\}.\ nprod\ R\ (ppa\ R\ P\ A\ i)\ n \in P\ l)\ \wedge$
$(nprod\ R\ (ppa\ R\ P\ A\ i)\ n \notin P\ i))$
⟨proof⟩

lemma *skip-im1*: $[\![i \le (Suc\ n);\ P \in \{j.\ j \le (Suc\ n)\} \to Collect\ (prime\text{-}ideal\ R)]\!]$
$\Longrightarrow$
$compose\ \{j.\ j \le n\}\ P\ (skip\ i)\ `\ \{j.\ j \le n\} = P\ `\ (\{j.\ j \le (Suc\ n)\} - \{i\})$
⟨proof⟩

lemma (**in** *Ring*) *mutch-aux1*: $[\![ideal\ R\ A;\ i \le (Suc\ n);$
$P \in \{j.\ j \le (Suc\ n)\} \to Collect\ (prime\text{-}ideal\ R)]\!] \Longrightarrow$
$compose\ \{j.\ j \le n\}\ P\ (skip\ i) \in \{j.\ j \le n\} \to Collect\ (prime\text{-}ideal\ R)$
⟨proof⟩

lemma (**in** *Ring*) *prime-ideal-cont1Tr*: $ideal\ R\ A \Longrightarrow$
$\forall P.\ ((P \in \{j.\ j \le (n{::}nat)\} \to \{X.\ prime\text{-}ideal\ R\ X\})\ \wedge$
$(A \subseteq \bigcup\ (P\ `\ \{j.\ j \le n\}))) \longrightarrow (\exists i{\le}\ n.\ A \subseteq (P\ i))$
⟨proof⟩

lemma (**in** *Ring*) *prime-ideal-cont1*: $[\![ideal\ R\ A;\ \forall i \le (n{::}nat).$
$prime\text{-}ideal\ R\ (P\ i);\ A \subseteq \bigcup\ \{X.\ (\exists i \le n.\ X = (P\ i))\}\ ]\!] \Longrightarrow$
$\exists i{\le}\ n.\ A{\subseteq}(P\ i)$
⟨proof⟩

lemma (**in** *Ring*) *prod-n-ideal-contTr0*: $(\forall l{\le}\ n.\ ideal\ R\ (J\ l)) \longrightarrow$
$i\Pi_{R,n}\ J\ \subseteq\ \bigcap\{X.\ (\exists k{\le}n.\ X = (J\ k))\}$
⟨proof⟩

lemma (**in** *Ring*) *prod-n-ideal-contTr*: $[\![\forall l{\le}\ n.\ ideal\ R\ (J\ l)]\!] \Longrightarrow$
$i\Pi_{R,n}\ J\ \subseteq\ \bigcap\{X.\ (\exists k \le n.\ X = (J\ k))\}$
⟨proof⟩

lemma (**in** *Ring*) *prod-n-ideal-cont2*: $[\![\forall l{\le}\ (n{::}nat).\ ideal\ R\ (J\ l);$
$prime\text{-}ideal\ R\ P;\ \bigcap\{X.\ (\exists k{\le}\ n.\ X = (J\ k))\} \subseteq P]\!] \Longrightarrow$
$\exists l{\le}\ n.\ (J\ l) \subseteq P$
⟨proof⟩

lemma (**in** *Ring*) *prod-n-ideal-cont3*: $[\![\forall l{\le}\ (n{::}nat).\ ideal\ R\ (J\ l);$
$prime\text{-}ideal\ R\ P;\ \bigcap\{X.\ (\exists k{\le}\ n.\ X = (J\ k))\} = P]\!] \Longrightarrow$
$\exists l{\le}\ n.\ (J\ l) = P$
⟨proof⟩

definition
ideal-quotient :: [- , 'a set, 'a set] ⇒ 'a set **where**
ideal-quotient R A B = {x| x. x ∈ carrier R ∧ (∀ b∈B. x $\cdot_{rR}$ b ∈ A)}

abbreviation
IDEALQT ((3-/ †-/ -) [82,82,83]82) **where**
A $\dagger_R$ B == ideal-quotient R A B

lemma (**in** *Ring*) *ideal-quotient-is-ideal*:
⟦ideal R A; ideal R B⟧ ⟹ ideal R (ideal-quotient R A B)
⟨proof⟩

4.9 Addition of finite elements of a ring and *ideal-multiplication*

We consider sum in an abelian group

lemma (**in** *aGroup*) *nsum-mem1Tr*: A +> J ⟹
(∀ j ≤ n. f j ∈ J) ⟶ nsum A f n ∈ J
⟨proof⟩

lemma (**in** *aGroup*) *fSum-mem*:⟦∀ j ∈ nset (Suc n) m. f j ∈ carrier A; n < m⟧
⟹
fSum A f (Suc n) m ∈ carrier A
⟨proof⟩

lemma (**in** *aGroup*) *nsum-mem1*:⟦A +> J; ∀ j ≤ n. f j ∈ J⟧ ⟹ nsum A f n ∈ J
⟨proof⟩

lemma (**in** *aGroup*) *nsum-eq-i*:⟦∀ j≤n. f j ∈ carrier A; ∀ j≤n. g j ∈ carrier A;
i ≤ n; ∀ l ≤ i. f l = g l⟧ ⟹ nsum A f i = nsum A g i
⟨proof⟩

lemma (**in** *aGroup*) *nsum-cmp-eq*:⟦f ∈ {j. j≤(n::nat)} → carrier A;
h1 ∈ {j. j ≤ n} → {j. j ≤ n}; h2 ∈ {j. j ≤ n} → {j. j ≤ n}; i ≤ n⟧ ⟹
nsum A (cmp f (cmp h2 h1)) i = nsum A (cmp (cmp f h2) h1) i
⟨proof⟩

lemma (**in** *aGroup*) *nsum-cmp-eq-transpos*:⟦ ∀ j≤(Suc n). f j ∈ carrier A;
i ≤ n;i ≠ n ⟧ ⟹
nsum A (cmp f (cmp (transpos i n) (cmp (transpos n (Suc n)) (transpos i n))))
(Suc n) = nsum A (cmp f (transpos i (Suc n))) (Suc n)
⟨proof⟩

lemma *transpos-Tr-n1*:Suc (Suc 0) ≤ n ⟹
transpos (n − Suc 0) n n = n − Suc 0
⟨proof⟩

lemma *transpos-Tr-n2*:Suc (Suc 0) ≤ n ⟹

$\quad transpos\ (n - (Suc\ 0))\ n\ (n - (Suc\ 0)) = n$
⟨proof⟩

lemma (**in** *aGroup*) *additionTr0*: $[\![0 < n; \forall j \le n.\ f\ j \in carrier\ A]\!]$
$\Longrightarrow nsum\ A\ (cmp\ f\ (transpos\ (n - 1)\ n))\ n = nsum\ A\ f\ n$
⟨proof⟩

lemma (**in** *aGroup*) *additionTr1*: $[\![\ \forall f.\ \forall h.\ f \in \{j.\ j \le (Suc\ n)\} \rightarrow carrier\ A \wedge$
$h \in \{j.\ j \le (Suc\ n)\} \rightarrow \{j.\ j \le (Suc\ n)\} \wedge inj\text{-}on\ h\ \{j.\ j \le (Suc\ n)\} \longrightarrow$
$nsum\ A\ (cmp\ f\ h)\ (Suc\ n) = nsum\ A\ f\ (Suc\ n);$
$f \in \{j.\ j \le (Suc\ (Suc\ n))\} \rightarrow carrier\ A;$
$h \in \{j.\ j \le (Suc\ (Suc\ n))\} \rightarrow \{j.\ j \le (Suc\ (Suc\ n))\};$
$inj\text{-}on\ h\ \{j.\ j \le (Suc\ (Suc\ n))\};\ h\ (Suc\ (Suc\ n)) = Suc\ (Suc\ n)]\!]$
$\Longrightarrow nsum\ A\ (cmp\ f\ h)\ (Suc\ (Suc\ n)) = nsum\ A\ f\ (Suc\ (Suc\ n))$
⟨proof⟩

lemma (**in** *aGroup*) *additionTr1-1*: $[\![\forall f.\ \forall h.\ f \in \{j.\ j \le Suc\ n\} \rightarrow carrier\ A \wedge$
$h \in \{j.\ j \le Suc\ n\} \rightarrow \{j.\ j \le Suc\ n\} \wedge inj\text{-}on\ h\ \{j.\ j \le Suc\ n\} \longrightarrow$
$nsum\ A\ (cmp\ f\ h)\ (Suc\ n) = nsum\ A\ f\ (Suc\ n);$
$f \in \{j.\ j \le Suc\ (Suc\ n)\} \rightarrow carrier\ A;\ i \le n]\!] \Longrightarrow$
$nsum\ A\ (cmp\ f\ (transpos\ i\ (Suc\ n)))\ (Suc\ (Suc\ n)) = nsum\ A\ f\ (Suc\ (Suc\ n))$
⟨proof⟩

lemma (**in** *aGroup*) *additionTr1-2*: $[\![\forall f.\ \forall h.\ f \in \{j.\ j \le Suc\ n\} \rightarrow carrier\ A \wedge$
$h \in \{j.\ j \le Suc\ n\} \rightarrow \{j.\ j \le Suc\ n\} \wedge$
$inj\text{-}on\ h\ \{j.\ j \le Suc\ n\} \longrightarrow$
$nsum\ A\ (cmp\ f\ h)\ (Suc\ n) = nsum\ A\ f\ (Suc\ n);$
$f \in \{j.\ j \le\ Suc\ (Suc\ n)\} \rightarrow carrier\ A;\ i \le (Suc\ n)]\!] \Longrightarrow$
$nsum\ A\ (cmp\ f\ (transpos\ i\ (Suc\ (Suc\ n))))\ (Suc\ (Suc\ n)) =$
$nsum\ A\ f\ (Suc\ (Suc\ n))$
⟨proof⟩

lemma (**in** *aGroup*) *additionTr2*: $\forall f.\ \forall h.\ f \in \{j.\ j \le (Suc\ n)\} \rightarrow carrier\ A \wedge$
$h \in \{j.\ j \le (Suc\ n)\} \rightarrow \{j.\ j \le (Suc\ n)\} \wedge$
$inj\text{-}on\ h\ \{j.\ j \le (Suc\ n)\} \longrightarrow$
$nsum\ A\ (cmp\ f\ h)\ (Suc\ n) = nsum\ A\ f\ (Suc\ n)$
⟨proof⟩

lemma (**in** *aGroup*) *addition2*: $[\![f \in \{j.\ j \le (Suc\ n)\} \rightarrow carrier\ A;$
$h \in \{j.\ j \le (Suc\ n)\} \rightarrow \{j.\ j \le (Suc\ n)\};\ inj\text{-}on\ h\ \{j.\ j \le (Suc\ n)\}]\!] \Longrightarrow$
$nsum\ A\ (cmp\ f\ h)\ (Suc\ n) = nsum\ A\ f\ (Suc\ n)$
⟨proof⟩

lemma (**in** *aGroup*) *addition21*: $[\![f \in \{j.\ j \le n\} \rightarrow carrier\ A;$
$h \in \{j.\ j \le n\} \rightarrow \{j.\ j \le n\};\ inj\text{-}on\ h\ \{j.\ j \le n\}]\!] \Longrightarrow$
$nsum\ A\ (cmp\ f\ h)\ n = nsum\ A\ f\ n$
⟨proof⟩

lemma (**in** *aGroup*) *addition3*: $[\![\forall j \le (Suc\ n).\ f\ j \in carrier\ A;\ j \le (Suc\ n);$

$j \neq Suc\ n\rrbracket \Longrightarrow nsum\ A\ f\ (Suc\ n) = nsum\ A\ (cmp\ f\ (transpos\ j\ (Suc\ n)))\ (Suc\ n)$
⟨proof⟩

lemma (**in** *aGroup*) *nsum-splitTr*: $(\forall j \leq (Suc\ (n + m)).\ f\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ f\ (Suc\ (n + m)) = nsum\ A\ f\ n \pm (nsum\ A\ (cmp\ f\ (slide\ (Suc\ n)))\ m)$
⟨proof⟩

lemma (**in** *aGroup*) *nsum-split*: $\forall j \leq (Suc\ (n + m)).\ f\ j \in carrier\ A \Longrightarrow$
$nsum\ A\ f\ (Suc\ (n + m)) = nsum\ A\ f\ n \pm (nsum\ A\ (cmp\ f\ (slide\ (Suc\ n)))\ m)$

⟨proof⟩

lemma (**in** *aGroup*) *nsum-split1*: $\llbracket\forall j \leq m.\ f\ j \in carrier\ A;\ n < m\rrbracket \Longrightarrow$
$nsum\ A\ f\ m = nsum\ A\ f\ n \pm (fSum\ A\ f\ (Suc\ n)\ m)$
⟨proof⟩

lemma (**in** *aGroup*) *nsum-minusTr*: $(\forall j \leq n.\ f\ j \in carrier\ A) \longrightarrow$
$-_a\ (nsum\ A\ f\ n) = nsum\ A\ (\lambda x{\in}\{j.\ j \leq n\}.\ -_a\ (f\ x))\ n$
⟨proof⟩

lemma (**in** *aGroup*) *nsum-minus*: $\forall j \leq n.\ f\ j \in carrier\ A \Longrightarrow$
$-_a\ (nsum\ A\ f\ n) = nsum\ A\ (\lambda x{\in}\{j.\ j \leq n\}.\ -_a\ (f\ x))\ n$
⟨proof⟩

lemma (**in** *aGroup*) *ring-nsum-zeroTr*: $(\forall j \leq (n{::}nat).\ f\ j \in carrier\ A) \wedge$
$(\forall j \leq n.\ f\ j = \mathbf{0}) \longrightarrow nsum\ A\ f\ n = \mathbf{0}$
⟨proof⟩

lemma (**in** *aGroup*) *ring-nsum-zero*: $\forall j \leq (n{::}nat).\ f\ j = \mathbf{0} \Longrightarrow \Sigma_e\ A\ f\ n = \mathbf{0}$
⟨proof⟩

lemma (**in** *aGroup*) *ag-nsum-1-nonzeroTr*:
$\forall f.\ (\forall j \leq n.\ f\ j \in carrier\ A) \wedge$
$(l \leq n \wedge (\forall j \in \{j.\ j \leq n\} - \{l\}.\ f\ j = \mathbf{0}))$
$\longrightarrow nsum\ A\ f\ n = f\ l$
⟨proof⟩

lemma (**in** *aGroup*) *ag-nsum-1-nonzero*: $\llbracket\forall j \leq n.\ f\ j \in carrier\ A;\ l \leq n;$
$\forall j{\in}(\{j.\ j \leq n\} - \{l\}).\ f\ j = \mathbf{0}\ \rrbracket \Longrightarrow nsum\ A\ f\ n = f\ l$
⟨proof⟩

definition
set-mult :: [- , 'a set, 'a set] ⇒ 'a set **where**
$set\text{-}mult\ R\ A\ B = \{z.\ \exists x{\in}A.\ \exists y{\in}B.\ x \cdot_{r R}\ y = z\}$

definition
sum-mult :: [- , 'a set, 'a set] ⇒ 'a set **where**
$sum\text{-}mult\ R\ A\ B = \{x.\ \exists n.\ \exists f \in \{j.\ j \leq (n{::}nat)\}$

$\rightarrow$ *set-mult R A B*. *nsum R f n* = *x*}

lemma (**in** *Ring*) *set-mult-sub*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*⟧ $\Longrightarrow$
set-mult R A B $\subseteq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-mono*:⟦*A1* $\subseteq$ *carrier R*; *A2* $\subseteq$ *carrier R*; *A1* $\subseteq$ *A2*;
$B \subseteq$ *carrier R*⟧ $\Longrightarrow$ *set-mult R A1 B* $\subseteq$ *set-mult R A2 B*
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-Tr1*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*⟧ $\Longrightarrow$
($\forall j \leq n$. *f j* $\in$ *set-mult R A B*) $\longrightarrow$ *nsum R f n* $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-mem*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*;
$\forall j \leq n$. *f j* $\in$ *set-mult R A B*⟧ $\Longrightarrow$ *nsum R f n* $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-mem1*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*;
$x \in$ *sum-mult R A B*⟧ $\Longrightarrow$
$\exists n$. $\exists f \in \{j.\ j \leq (n::nat)\}$ $\rightarrow$ *set-mult R A B*. *nsum R f n* = *x*
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-subR*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*⟧ $\Longrightarrow$
sum-mult R A B $\subseteq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *times-mem-sum-mult*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*;
$a \in A$; $b \in B$ ⟧ $\Longrightarrow$ $a \cdot_r b \in$ *sum-mult R A B*
⟨*proof*⟩

lemma (**in** *Ring*) *mem-minus-sum-multTr2*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*;
$\forall j \leq n$. *f j* $\in$ *set-mult R A B*; $i \leq n$⟧ $\Longrightarrow$ *f i* $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-jointfun*:⟦$\forall j \leq n$. *f j* $\in$ *carrier A*;
$\forall j \leq m$. *g j* $\in$ *carrier A*⟧ $\Longrightarrow$
Σ_e *A* (*jointfun n f m g*) (*Suc* (*n* + *m*)) = Σ_e *A f n* $\pm$ (Σ_e *A g m*)
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-pOp-closed*:⟦$A \subseteq$ *carrier R*; $B \subseteq$ *carrier R*;
$a \in$ *sum-mult R A B*; $b \in$ *sum-mult R A B* ⟧ $\Longrightarrow$ $a \pm_R b \in$ *sum-mult R A*
B
⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-mOp-closed*:⟦$A \subseteq$ *carrier R*; *ideal R B*;
$x \in$ *set-mult R A B*⟧ $\Longrightarrow$ $-_a\ x \in$ *set-mult R A B*
⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-ring-times-closed*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B;$
$x \in set\text{-}mult\ R\ A\ B;\ r \in carrier\ R]\!] \Longrightarrow r \cdot_r\ x \in set\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-sub-sum-mult*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B]\!] \Longrightarrow$
$set\text{-}mult\ R\ A\ B \subseteq sum\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-pOp-closedn*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B]\!] \Longrightarrow$
$(\forall j \le n.\ f\ j \in set\text{-}mult\ R\ A\ B) \longrightarrow \Sigma_e\ R\ f\ n \in sum\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *mem-minus-sum-multTr4*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B]\!] \Longrightarrow$
$(\forall j \le n.\ f\ j \in set\text{-}mult\ R\ A\ B) \longrightarrow -_a\ (nsum\ R\ f\ n) \in sum\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-iOp-closed*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B;$
$x \in sum\text{-}mult\ R\ A\ B\ ]\!] \Longrightarrow -_a\ x \in sum\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-ring-multiplicationTr*:
$[\![A \subseteq carrier\ R;\ ideal\ R\ B;\ r \in carrier\ R]\!] \Longrightarrow$
$(\forall j \le n.\ f\ j \in set\text{-}mult\ R\ A\ B) \longrightarrow r \cdot_r\ (nsum\ R\ f\ n) \in sum\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-ring-multiplication*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B;$
$r \in carrier\ R;\ a \in sum\text{-}mult\ R\ A\ B]\!] \Longrightarrow r \cdot_r\ a \in sum\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-sum-mult*:$[\![A \subseteq carrier\ R;\ A \neq \{\};\ ideal\ R\ B]\!] \Longrightarrow$
$ideal\ R\ (sum\text{-}mult\ R\ A\ B)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-set-multTr*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B;\ ideal\ R\ C;$
$set\text{-}mult\ R\ A\ B \subseteq C\ ]\!] \Longrightarrow$
$\forall f \in \{j.\ j \le (n::nat)\} \to set\text{-}mult\ R\ A\ B.\ \Sigma_e\ R\ f\ n \in C$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-set-mult*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B;\ ideal\ R\ C;$
$set\text{-}mult\ R\ A\ B \subseteq C\ ]\!] \Longrightarrow sum\text{-}mult\ R\ A\ B \subseteq C$
⟨*proof*⟩

lemma (**in** *Ring*) *AB-inc-sum-mult*:$[\![ideal\ R\ A;\ ideal\ R\ B]\!] \Longrightarrow$
$sum\text{-}mult\ R\ A\ B \subseteq A \cap B$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-is-ideal-prod*:$[\![ideal\ R\ A;\ ideal\ R\ B]\!] \Longrightarrow$
$sum\text{-}mult\ R\ A\ B = A \diamondsuit_r\ B$

$\langle proof \rangle$

lemma (**in** *Ring*) *ideal-prod-assocTr0*: $[\![ideal\ R\ A;\ ideal\ R\ B;\ ideal\ R\ C;\ y \in C;$
$z \in set\text{-}mult\ R\ A\ B]\!] \Longrightarrow z \cdot_r y \in sum\text{-}mult\ R\ A\ (B \diamondsuit_r C)$
$\langle proof \rangle$

lemma (**in** *Ring*) *ideal-prod-assocTr1*: $[\![ideal\ R\ A;\ ideal\ R\ B;\ ideal\ R\ C;\ y \in C]\!]$
$\Longrightarrow \forall f \in \{j.\ j \le (n::nat)\} \rightarrow set\text{-}mult\ R\ A\ B.\ (\Sigma_e\ R\ f\ n) \cdot_r y \in A \diamondsuit_r (B \diamondsuit_r C)$
$\langle proof \rangle$

lemma (**in** *Ring*) *ideal-quotient-idealTr*: $[\![ideal\ R\ A;\ ideal\ R\ B;\ ideal\ R\ C;$
$x \in carrier\ R; \forall c \in C.\ x \cdot_r c \in ideal\text{-}quotient\ R\ A\ B]\!] \Longrightarrow$
$f \in \{j.\ j \le n\} \rightarrow set\text{-}mult\ R\ B\ C \longrightarrow x \cdot_r (nsum\ R\ f\ n) \in A$

$\langle proof \rangle$

lemma (**in** *Ring*) *ideal-quotient-ideal*: $[\![ideal\ R\ A;\ ideal\ R\ B;\ ideal\ R\ C]\!] \Longrightarrow$
$A \dagger_R B \dagger_R C = A \dagger_R B \diamondsuit_r C$
$\langle proof \rangle$

lemma (**in** *Ring*) *ideal-prod-assocTr*: $[\![ideal\ R\ A;\ ideal\ R\ B;\ ideal\ R\ C]\!] \Longrightarrow$
$\forall f.\ (f \in \{j.\ j \le (n::nat)\} \rightarrow set\text{-}mult\ R\ (A \diamondsuit_r B)\ C \longrightarrow$
$(\Sigma_e\ R\ f\ n) \in A \diamondsuit_r (B \diamondsuit_r C))$
$\langle proof \rangle$

lemma (**in** *Ring*) *ideal-prod-assoc*: $[\![ideal\ R\ A;\ ideal\ R\ B;\ ideal\ R\ C]\!] \Longrightarrow$
$(A \diamondsuit_r B) \diamondsuit_r C = A \diamondsuit_r (B \diamondsuit_r C)$
$\langle proof \rangle$

lemma (**in** *Ring*) *prod-principal-idealTr0*: $[\![a \in carrier\ R;\ b \in carrier\ R;$
$z \in set\text{-}mult\ R\ (R \diamondsuit_p a)\ (R \diamondsuit_p b)]\!] \Longrightarrow z \in R \diamondsuit_p (a \cdot_r b)$
$\langle proof \rangle$

lemma (**in** *Ring*) *prod-principal-idealTr1*: $[\![a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow$
$\forall f \in \{j.\ j \le (n::nat)\} \rightarrow set\text{-}mult\ R\ (R \diamondsuit_p a)\ (R \diamondsuit_p b).$
$\Sigma_e\ R\ f\ n \in R \diamondsuit_p (a \cdot_r b)$
$\langle proof \rangle$

lemma (**in** *Ring*) *prod-principal-ideal*: $[\![a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow$
$(Rxa\ R\ a) \diamondsuit_r (Rxa\ R\ b) = Rxa\ R\ (a \cdot_r b)$
$\langle proof \rangle$

lemma (**in** *Ring*) *principal-ideal-n-pow1*: $a \in carrier\ R \Longrightarrow$
$(Rxa\ R\ a)^{\diamondsuit R\ n} = Rxa\ R\ (a^{\hat{}R\ n})$
$\langle proof \rangle$

lemma (**in** *Ring*) *principal-ideal-n-pow*: $[\![a \in carrier\ R;\ I = Rxa\ R\ a]\!] \Longrightarrow$
$I^{\diamondsuit R\ n} = Rxa\ R\ (a^{\hat{}R\ n})$

$\langle proof \rangle$

more about *ideal-n-prod*

lemma (**in** *Ring*) *nprod-eqTr*: $f \in \{j.\ j \leq (n::nat)\} \rightarrow carrier\ R\ \wedge$
$g \in \{j.\ j \leq n\} \rightarrow carrier\ R \wedge (\forall j \leq n.\ f\ j = g\ j) \longrightarrow$
$nprod\ R\ f\ n = nprod\ R\ g\ n$
$\langle proof \rangle$

lemma (**in** *Ring*) *nprod-eq*: $[\![\forall j \leq n.\ f\ j \in carrier\ R;\ \forall j \leq n.\ g\ j \in carrier\ R;$
$(\forall j \leq (n::nat).\ f\ j = g\ j)]\!] \Longrightarrow nprod\ R\ f\ n = nprod\ R\ g\ n$
$\langle proof \rangle$

definition
$mprod\text{-}expR :: [('b,\ 'm)\ Ring\text{-}scheme,\ nat \Rightarrow nat,\ nat \Rightarrow 'b,\ nat] \Rightarrow 'b$ **where**
$mprod\text{-}expR\ R\ e\ f\ n = nprod\ R\ (\lambda j.\ ((f\ j)\ \hat{}^{R\ (e\ j)}))\ n$

lemma (**in** *Ring*) *mprodR-Suc*: $[\![e \in \{j.\ j \leq (Suc\ n)\} \rightarrow \{j.\ (0::nat) \leq j\};$
$f \in \{j.\ j \leq (Suc\ n)\} \rightarrow carrier\ R]\!] \Longrightarrow$
$mprod\text{-}expR\ R\ e\ f\ (Suc\ n) =$
$(mprod\text{-}expR\ R\ e\ f\ n) \cdot_r ((f\ (Suc\ n))\ \hat{}^{R\ (e\ (Suc\ n))})$
$\langle proof \rangle$

lemma (**in** *Ring*) *mprod-expR-memTr*: $e \in \{j.\ j \leq n\} \rightarrow \{j.\ (0::nat) \leq j\}\ \wedge$
$f \in \{j.\ j \leq n\} \rightarrow carrier\ R\ \longrightarrow\ mprod\text{-}expR\ R\ e\ f\ n \in carrier\ R$
$\langle proof \rangle$

lemma (**in** *Ring*) *mprod-expR-mem*: $[\![\ e \in \{j.\ j \leq n\} \rightarrow \{j.\ (0::nat) \leq j\};$
$f \in \{j.\ j \leq n\} \rightarrow carrier\ R]\!]\ \Longrightarrow\ mprod\text{-}expR\ R\ e\ f\ n \in carrier\ R$
$\langle proof \rangle$

lemma (**in** *Ring*) *prod-n-principal-idealTr*: $e \in \{j.\ j \leq n\} \rightarrow \{j.\ (0::nat) \leq j\}\ \wedge$
$f \in \{j.\ j \leq n\} \rightarrow carrier\ R \wedge (\forall k \leq n.\ J\ k = (Rxa\ R\ (f\ k))^{\diamondsuit R\ (e\ k)}) \longrightarrow$
$ideal\text{-}n\text{-}prod\ R\ n\ J = Rxa\ R\ (mprod\text{-}expR\ R\ e\ f\ n)$
$\langle proof \rangle$

lemma (**in** *Ring*) *prod-n-principal-ideal*: $[\![e \in \{j.\ j \leq n\} \rightarrow \{j.\ (0::nat) \leq j\};$
$f \in \{j.\ j \leq n\} \rightarrow carrier\ R;\ \forall k \leq n.\ J\ k = (Rxa\ R\ (f\ k))^{\diamondsuit R\ (e\ k)}]\!] \Longrightarrow$
$ideal\text{-}n\text{-}prod\ R\ n\ J = Rxa\ R\ (mprod\text{-}expR\ R\ e\ f\ n)$
$\langle proof \rangle$

lemma (**in** *Idomain*) *a-notin-n-pow1*: $[\![a \in carrier\ R;\ \neg\ Unit\ R\ a;\ a \neq \mathbf{0};\ 0 < n]\!]$
$\Longrightarrow\ a \notin (Rxa\ R\ a)\ ^{\diamondsuit R\ \ (Suc\ n)}$
$\langle proof \rangle$

lemma (**in** *Idomain*) *a-notin-n-pow2*: $[\![a \in carrier\ R;\ \neg\ Unit\ R\ a;\ a \neq \mathbf{0};$

0 < *n*⟧ ⟹ a ^$^{R\ n}$ ∉ (*Rxa R a*) $^{\diamondsuit R\ (Suc\ n)}$
⟨*proof*⟩

lemma (**in** *Idomain*) *n-pow-not-prime*:⟦*a* ∈ *carrier R*; *a* ≠ **0**; *0* < *n*⟧
⟹ ¬ *prime-ideal R* ((*Rxa R a*) $^{\diamondsuit R\ (Suc\ n)}$)
⟨*proof*⟩

lemma (**in** *Idomain*) *principal-pow-prime-condTr*:
⟦*a* ∈ *carrier R*; *a* ≠ **0**; *prime-ideal R* ((*Rxa R a*) $^{\diamondsuit R\ (Suc\ n)}$)⟧ ⟹ *n* = *0*
⟨*proof*⟩

lemma (**in** *Idomain*) *principal-pow-prime-cond*:
⟦*a* ∈ *carrier R*; *a* ≠ **0**; *prime-ideal R* ((*Rxa R a*) $^{\diamondsuit R\ n}$)⟧ ⟹ *n* = *Suc 0*
⟨*proof*⟩

4.10 Extension and contraction

locale *TwoRings* = *Ring* +
fixes *R'* (**structure**)
assumes *secondR*: *Ring R'*

definition
i-contract :: [*'a* ⇒ *'b*, (*'a*, *'m1*) *Ring-scheme*, (*'b*, *'m2*) *Ring-scheme*,
'b set] ⇒ *'a set* **where**
i-contract f R R' J = *invim f* (*carrier R*) *J*

definition
i-extension :: [*'a* ⇒ *'b*, (*'a*, *'m1*) *Ring-scheme*, (*'b*, *'m2*) *Ring-scheme*,
'a set] ⇒ *'b set* **where**
i-extension f R R' I = *sum-mult R'* (*f* ' *I*) (*carrier R'*)

lemma (**in** *TwoRings*) *i-contract-sub*:⟦*f* ∈ *rHom R R'*; *ideal R' J* ⟧ ⟹
(*i-contract f R R' J*) ⊆ *carrier R*
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-contract-ideal*:⟦*f* ∈ *rHom R R'*; *ideal R' J* ⟧ ⟹
ideal R (*i-contract f R R' J*)
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-contract-mono*:⟦*f* ∈ *rHom R R'*; *ideal R' J1*; *ideal R' J2*;
J1 ⊆ *J2* ⟧ ⟹ *i-contract f R R' J1* ⊆ *i-contract f R R' J2*
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-contract-prime*:⟦*f* ∈ *rHom R R'*; *prime-ideal R' P*⟧ ⟹
prime-ideal R (*i-contract f R R' P*)
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-extension-ideal*:$[\![f \in rHom\ R\ R';\ ideal\ R\ I\ ]\!] \Longrightarrow$
$ideal\ R'\ (i\text{-}extension\ f\ R\ R'\ I)$
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-extension-mono*:$[\![f \in rHom\ R\ R';\ ideal\ R\ I1;\ ideal\ R\ I2;$
$I1 \subseteq I2\ ]\!] \Longrightarrow (i\text{-}extension\ f\ R\ R'\ I1) \subseteq (i\text{-}extension\ f\ R\ R'\ I2)$
⟨*proof*⟩

lemma (**in** *TwoRings*) *e-c-inc-self*:$[\![f \in rHom\ R\ R';\ ideal\ R\ I]\!] \Longrightarrow$
$I \subseteq i\text{-}contract\ f\ R\ R'\ (i\text{-}extension\ f\ R\ R'\ I)$
⟨*proof*⟩

lemma (**in** *TwoRings*) *c-e-incd-self*:$[\![f \in rHom\ R\ R';\ ideal\ R'\ J\ ]\!] \Longrightarrow$
$i\text{-}extension\ f\ R\ R'\ (i\text{-}contract\ f\ R\ R'\ J) \subseteq J$
⟨*proof*⟩

lemma (**in** *TwoRings*) *c-e-c-eq-c*:$[\![f \in rHom\ R\ R';\ ideal\ R'\ J\ ]\!] \Longrightarrow$
$i\text{-}contract\ f\ R\ R'\ (i\text{-}extension\ f\ R\ R'\ (i\text{-}contract\ f\ R\ R'\ J))$
$= i\text{-}contract\ f\ R\ R'\ J$
⟨*proof*⟩

lemma (**in** *TwoRings*) *e-c-e-eq-e*:$[\![f \in rHom\ R\ R';\ ideal\ R\ I\ ]\!] \Longrightarrow$
$i\text{-}extension\ f\ R\ R'\ (i\text{-}contract\ f\ R\ R'\ (i\text{-}extension\ f\ R\ R'\ I))$
$= i\text{-}extension\ f\ R\ R'\ I$
⟨*proof*⟩

4.11 Complete system of representatives

definition
csrp-fn :: [-, *'a set*] ⇒ *'a set* ⇒ *'a* **where**
$csrp\text{-}fn\ R\ I = (\lambda x{\in}carrier\ (R\ /_r\ I).\ (if\ x = I\ then\ \mathbf{0}_R\ else\ SOME\ y.\ y \in x))$

definition
csrp :: [- , *'a set*] ⇒ *'a set* **where**
$csrp\ R\ I == (csrp\text{-}fn\ R\ I)\ `\ (carrier\ (R\ /_r\ I))$

lemma (**in** *Ring*) *csrp-mem*:$[\![ideal\ R\ I;\ a \in carrier\ R]\!] \Longrightarrow$
$csrp\text{-}fn\ R\ I\ (a \uplus_R I) \in a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-same*:$[\![ideal\ R\ I;\ a \in carrier\ R]\!] \Longrightarrow$
$csrp\text{-}fn\ R\ I\ (a \uplus_R I) \uplus_R I = a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-mem1*:$[\![ideal\ R\ I;\ x \in carrier\ (R\ /_r\ I)]\!] \Longrightarrow$
$csrp\text{-}fn\ R\ I\ x \in x$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-fn-mem*:$[\![ideal\ R\ I;\ x \in carrier\ (R\ /_r\ I)]\!] \Longrightarrow$
$(csrp\text{-}fn\ R\ I\ x) \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-eq-coset*:$[\![ideal\ R\ I;\ x \in carrier\ (R\ /_r\ I)]\!] \Longrightarrow$
$(csrp\text{-}fn\ R\ I\ x) \uplus_R I = x$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-nz-nz*:$[\![ideal\ R\ I;\ x \in carrier\ (R\ /_r\ I);$
$x \neq \mathbf{0}_{(R\ /_r\ I)}]\!] \Longrightarrow (csrp\text{-}fn\ R\ I\ x) \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-diff-in-vpr*:$[\![ideal\ R\ I;\ x \in carrier\ R]\!] \Longrightarrow$
$x \pm (-_a\ (csrp\text{-}fn\ R\ I\ (pj\ R\ I\ x))) \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-pj*:$[\![ideal\ R\ I;\ x \in carrier\ (R\ /_r\ I)]\!] \Longrightarrow$
$(pj\ R\ I)\ (csrp\text{-}fn\ R\ I\ x) = x$
⟨*proof*⟩

4.12 Polynomial ring

In this section, we treat a ring of polynomials over a ring S. Numbers are of type ant

definition
pol-coeff :: $[('a,\ 'more)\ Ring\text{-}scheme,\ (nat \times (nat \Rightarrow 'a))] \Rightarrow bool$ **where**
$pol\text{-}coeff\ S\ c \longleftrightarrow (\forall j \leq (fst\ c).\ (snd\ c)\ j \in carrier\ S)$

definition
c-max :: $[('a,\ 'more)\ Ring\text{-}scheme,\ nat \times (nat \Rightarrow 'a)] \Rightarrow nat$ **where**
$c\text{-}max\ S\ c = (if\ \{j.\ j \leq (fst\ c) \wedge (snd\ c)\ j \neq \mathbf{0}_S\} = \{\}\ then\ 0\ else$
$n\text{-}max\ \{j.\ j \leq (fst\ c) \wedge (snd\ c)\ j \neq \mathbf{0}_S\})$

definition
polyn-expr :: $[('a,\ 'more)\ Ring\text{-}scheme,\ 'a,\ nat,\ nat \times (nat \Rightarrow 'a)] \Rightarrow 'a$ **where**
$polyn\text{-}expr\ R\ X\ k\ c == nsum\ R\ (\lambda j.\ ((snd\ c)\ j) \cdot_{rR} (X\hat{\ }^{R\ j}))\ k$

definition
algfree-cond :: $[('a,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'm1)\ Ring\text{-}scheme,$
$'a] \Rightarrow bool$ **where**
$algfree\text{-}cond\ R\ S\ X \longleftrightarrow (\forall c.\ pol\text{-}coeff\ S\ c \wedge (\forall k \leq (fst\ c).$
$(nsum\ R\ (\lambda j.\ ((snd\ c)\ j) \cdot_{rR} (X\hat{\ }^{R\ j}))\ k = \mathbf{0}_R \longrightarrow$
$(\forall j \leq k.\ (snd\ c)\ j = \mathbf{0}_S))))$

locale *PolynRg* = *Ring* +
fixes *S* (**structure**)

fixes X (**structure**)
assumes *X-mem-R*:$X \in$ *carrier R*
and *not-zeroring*:$\neg$ *Zero-ring S*
and *subring*: *Subring R S*
and *algfree*: *algfree-cond R S X*
and *S-X-generate*:$x \in$ *carrier R* $\Longrightarrow$
$\exists f.$ *pol-coeff S f* $\wedge$ $x =$ *polyn-expr R X* (*fst f*) *f*

4.13 Addition and multiplication of *polyn-exprs*

4.13.1 Simple properties of a *polyn-ring*

lemma *Subring-subset*:*Subring R S* $\Longrightarrow$ *carrier S* $\subseteq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *subring-Ring*:*Subring R S* $\Longrightarrow$ *Ring S*
⟨*proof*⟩

lemma (**in** *Ring*) *mem-subring-mem-ring*:⟦*Subring R S*; $x \in$ *carrier S*⟧ $\Longrightarrow$
$x \in$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-pOp-ring-pOp*:⟦*Subring R S*; $a \in$ *carrier S*;
$b \in$ *carrier S* ⟧ $\Longrightarrow$ $a \pm_S b = a \pm b$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-tOp-ring-tOp*:⟦*Subring R S*; $a \in$ *carrier S*;
$b \in$ *carrier S* ⟧ $\Longrightarrow$ $a \cdot_{r S} b = a \cdot_r b$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-one-ring-one*:*Subring R S* $\Longrightarrow$ $1_{r S} = 1_r$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-zero-ring-zero*:*Subring R S* $\Longrightarrow$ $\mathbf{0}_S = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-minus-ring-minus*:⟦*Subring R S*; $x \in$ *carrier S*⟧
$\Longrightarrow$ $-_{a S}\ x = -_a\ x$
⟨*proof*⟩

lemma (**in** *PolynRg*) *Subring-pow-ring-pow*:$x \in$ *carrier S* $\Longrightarrow$
$x^{\hat{}S\ n} = x^{\hat{}R\ n}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *is-Ring*: *Ring R* ⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-nonzero*:$1_r \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-S-nonzero*: $1_{r\,S} \neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-X-nonzero*: $X \neq \mathbf{0}$
⟨*proof*⟩

4.13.2 Coefficients of a polynomial

lemma (**in** *PolynRg*) *pol-coeff-split*: *pol-coeff S f* = *pol-coeff S* (*fst f*, *snd f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-cartesian*: *pol-coeff S c* ⟹
(*fst c*, *snd c*) = *c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *split-pol-coeff*: ⟦*pol-coeff S c*; $k \leq$ (*fst c*)⟧ ⟹
pol-coeff S (*k*, *snd c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-pre*: *pol-coeff S* ((*Suc n*), *f*) ⟹
pol-coeff S (*n*, *f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-le*: ⟦*pol-coeff S c*; $n \leq$ (*fst c*)⟧ ⟹
pol-coeff S (*n*, (*snd c*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-mem*: ⟦*pol-coeff S c*; $j \leq$ (*fst c*)⟧ ⟹
((*snd c*) *j*) ∈ *carrier S*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-mem-R*: ⟦*pol-coeff S c*; $j \leq$ (*fst c*)⟧
⟹ ((*snd c*) *j*) ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *Slide-pol-coeff*: ⟦*pol-coeff S c*; $n <$ (*fst c*)⟧ ⟹
pol-coeff S (((*fst c*) − *Suc n*), (λ*x*. (*snd c*) (*Suc* (*n* + *x*))))
⟨*proof*⟩

4.13.3 Addition of *polyn-exprs*

lemma (**in** *PolynRg*) *monomial-mem*: *pol-coeff S c* ⟹
$\forall j \leq$ (*fst c*). (*snd c*) *j* $\cdot_r$ $X^{\hat{}R\,j}$ ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-mem*: ⟦*pol-coeff S c*; $k \leq$ (*fst c*)⟧ ⟹
polyn-expr R X k c ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-exprs-eq*: ⟦*pol-coeff S c*; *pol-coeff S d*;

$k \leq (min\ (fst\ c)\ (fst\ d))$; $\forall j \leq k.\ (snd\ c)\ j = (snd\ d)\ j$⟧ $\Longrightarrow$
polyn-expr R X k c = *polyn-expr R X k d*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-restrict*:*pol-coeff S* (*Suc n*, *f*) $\Longrightarrow$
polyn-expr R X n (*Suc n*, *f*) = *polyn-expr R X n* (*n*, *f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-short*:⟦*pol-coeff S c*; $k \leq (fst\ c)$⟧ $\Longrightarrow$
polyn-expr R X k c = *polyn-expr R X k* (*k*, *snd c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr0*:*pol-coeff S c* $\Longrightarrow$
polyn-expr R X 0 c = (*snd c*) *0*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-split*:
polyn-expr R X k f = *polyn-expr R X k* (*fst f*, *snd f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-Suc*:*Suc n* $\leq$ (*fst c*) $\Longrightarrow$
polyn-expr R X (*Suc n*) ((*Suc n*), (*snd c*)) =
polyn-expr R X n c $\pm$ ((*snd c*) (*Suc n*)) $\cdot_r$ $(X^{\wedge R\ (Suc\ n)})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-Suc-split*:*pol-coeff S* (*Suc n*, *f*) $\Longrightarrow$
polyn-expr R X (*Suc n*) ((*Suc n*), *f*) =
polyn-expr R X n (*n*, *f*) $\pm$ (*f* (*Suc n*)) $\cdot_r$ $(X^{\wedge R\ (Suc\ n)})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-m*:⟦*pol-coeff S c*; $n < m$; $m \leq (fst\ c)$⟧ $\Longrightarrow$
polyn-expr R X m (*m*, (*snd c*)) = *polyn-expr R X n* (*n*, (*snd c*)) $\pm$
(*fSum R* (λj. ((*snd c*) *j*) $\cdot_r$ $(X^{\wedge R\ j})$) (*Suc n*) *m*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-m1*:⟦*pol-coeff S c*; $n < m$; $m \leq (fst\ c)$⟧ $\Longrightarrow$
polyn-expr R X m c = *polyn-expr R X n c* $\pm$
(*fSum R* (λj. ((*snd c*) *j*) $\cdot_r$ $(X^{\wedge R\ j})$) (*Suc n*) *m*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-m-mem*:⟦*pol-coeff S c*; $n < m$; $m \leq (fst\ c)$⟧ $\Longrightarrow$
(*fSum R* (λj. ((*snd c*) *j*) $\cdot_r$ $(X^{\wedge R\ j})$) (*Suc n*) *m*) $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-ms-eq*:⟦*pol-coeff S c*; *pol-coeff S d*;
$m \leq min\ (fst\ c)\ (fst\ d)$; $n < m$;
$\forall j \in nset\ (Suc\ n)\ m.\ (snd\ c)\ j = (snd\ d)\ j$⟧ $\Longrightarrow$
(*fSum R* (λj. ((*snd c*) *j*) $\cdot_r$ $(X^{\wedge R\ j})$) (*Suc n*) *m*) =

$(fSum\ R\ (\lambda j.\ ((snd\ d)\ j)\ \cdot_r\ (X\hat{\ }^{R\ j}))\ (Suc\ n)\ m)$
⟨proof⟩

lemma (**in** *PolynRg*) *polyn-addTr*:
$(pol\text{-}coeff\ S\ (n,\ f)) \wedge (pol\text{-}coeff\ S\ (n,\ g)) \longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \pm (polyn\text{-}expr\ R\ X\ n\ (n,\ g)) =$
$nsum\ R\ (\lambda j.\ ((f\ j) \pm_S (g\ j)) \cdot_r (X\hat{\ }^{R\ j}))\ n$
⟨proof⟩

lemma (**in** *PolynRg*) *polyn-add-n*:$[\![pol\text{-}coeff\ S\ (n,\ f);\ pol\text{-}coeff\ S\ (n,\ g)]\!] \Longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \pm (polyn\text{-}expr\ R\ X\ n\ (n,\ g)) =$
$nsum\ R\ (\lambda j.\ ((f\ j) \pm_S (g\ j)) \cdot_r (X\hat{\ }^{R\ j}))\ n$
⟨proof⟩

definition
$add\text{-}cf :: [('a,\ 'm)\ Ring\text{-}scheme,\ nat \times (nat \Rightarrow 'a),\ nat \times (nat \Rightarrow 'a)] \Rightarrow$
$nat \times (nat \Rightarrow 'a)$ **where**
$add\text{-}cf\ S\ c\ d =$
$(if\ (fst\ c) < (fst\ d)\ then\ ((fst\ d),\ \ \lambda j.\ (if\ j \le (fst\ c)$
$then\ (((snd\ c)\ j) \pm_S ((snd\ d)\ j))\ else\ ((snd$
$d)\ j)))$
$else\ if\ (fst\ c) = (fst\ d)\ then\ ((fst\ c),\ \lambda j.\ ((snd\ c)\ j \pm_S (snd\ d)\ j))$
$else\ ((fst\ c),\ \lambda j.\ (if\ j \le (fst\ d)\ then$
$((snd\ c)\ j \pm_S (snd\ d)\ j)\ else\ ((snd\ c)\ j))))$

lemma (**in** *PolynRg*) *add-cf-pol-coeff*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d]\!]$
$\Longrightarrow\ pol\text{-}coeff\ S\ (add\text{-}cf\ S\ c\ d)$
⟨proof⟩

lemma (**in** *PolynRg*) *add-cf-len*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d]\!]$
$\Longrightarrow fst\ (add\text{-}cf\ S\ c\ d) = (max\ (fst\ c)\ (fst\ d))$
⟨proof⟩

lemma (**in** *PolynRg*) *polyn-expr-restrict1*:$[\![pol\text{-}coeff\ S\ (n,\ f);$
$pol\text{-}coeff\ S\ (Suc\ (m + n),\ g)]\!] \Longrightarrow$
$polyn\text{-}expr\ R\ X\ (m + n)\ (add\text{-}cf\ S\ (n,\ f)\ (m + n,\ g)) =$
$polyn\text{-}expr\ R\ X\ (m + n)\ (m + n,\ snd\ (add\text{-}cf\ S\ (n,\ f)\ (Suc\ (m + n),\ g)))$
⟨proof⟩

lemma (**in** *PolynRg*) *polyn-add-n1*:$[\![pol\text{-}coeff\ S\ (n,\ f);\ pol\text{-}coeff\ S\ (n,\ g)]\!] \Longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \pm (polyn\text{-}expr\ R\ X\ n\ (n,\ g)) =$
$polyn\text{-}expr\ R\ X\ n\ (add\text{-}cf\ S\ (n,\ f)\ (n,\ g))$
⟨proof⟩

lemma (**in** *PolynRg*) *add-cf-val-hi*:$(fst\ c) < (fst\ d) \Longrightarrow$
$snd\ (add\text{-}cf\ S\ c\ d)\ (fst\ d) = (snd\ d)\ (fst\ d)$
⟨proof⟩

lemma (**in** *PolynRg*) *add-cf-commute*:⟦*pol-coeff S c*; *pol-coeff S d*⟧
$\Longrightarrow \forall j \leq (max\ (fst\ c)\ (fst\ d)).\ snd\ (add\text{-}cf\ S\ c\ d)\ j =$
$snd\ (add\text{-}cf\ S\ d\ c)\ j$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-addTr1*:*pol-coeff S* $(n, f) \Longrightarrow$
$\forall g.\ pol\text{-}coeff\ S\ (n + m, g) \longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n, f) \pm (polyn\text{-}expr\ R\ X\ (n + m)\ ((n + m), g))$
$= polyn\text{-}expr\ R\ X\ (n + m)\ (add\text{-}cf\ S\ (n, f)\ ((n + m), g)))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-add*:⟦*pol-coeff S* (n, f); *pol-coeff S* (m, g)⟧
$\Longrightarrow polyn\text{-}expr\ R\ X\ n\ (n, f) \pm (polyn\text{-}expr\ R\ X\ m\ (m, g))$
$= polyn\text{-}expr\ R\ X\ (max\ n\ m)\ (add\text{-}cf\ S\ (n, f)\ (m, g))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-add1*:⟦*pol-coeff S c*; *pol-coeff S d*⟧
$\Longrightarrow polyn\text{-}expr\ R\ X\ (fst\ c)\ c \pm (polyn\text{-}expr\ R\ X\ (fst\ d)\ d)$
$= polyn\text{-}expr\ R\ X\ (max\ (fst\ c)\ (fst\ d))\ (add\text{-}cf\ S\ c\ d)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-minus-nsum*:⟦*pol-coeff S c*; $k \leq (fst\ c)$⟧ $\Longrightarrow$
$-_a\ (polyn\text{-}expr\ R\ X\ k\ c) = nsum\ R\ (\lambda j.\ ((-_{a\,S}\ ((snd\ c)\ j)) \cdot_r (X^{\wedge R\ j})))\ k$
⟨*proof*⟩

lemma (**in** *PolynRg*) *minus-pol-coeff*:*pol-coeff S c* $\Longrightarrow$
$pol\text{-}coeff\ S\ ((fst\ c), (\lambda j.\ (-_{a\,S}\ ((snd\ c)\ j))))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-minus*:⟦*pol-coeff S c*; $k \leq (fst\ c)$⟧ $\Longrightarrow$
$-_a\ (polyn\text{-}expr\ R\ X\ k\ c) =$
$polyn\text{-}expr\ R\ X\ k\ (fst\ c, (\lambda j.\ (-_{a\,S}\ ((snd\ c)\ j))))$
⟨*proof*⟩

definition
m-cf :: $[('a, 'm)\ Ring\text{-}scheme, nat \times (nat \Rightarrow 'a)] \Rightarrow nat \times (nat \Rightarrow 'a)$ **where**
m-cf S c $= (fst\ c, (\lambda j.\ (-_{a\,S}\ ((snd\ c)\ j))))$

lemma (**in** *PolynRg*) *m-cf-pol-coeff*:*pol-coeff S c* $\Longrightarrow$
pol-coeff S (*m-cf S c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *m-cf-len*:*pol-coeff S c* $\Longrightarrow$
fst (*m-cf S c*) = *fst c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-minus-m-cf*:⟦*pol-coeff S c*; $k \leq (fst\ c)$⟧ $\Longrightarrow$
$-_a\ (polyn\text{-}expr\ R\ X\ k\ c) =$
$polyn\text{-}expr\ R\ X\ k\ (m\text{-}cf\ S\ c)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-zero-minus-zero*:⟦*pol-coeff S c*; $k \leq$ (*fst c*)⟧ $\Longrightarrow$
(*polyn-expr R X k c* = **0**) = (*polyn-expr R X k* (*m-cf S c*) = **0**)
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-0-pol-0*:⟦*pol-coeff S c*; $k \leq$ *fst c*⟧ $\Longrightarrow$
($\forall j \leq k$. (*snd c*) *j* = $\mathbf{0}_S$) = (*polyn-expr R X k c* = **0**)
⟨*proof*⟩

4.13.4 Multiplication of *pol-exprs*

4.13.5 Multiplication

definition
ext-cf :: [('*a*, '*m*) *Ring-scheme*, *nat*, *nat* × (*nat* ⇒ '*a*)] ⇒
nat × (*nat* ⇒ '*a*) **where**
ext-cf S n c = (*n* + *fst c*, λ*i*. *if n* ≤ *i then* (*snd c*) (*sliden n i*) *else* $\mathbf{0}_S$)

definition
sp-cf :: [('*a*, '*m*) *Ring-scheme*, '*a*, *nat* × (*nat* ⇒ '*a*)] ⇒ *nat* × (*nat* ⇒ '*a*) **where**
sp-cf S a c = (*fst c*, λ*j*. *a* $\cdot_{r S}$ ((*snd c*) *j*))

definition
special-cf :: ('*a*, '*m*) *Ring-scheme* ⇒ *nat* × (*nat* ⇒ '*a*) (C_0) **where**
C_0 *S* = (*0*, λ*j*. $1_{r S}$)

lemma (**in** *PolynRg*) *special-cf-pol-coeff*:*pol-coeff S* (C_0 *S*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *special-cf-len*:*fst* (C_0 *S*) = *0*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-pol-coeff*:*pol-coeff S c* $\Longrightarrow$
pol-coeff S (*ext-cf S n c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-len*:*pol-coeff S c* $\Longrightarrow$
fst (*ext-cf S m c*) = *m* + *fst c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-special-cf-len*:*fst* (*ext-cf S m* (C_0 *S*)) = *m*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-self*:*pol-coeff S c* $\Longrightarrow$
∀*j* ≤ (*fst c*). *snd* (*ext-cf S 0 c*) *j* = (*snd c*) *j*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-hi*:*pol-coeff S c* $\Longrightarrow$
(*snd c*) (*fst c*) =
snd (*ext-cf S n c*) (*n* + (*fst c*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-special-cf-hi*:*snd* (*ext-cf S n* (C_0 *S*)) *n* = $1_{r S}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-lo-zero*:⟦*pol-coeff S c*; *0* < *n*; *x* ≤ (*n* − *Suc 0*)⟧
$\Longrightarrow$ *snd* (*ext-cf S n c*) *x* = $\mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-special-cf-lo-zero*:⟦*0* < *n*; *x* ≤ (*n* − *Suc 0*)⟧
$\Longrightarrow$ *snd* (*ext-cf S n* (C_0 *S*)) *x* = $\mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *sp-cf-pol-coeff*:⟦*pol-coeff S c*; *a* ∈ *carrier S*⟧ $\Longrightarrow$
pol-coeff S (*sp-cf S a c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *sp-cf-len*:⟦*pol-coeff S c*; *a* ∈ *carrier S*⟧ $\Longrightarrow$
fst (*sp-cf S a c*) = *fst c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *sp-cf-val*:⟦*pol-coeff S c*; *j* ≤ (*fst c*); *a* ∈ *carrier S*⟧ $\Longrightarrow$
snd (*sp-cf S a c*) *j* = *a* $\cdot_{r S}$ ((*snd c*) *j*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ext-cf-lo-zero*:⟦*pol-coeff S c*; *0* < *j*⟧ $\Longrightarrow$
polyn-expr R X (*j* − *Suc 0*) (*ext-cf S j c*) = $\mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *monomial-d*:*pol-coeff S c* $\Longrightarrow$
polyn-expr R X d (*ext-cf S d c*) = ((*snd c*) *0*) $\cdot_r$ $X^{\wedge R\ d}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *X-to-d*: $X^{\wedge R\ d}$ = *polyn-expr R X d* (*ext-cf S d* (C_0 *S*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *c-max-ext-special-cf*:*c-max S* (*ext-cf S n* (C_0 *S*)) = *n*
⟨*proof*⟩

lemma (**in** *PolynRg*) *scalar-times-polynTr*:*a* ∈ *carrier S* $\Longrightarrow$
∀*f*. *pol-coeff S* (*n*, *f*) ⟶
a $\cdot_r$ (*polyn-expr R X n* (*n*, *f*)) = *polyn-expr R X n* (*sp-cf S a* (*n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *scalar-times-pol-expr*:⟦*a* ∈ *carrier S*; *pol-coeff S c*;

$n \leq fst\ c]\!] \Longrightarrow$
$a \cdot_r (polyn\text{-}expr\ R\ X\ n\ c) = polyn\text{-}expr\ R\ X\ n\ (sp\text{-}cf\ S\ a\ c)$
⟨proof⟩

lemma (**in** *PolynRg*) *sp-coeff-nonzero*: $[\![Idomain\ S;\ a \in carrier\ S;\ a \neq \mathbf{0}_S;$
$pol\text{-}coeff\ S\ c;\ (snd\ c)\ j \neq \mathbf{0}_S;\ j \leq (fst\ c)]\!] \Longrightarrow$
$snd\ (sp\text{-}cf\ S\ a\ c)\ j \neq \mathbf{0}_S$
⟨proof⟩

lemma (**in** *PolynRg*) *ext-cf-inductTl*: $pol\text{-}coeff\ S\ (Suc\ n,\ f) \Longrightarrow$
$polyn\text{-}expr\ R\ X\ (n + j)\ (ext\text{-}cf\ S\ j\ (Suc\ n,\ f)) =$
$polyn\text{-}expr\ R\ X\ (n + j)\ (ext\text{-}cf\ S\ j\ (n,\ f))$
⟨proof⟩

lemma (**in** *PolynRg*) *low-deg-terms-zeroTr*:
$pol\text{-}coeff\ S\ (n,\ f) \longrightarrow$
$polyn\text{-}expr\ R\ X\ (n + j)\ (ext\text{-}cf\ S\ j\ (n,\ f)) =$
$(X^{\wedge R\ j}) \cdot_r (polyn\text{-}expr\ R\ X\ n\ (n,\ f))$
⟨proof⟩

lemma (**in** *PolynRg*) *low-deg-terms-zero*: $pol\text{-}coeff\ S\ (n,\ f) \Longrightarrow$
$polyn\text{-}expr\ R\ X\ (n + j)\ (ext\text{-}cf\ S\ j\ (n,\ f)) =$
$(X^{\wedge R\ j}) \cdot_r (polyn\text{-}expr\ R\ X\ n\ (n,\ f))$
⟨proof⟩

lemma (**in** *PolynRg*) *low-deg-terms-zero1*: $pol\text{-}coeff\ S\ c \Longrightarrow$
$polyn\text{-}expr\ R\ X\ ((fst\ c) + j)\ (ext\text{-}cf\ S\ j\ c) =$
$(X^{\wedge R\ j}) \cdot_r (polyn\text{-}expr\ R\ X\ (fst\ c)\ c)$
⟨proof⟩

lemma (**in** *PolynRg*) *polyn-expr-tOpTr*: $pol\text{-}coeff\ S\ (n,\ f) \Longrightarrow$
$\forall g.\ (pol\text{-}coeff\ S\ (m,\ g) \longrightarrow (\exists h.\ pol\text{-}coeff\ S\ ((n + m),\ h) \wedge$
$h\ (n + m) = (f\ n) \cdot_{r S} (g\ m) \wedge$
$(polyn\text{-}expr\ R\ X\ (n + m)\ (n + m,\ h) =$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \cdot_r (polyn\text{-}expr\ R\ X\ m\ (m,\ g)))))$
⟨proof⟩

lemma (**in** *PolynRg*) *polyn-expr-tOp*: $[\![$
$pol\text{-}coeff\ S\ (n,\ f);\ pol\text{-}coeff\ S\ (m,\ g)]\!] \Longrightarrow \exists e.\ pol\text{-}coeff\ S\ ((n + m),\ e) \wedge$
$e\ (n + m) = (f\ n) \cdot_{r S} (g\ m) \wedge$
$polyn\text{-}expr\ R\ X\ (n + m)(n + m,\ e) =$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \cdot_r (polyn\text{-}expr\ R\ X\ m\ (m,\ g))$
⟨proof⟩

lemma (**in** *PolynRg*) *polyn-expr-tOp-c*: $[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d]\!] \Longrightarrow$
$\exists e.\ pol\text{-}coeff\ S\ e \wedge (fst\ e = fst\ c + fst\ d) \wedge$
$(snd\ e)\ (fst\ e) = (snd\ c\ (fst\ c)) \cdot_{r S} (snd\ d)\ (fst\ d) \wedge$

$polyn\text{-}expr\ R\ X\ (fst\ e)\ e =$
$(polyn\text{-}expr\ R\ X\ (fst\ c)\ c) \cdot_r (polyn\text{-}expr\ R\ X\ (fst\ d)\ d)$
⟨*proof*⟩

4.14 The degree of a polynomial

lemma (**in** *PolynRg*) *polyn-degreeTr*:$[\![pol\text{-}coeff\ S\ c;\ k \le (fst\ c)]\!] \Longrightarrow$
$(polyn\text{-}expr\ R\ X\ k\ c = \mathbf{0}\ ) = (\{j.\ j \le k \wedge (snd\ c)\ j \ne \mathbf{0}_S\} = \{\})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *higher-part-zero*:$[\![pol\text{-}coeff\ S\ c;\ k < fst\ c;$
$\forall j{\in}nset\ (Suc\ k)\ (fst\ c).\ snd\ c\ j = \mathbf{0}_S]\!] \Longrightarrow$
$\Sigma_f\ R\ (\lambda j.\ snd\ c\ j \cdot_r X\hat{}^{R\ j})\ (Suc\ k)\ (fst\ c) = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-nonzero-polyn-nonzero*:$[\![pol\text{-}coeff\ S\ c;\ k \le (fst\ c)]\!]$
$\Longrightarrow (polyn\text{-}expr\ R\ X\ k\ c \ne \mathbf{0}) = (\exists j{\le}k.\ (snd\ c)\ j \ne \mathbf{0}_S\ )$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-unique*:$[\![p \in carrier\ R;\ p \ne \mathbf{0};$
$pol\text{-}coeff\ S\ c;\ p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c;\ (snd\ c)\ (fst\ c) \ne \mathbf{0}_S;$
$pol\text{-}coeff\ S\ d;\ p = polyn\text{-}expr\ R\ X\ (fst\ d)\ d;\ (snd\ d)\ (fst\ d) \ne \mathbf{0}_S]\!] \Longrightarrow$
$(fst\ c) = (fst\ d) \wedge (\forall j \le (fst\ c).\ (snd\ c)\ j = (snd\ d)\ j)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-unique2*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d;$
$fst\ c = fst\ d]\!] \Longrightarrow$
$(polyn\text{-}expr\ R\ X\ (fst\ c)\ c = polyn\text{-}expr\ R\ X\ (fst\ d)\ d\ ) =$
$(\forall j \le (fst\ c).\ (snd\ c)\ j = (snd\ d)\ j)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-unique3*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d;$
$fst\ c < fst\ d]\!] \Longrightarrow$
$(polyn\text{-}expr\ R\ X\ (fst\ c)\ c = polyn\text{-}expr\ R\ X\ (fst\ d)\ d\ ) =$
$((\forall j \le (fst\ c).\ (snd\ c)\ j = (snd\ d)\ j) \wedge$
$(\forall j{\in}nset\ (Suc\ (fst\ c))\ (fst\ d).\ (snd\ d)\ j = \mathbf{0}_S))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-degree-unique*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d;$
$polyn\text{-}expr\ R\ X\ (fst\ c)\ c = polyn\text{-}expr\ R\ X\ (fst\ d)\ d]\!] \Longrightarrow$
$c\text{-}max\ S\ c = c\text{-}max\ S\ d$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ex-polyn-expr*:$p \in carrier\ R \Longrightarrow$
$\exists c.\ pol\text{-}coeff\ S\ c \wedge p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$
⟨*proof*⟩

lemma (**in** *PolynRg*) *c-max-eqTr0*:$[\![pol\text{-}coeff\ S\ c;\ k \le (fst\ c);$

$polyn\text{-}expr\ R\ X\ k\ c = polyn\text{-}expr\ R\ X\ (fst\ c)\ c;\ \exists j \le k.\ (snd\ c)\ j \neq \mathbf{0}_S]\!] \Longrightarrow$
$c\text{-}max\ S\ (k,\ snd\ c) = c\text{-}max\ S\ c$
⟨proof⟩

definition
$cf\text{-}sol :: [('a,\ 'b)\ Ring\text{-}scheme,\ ('a,\ 'b1)\ Ring\text{-}scheme,\ 'a,\ 'a,$
$nat \times (nat \Rightarrow 'a)] \Rightarrow bool$ **where**
$cf\text{-}sol\ R\ S\ X\ p\ c \longleftrightarrow pol\text{-}coeff\ S\ c \wedge (p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c)$

definition
$deg\text{-}n ::[('a,\ 'b)\ Ring\text{-}scheme,\ ('a,\ 'b1)\ Ring\text{-}scheme,\ 'a,\ 'a] \Rightarrow nat$ **where**
$deg\text{-}n\ R\ S\ X\ p = c\text{-}max\ S\ (SOME\ c.\ cf\text{-}sol\ R\ S\ X\ p\ c)$

definition
$deg ::[('a,\ 'b)\ Ring\text{-}scheme,\ ('a,\ 'b1)\ Ring\text{-}scheme,\ 'a,\ 'a] \Rightarrow ant$ **where**
$deg\ R\ S\ X\ p = (if\ p = \mathbf{0}_R\ then\ -\infty\ else\ (an\ (deg\text{-}n\ R\ S\ X\ p)))$

lemma (**in** *PolynRg*) $ex\text{-}cf\text{-}sol{:}p \in carrier\ R \Longrightarrow$
$\exists\, c.\ cf\text{-}sol\ R\ S\ X\ p\ c$
⟨proof⟩

lemma (**in** *PolynRg*) $deg\text{-}in\text{-}aug\text{-}minf{:}p \in carrier\ R \Longrightarrow$
$deg\ R\ S\ X\ p \in Z_{-\infty}$
⟨proof⟩

lemma (**in** *PolynRg*) $deg\text{-}noninf{:}p \in carrier\ R \Longrightarrow$
$deg\ R\ S\ X\ p \neq \infty$
⟨proof⟩

lemma (**in** *PolynRg*) $deg\text{-}ant\text{-}int{:}[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!]$
$\Longrightarrow deg\ R\ S\ X\ p = ant\ (int\ (deg\text{-}n\ R\ S\ X\ p))$
⟨proof⟩

lemma (**in** *PolynRg*) $deg\text{-}an{:}[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!]$
$\Longrightarrow deg\ R\ S\ X\ p = an\ (deg\text{-}n\ R\ S\ X\ p)$
⟨proof⟩

lemma (**in** *PolynRg*) $pol\text{-}SOME\text{-}1{:}p \in carrier\ R \Longrightarrow$
$cf\text{-}sol\ R\ S\ X\ p\ (SOME\ f.\ cf\text{-}sol\ R\ S\ X\ p\ f)$
⟨proof⟩

lemma (**in** *PolynRg*) $pol\text{-}SOME\text{-}2{:}p \in carrier\ R \Longrightarrow$
$pol\text{-}coeff\ S\ (SOME\ c.\ cf\text{-}sol\ R\ S\ X\ p\ c) \wedge$
$p = polyn\text{-}expr\ R\ X\ (fst\ (SOME\ c.\ cf\text{-}sol\ R\ S\ X\ p\ c))$
$(SOME\ c.\ cf\text{-}sol\ R\ S\ X\ p\ c)$
⟨proof⟩

lemma (**in** *PolynRg*) $coeff\text{-}max\text{-}zeroTr{:}pol\text{-}coeff\ S\ c \Longrightarrow$
$\forall j.\ j \le (fst\ c) \wedge (c\text{-}max\ S\ c) < j \longrightarrow (snd\ c)\ j = \mathbf{0}_S$

⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-max-nonzeroTr*:⟦*pol-coeff S c*;
$\exists j \le (fst\ c).\ (snd\ c)\ j \neq \mathbf{0}_S$⟧ $\Longrightarrow (snd\ c)\ (c\text{-}max\ S\ c) \neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-max-bddTr*:*pol-coeff S c* $\Longrightarrow$ *c-max S c* $\le$ (*fst c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-max*:*pol-coeff S c* $\Longrightarrow$
pol-coeff S ((*c-max S c*), *snd c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-c-max*:*pol-coeff S c* $\Longrightarrow$
polyn-expr R X (*fst c*) *c* = *polyn-expr R X* (*c-max S c*) *c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-eq-c-max*:⟦$p \in$ *carrier R*;
pol-coeff S c; *p* = *polyn-expr R X* (*fst c*) *c*⟧ $\Longrightarrow$
deg-n R S X p = *c-max S c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-le-n*:⟦$p \in$ *carrier R*; *pol-coeff S c*;
p = *polyn-expr R X* (*fst c*) *c*⟧ $\Longrightarrow$ *deg-n R S X p* $\le$ (*fst c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-le-n1*:⟦$p \in$ *carrier R*; *pol-coeff S c*; $k \le$ (*fst c*);
p = *polyn-expr R X k c*⟧ $\Longrightarrow$ *deg-n R S X p* $\le k$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-len-gt-deg*:⟦$p \in$ *carrier R*; *pol-coeff S c*;
p = *polyn-expr R X* (*fst c*) *c*; *deg R S X p* < (*an j*); $j \le$ (*fst c*)⟧
$\Longrightarrow$ (*snd c*) $j = \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-diff-deg-less*:⟦$p \in$ *carrier R*; *pol-coeff S c*;
p = *polyn-expr R X* (*fst c*) *c*; *pol-coeff S d*;
fst c = *fst d*; (*snd c*) (*fst c*) = (*snd d*) (*fst d*)⟧ $\Longrightarrow$
$p \pm (-_a$ (*polyn-expr R X* (*fst d*) *d*)) = $\mathbf{0} \vee$
deg-n R S X ($p \pm (-_a$ (*polyn-expr R X* (*fst d*) *d*))) < (*fst c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-pre-lt-deg*:⟦$p \in$ *carrier R*; *pol-coeff S c*;
deg-n R S X p $\le$ (*fst c*); (*deg-n R S X p*) $\neq$ *0*;
p = *polyn-expr R X* (*deg-n R S X p*) *c* ⟧ $\Longrightarrow$
(*deg-n R S X* (*polyn-expr R X* ((*deg-n R S X p*) − *Suc 0*) *c*)) < (*deg-n R S X p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-n*:⟦$p \in$ *carrier R*; *pol-coeff S c*;

n ≤ *fst c*; *p* = *polyn-expr R X n c*; (*snd c*) *n* ≠ $\mathbf{0}_S$⟧ ⟹
deg-n R S X p = *n*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-deg*:⟦*p* ∈ *carrier R*; *p* ≠ **0**⟧
⟹ ∃ *c*. *pol-coeff S c* ∧ *deg-n R S X p* ≤ (*fst c*) ∧
p = *polyn-expr R X* (*deg-n R S X p*) *c* ∧
(*snd c*) (*deg-n R S X p*) ≠ $\mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-n-pos*:*p* ∈ *carrier R* ⟹ *0* ≤ *deg-n R S X p*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-deg1*:⟦*p* ∈ *carrier R*; *d* = *na* (*deg R S X p*)⟧ ⟹
∃ *c*. (*pol-coeff S c* ∧ *p* = *polyn-expr R X d c*)
⟨*proof*⟩

end

theory *Algebra6* **imports** *Algebra5* **begin**

definition
s-cf :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m1*) *Ring-scheme*, *'a*, *'a*]
⇒ *nat* × (*nat* ⇒ *'a*) **where**
s-cf R S X p = (*if p* = $\mathbf{0}_R$ *then* (*0*, λ*j*. $\mathbf{0}_S$) *else*
SOME c. (*pol-coeff S c* ∧ *p* = *polyn-expr R X* (*fst c*) *c* ∧
(*snd c*) (*fst c*) ≠ $\mathbf{0}_S$))

definition
lcf :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m1*) *Ring-scheme*, *'a*, *'a*] ⇒ *'a* **where**
lcf R S X p = (*snd* (*s-cf R S X p*)) (*fst* (*s-cf R S X p*))

lemma (**in** *PolynRg*) *lcf-val-0*:*lcf R S X* **0** = $\mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *lcf-val*:⟦*p* ∈ *carrier R*; *p* ≠ **0** ⟧ ⟹
lcf R S X p = (*snd* (*s-cf R S X p*)) (*fst* (*s-cf R S X p*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-pol-coeff*:*p* ∈ *carrier R* ⟹
pol-coeff S (*s-cf R S X p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *lcf-mem*:*p* ∈ *carrier R* ⟹ (*lcf R S X p*) ∈ *carrier S*
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-expr0*:$p \in carrier\ R \Longrightarrow$
pol-coeff S (*s-cf R S X p*) $\wedge$
p = *polyn-expr R X* (*fst* (*s-cf R S X p*)) (*s-cf R S X p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pos-deg-nonzero*:⟦$p \in carrier\ R$; $0 <$ *deg-n R S X p*⟧ $\Longrightarrow$
$p \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-expr*:⟦$p \in carrier\ R$; $p \neq \mathbf{0}$⟧ $\Longrightarrow$
pol-coeff S (*s-cf R S X p*) $\wedge$
p = *polyn-expr R X* (*fst* (*s-cf R S X p*)) (*s-cf R S X p*) $\wedge$
(*snd* (*s-cf R S X p*)) (*fst* (*s-cf R S X p*)) $\neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *lcf-nonzero*:⟦$p \in carrier\ R$; $p \neq \mathbf{0}$ ⟧ $\Longrightarrow$
lcf R S X p $\neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-deg*:⟦$p \in carrier\ R$; $p \neq \mathbf{0}$⟧ $\Longrightarrow$
deg-n R S X p = *fst* (*s-cf R S X p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-edeg*:⟦$p \in carrier\ R$; *deg R S X p* $\leq$ (*an d*)⟧ $\Longrightarrow$
$\exists f$. (*pol-coeff S f* $\wedge$ *fst f* = d $\wedge$ p = *polyn-expr R X d f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-scf*:⟦*pol-coeff S c*; $k \leq$ *fst c*; *polyn-expr R X k c* $\neq \mathbf{0}$⟧
$\Longrightarrow$ $\forall j \leq$ *fst* (*s-cf R S X* (*polyn-expr R X k c*)).
snd (*s-cf R S X* (*polyn-expr R X k c*)) *j* = *snd c j*
⟨*proof*⟩

definition
scf-cond :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, '*a*,
nat, *nat* $\times$ (*nat* $\Rightarrow$ '*a*)] $\Rightarrow$ *bool* **where**
scf-cond R S X p d c $\longleftrightarrow$ *pol-coeff S c* $\wedge$ *fst c* = d $\wedge$ p = *polyn-expr R X d c*

definition
scf-d :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, '*a*, *nat*]
$\Rightarrow$ *nat* $\times$ (*nat* $\Rightarrow$ '*a*) **where**
scf-d R S X p d = (*SOME f*. *scf-cond R S X p d f*)

lemma (**in** *PolynRg*) *scf-d-polTr*:⟦$p \in carrier\ R$; *deg R S X p* $\leq$ *an d*⟧ $\Longrightarrow$
scf-cond R S X p d (*scf-d R S X p d*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *scf-d-pol*:⟦$p \in carrier\ R$; *deg R S X p* $\leq$ *an d*⟧ $\Longrightarrow$

$pol\text{-}coeff\ S\ (scf\text{-}d\ R\ S\ X\ p\ d) \wedge fst\ (scf\text{-}d\ R\ S\ X\ p\ d) = d \wedge$
$p = polyn\text{-}expr\ R\ X\ d\ (scf\text{-}d\ R\ S\ X\ p\ d)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-of-X*:
$X = polyn\text{-}expr\ R\ X\ (Suc\ 0)\ (ext\text{-}cf\ S\ (Suc\ 0)\ (C_0\ S))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-n-of-X*:$deg\text{-}n\ R\ S\ X\ X = Suc\ 0$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-X*:$cf\text{-}sol\ R\ S\ X\ X\ c \Longrightarrow$
$snd\ c\ 0 = \mathbf{0}_S \wedge snd\ c\ (Suc\ 0) = 1_{r\,S}$

⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-of-deg0*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!]$
$\Longrightarrow (deg\text{-}n\ R\ S\ X\ p = 0) = (p \in carrier\ S)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pols-const*:$[\![p \in carrier\ R;\ (deg\ R\ S\ X\ p) \leq 0]\!] \Longrightarrow$
$p \in carrier\ S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *less-deg-add-nonzero*:$[\![p \in carrier\ R;\ p \neq \mathbf{0};$
$q \in carrier\ R;\ q \neq \mathbf{0};$
$(deg\text{-}n\ R\ S\ X\ p) < (deg\text{-}n\ R\ S\ X\ q)]\!] \Longrightarrow p \pm q \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add1*:$[\![p \in carrier\ R;\ p \neq \mathbf{0};\ q \in carrier\ R;$
$q \neq \mathbf{0};\ (deg\text{-}n\ R\ S\ X\ p) < (deg\text{-}n\ R\ S\ X\ q)]\!] \Longrightarrow$
$deg\text{-}n\ R\ S\ X\ (p \pm q) = (deg\text{-}n\ R\ S\ X\ q)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add2*:$[\![p \in carrier\ R;\ p \neq \mathbf{0};\ q \in carrier\ R;$
$q \neq \mathbf{0};\ p \pm q \neq \mathbf{0};\ (deg\text{-}n\ R\ S\ X\ p) = (deg\text{-}n\ R\ S\ X\ q)]\!] \Longrightarrow$
$deg\text{-}n\ R\ S\ X\ (p \pm q) \leq (deg\text{-}n\ R\ S\ X\ q)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add3*:$[\![p \in carrier\ R;\ p \neq \mathbf{0};\ q \in carrier\ R;$
$q \neq \mathbf{0};\ p \pm q \neq \mathbf{0};\ (deg\text{-}n\ R\ S\ X\ p) \leq n;\ (deg\text{-}n\ R\ S\ X\ q) \leq n]\!] \Longrightarrow$
$deg\text{-}n\ R\ S\ X\ (p \pm q) \leq n$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add4*:$[\![p \in carrier\ R;\ q \in carrier\ R;$
$(deg\ R\ S\ X\ p) \leq (an\ n);\ (deg\ R\ S\ X\ q) \leq (an\ n)]\!] \Longrightarrow$
$deg\ R\ S\ X\ (p \pm q) \leq (an\ n)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add5*:⟦$p \in$ *carrier R*; $q \in$ *carrier R*;
(*deg R S X p*) $\leq a$; (*deg R S X q*) $\leq a$⟧ $\Longrightarrow$
deg R S X $(p \pm q) \leq a$
⟨*proof*⟩

lemma (**in** *PolynRg*) *lower-deg-part*:⟦$p \in$ *carrier R*; $p \neq \mathbf{0}$; $0 <$ *deg-n R S X p*⟧
$\Longrightarrow$
deg R S X (*polyn-expr R X* (*deg-n R S X p* $-$ *Suc 0*)(*SOME f. cf-sol R S X p f*))
$<$ *deg R S X p*
⟨*proof*⟩

definition
ldeg-p :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, *nat*, '*a*]
$\Rightarrow$ '*a* **where**
ldeg-p R S X d p = *polyn-expr R X d* (*scf-d R S X p* (*Suc d*))

definition
hdeg-p :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, *nat*, '*a*]
$\Rightarrow$ '*a* **where**
hdeg-p R S X d p = (*snd* (*scf-d R S X p d*) *d*) $\cdot_{rR}$ $(X^{\hat{}R\ d})$

lemma (**in** *PolynRg*) *ldeg-p-mem*:⟦$p \in$ *carrier R*; *deg R S X p* $\leq$ *an* (*Suc d*) ⟧ $\Longrightarrow$
ldeg-p R S X d p $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ldeg-p-zero*:$p = \mathbf{0}_R \Longrightarrow$ *ldeg-p R S X d p* $= \mathbf{0}_R$
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-mem*:⟦$p \in$ *carrier R*; *deg R S X p* $\leq$ *an* (*Suc d*)⟧ $\Longrightarrow$
hdeg-p R S X (*Suc d*) $p \in$ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-zero*:$p = \mathbf{0} \Longrightarrow$ *hdeg-p R S X* (*Suc d*) $p = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *decompos-p*:⟦$p \in$ *carrier R*; *deg R S X p* $\leq$ *an* (*Suc d*)⟧ $\Longrightarrow$
p = (*ldeg-p R S X d p*) $\pm$ (*hdeg-p R S X* (*Suc d*) *p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-ldeg-p*:⟦$p \in$ *carrier R*; *deg R S X p* $\leq$ *an* (*Suc d*)⟧ $\Longrightarrow$
deg R S X (*ldeg-p R S X d p*) $\leq$ *an d*
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-minus-eq*:⟦$p \in$ *carrier R*; $p \neq \mathbf{0}$⟧ $\Longrightarrow$
deg-n R S X ($-_a$ *p*) = *deg-n R S X p*
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-minus-eq1*:$p \in$ *carrier R* $\Longrightarrow$
deg R S X ($-_a$ *p*) = *deg R S X p*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ldeg-p-pOp*:⟦$p \in$ *carrier R*; $q \in$ *carrier R*;
deg R S X p $\leq$ *an* (*Suc d*); *deg R S X q* $\leq$ *an* (*Suc d*)⟧ $\Longrightarrow$
(*ldeg-p R S X d p*) $\pm_R$ (*ldeg-p R S X d q*) =
ldeg-p R S X d ($p \pm_R q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-pOp*:⟦$p \in$ *carrier R*; $q \in$ *carrier R*;
deg R S X p $\leq$ *an* (*Suc d*); *deg R S X q* $\leq$ *an* (*Suc d*)⟧ $\Longrightarrow$
(*hdeg-p R S X* (*Suc d*) *p*) $\pm$ (*hdeg-p R S X* (*Suc d*) *q*) =
hdeg-p R S X (*Suc d*) ($p \pm q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *ldeg-p-mOp*:⟦$p \in$ *carrier R*; *deg R S X p* $\leq$ *an* (*Suc d*)⟧ $\Longrightarrow$

$-_a$ (*ldeg-p R S X d p*) = *ldeg-p R S X d* ($-_a$ *p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-mOp*:⟦$p \in$ *carrier R*;*deg R S X p* $\leq$ *an* (*Suc d*)⟧
$\Longrightarrow$ $-_a$ (*hdeg-p R S X* (*Suc d*) *p*) = *hdeg-p R S X* (*Suc d*) ($-_a$ *p*)
⟨*proof*⟩

4.14.1 Multiplication of polynomials

lemma (**in** *PolynRg*) *deg-mult-pols*:⟦*Idomain S*;
$p \in$ *carrier R*; $p \neq \mathbf{0}$; $q \in$ *carrier R*; $q \neq \mathbf{0}$ ⟧ $\Longrightarrow$
$p \cdot_r q \neq \mathbf{0} \wedge$
deg-n R S X ($p \cdot_r q$) = *deg-n R S X p* + *deg-n R S X q*
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-mult-pols1*:⟦*Idomain S*; $p \in$ *carrier R*; $q \in$ *carrier R*⟧
$\Longrightarrow$
deg R S X ($p \cdot_r q$) = *deg R S X p* + *deg R S X q*
⟨*proof*⟩

lemma (**in** *PolynRg*) *const-times-polyn*:⟦*Idomain S*; $c \in$ *carrier S*; $c \neq \mathbf{0}_S$;
$p \in$ *carrier R*; $p \neq \mathbf{0}$⟧ $\Longrightarrow$ ($c \cdot_r p$) $\neq \mathbf{0}$ $\wedge$

$deg\text{-}n\ R\ S\ X\ (c \cdot_r p) = deg\text{-}n\ R\ S\ X\ p$
⟨*proof*⟩

lemma (**in** *PolynRg*) *p-times-monomial-nonzero*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!] \Longrightarrow (X^{\hat{}R\ j}) \cdot_r p \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *p-times-monomial-nonzero1*:$[\![Idomain\ S;\ p \in carrier\ R;\ p \neq \mathbf{0};\ c \in carrier\ S;\ c \neq \mathbf{0}_S]\!] \Longrightarrow (c \cdot_r (X^{\hat{}R\ j})) \cdot_r p \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-integral*:$Idomain\ S = Idomain\ R$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-to-X-d*:$Idomain\ S \Longrightarrow deg\text{-}n\ R\ S\ X\ (X^{\hat{}R\ d}) = d$
⟨*proof*⟩

4.14.2 Degree with value in *aug-minf*

lemma (**in** *PolynRg*) *nonzero-deg-pos*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!] \Longrightarrow 0 \leq deg\ R\ S\ X\ p$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-minf-pol-0*:$p \in carrier\ R \Longrightarrow (deg\ R\ S\ X\ p = -\infty) = (p = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-nonzero*:$p \in carrier\ R \Longrightarrow (0 \leq deg\ R\ S\ X\ p) = (p \neq \mathbf{0})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *minus-deg-in-aug-minf*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!] \Longrightarrow -\ (deg\ R\ S\ X\ p) \in Z_{-\infty}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-of-X*:$deg\ R\ S\ X\ X = 1$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-0*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!] \Longrightarrow (deg\ R\ S\ X\ p = 0) = (p \in carrier\ S)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-of-X2n*:$Idomain\ S \Longrightarrow deg\ R\ S\ X\ (X^{\hat{}R\ n}) = an\ n$
⟨*proof*⟩

lemma (**in** *PolynRg*) *add-pols-nonzero*:$[\![p \in carrier\ R;\ q \in carrier\ R;\ (deg\ R\ S\ X\ p) \neq (deg\ R\ S\ X\ q)]\!] \Longrightarrow p \pm q \neq \mathbf{0}$

⟨proof⟩

lemma (**in** *PolynRg*) *deg-pols-add1*:⟦$p \in$ *carrier R*; $q \in$ *carrier R*;
(*deg R S X p*) < (*deg R S X q*)⟧ ⟹
deg R S X ($p \pm q$) = *deg R S X q*
⟨proof⟩

lemma (**in** *PolynRg*) *deg-pols-add2*:⟦$p \in$ *carrier R*; $q \in$ *carrier R*;
(*deg R S X p*) = (*deg R S X q*)⟧ ⟹
deg R S X ($p \pm q$) ≤ (*deg R S X q*)
⟨proof⟩

lemma (**in** *PolynRg*) *deg-pols-add3*:⟦$p \in$ *carrier R*; $q \in$ *carrier R*;
(*deg R S X p*) ≤ *an n*; (*deg R S X q*) ≤ *an n*⟧ ⟹
deg R S X ($p \pm q$) ≤ *an n*
⟨proof⟩

lemma (**in** *PolynRg*) *const-times-polyn1*:⟦*Idomain S*; $p\in$ *carrier R*; $c \in$ *carrier S*;
$c \neq \mathbf{0}_S$⟧ ⟹ *deg R S X* ($c \cdot_r p$) = *deg R S X p*
⟨proof⟩

4.15 Homomorphism of polynomial rings

definition
cf-h :: ($'a \Rightarrow 'b$) ⇒ *nat* × (*nat* ⇒ $'a$) ⇒ *nat* × (*nat* ⇒ $'b$) **where**
cf-h f = (λ*c*. (*fst c*, *cmp f* (*snd c*)))

definition
polyn-Hom :: [($'a$, $'m$) *Ring-scheme*, ($'a$, $'m1$) *Ring-scheme*, $'a$,
($'b$, $'n$) *Ring-scheme*, ($'b$, $'n1$) *Ring-scheme*, $'b$] ⇒
($'a \Rightarrow 'b$) *set*
((*pHom* - - -, - - -) [67,67,67,67,67,68]67) **where**
pHom R S X, *A B Y* = {*f*. *f* ∈ *rHom R A* ∧ *f*‘(*carrier S*) ⊆ *carrier B* ∧
f X = *Y*}

definition
erh :: [($'a$, $'m$) *Ring-scheme*, ($'a$, $'m1$) *Ring-scheme*, $'a$,
($'b$, $'n$) *Ring-scheme*, ($'b$, $'n1$) *Ring-scheme*, $'b$, $'a \Rightarrow 'b$,
nat, *nat* × (*nat* ⇒ $'a$)] ⇒ $'b$ **where**
erh R S X A B Y f n c = *polyn-expr A Y n* (*cf-h f c*)

lemma (**in** *PolynRg*) *cf-h-len*:⟦*PolynRg A B Y*; *f* ∈ *rHom S B*;
pol-coeff S c⟧ ⟹ *fst* (*cf-h f c*) = *fst c*
⟨proof⟩

lemma (**in** *PolynRg*) *cf-h-coeff*:⟦*PolynRg A B Y*; *f* ∈ *rHom S B*;
pol-coeff S c⟧ ⟹ *pol-coeff B* (*cf-h f c*)
⟨proof⟩

lemma (**in** *PolynRg*) *cf-h-cmp*:⟦*PolynRg A B Y*; *pol-coeff S* (n, f); $h \in$ *rHom S B*;
$j \leq n$⟧ $\Longrightarrow$
(*snd* (*cf-h h* (n, f))) j = (*cmp h f*) j
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-special-cf*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*⟧ $\Longrightarrow$
polyn-expr A Y (*Suc 0*) (*cf-h h* (*ext-cf S* (*Suc 0*) (C_0 *S*))) =
polyn-expr A Y (*Suc 0*) (*ext-cf B* (*Suc 0*) (C_0 *B*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-Hom-coeff-to-coeff*:
⟦*PolynRg A B Y*; $f \in$ *pHom R S X*, *A B Y*; *pol-coeff S c*⟧
$\Longrightarrow$ *pol-coeff B* (*cf-h f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-len1*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*;
$f \in$ *pHom R S X*, *A B Y*; $\forall x \in$ *carrier S*. $f\ x = h\ x$; *pol-coeff S c*⟧ $\Longrightarrow$
fst (*cf-h f c*) = *fst* (*cf-h h c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-len2*:⟦*PolynRg A B Y*; $f \in$ *pHom R S X*, *A B Y*;
pol-coeff S c⟧ $\Longrightarrow$ *fst* (*cf-h f c*) = *fst c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *cmp-pol-coeff*:⟦$f \in$ *rHom S B*;
pol-coeff S (n, c)⟧ $\Longrightarrow$ *pol-coeff B* $(n,$(*cmp f c*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *cmp-pol-coeff-e*:⟦*PolynRg A B Y*; $f \in$ *pHom R S X*, *A B Y*;

pol-coeff S (n, c)⟧ $\Longrightarrow$ *pol-coeff B* $(n,$ (*cmp f c*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-pol-coeff*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*;
pol-coeff S (n, f)⟧ $\Longrightarrow$ *cf-h h* $(n, f) = (n,$ *cmp h f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-polyn*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*;
pol-coeff S (n, f)⟧ $\Longrightarrow$
polyn-expr A Y n (*cf-h h* (n, f)) = *polyn-expr A Y n* $(n,$ (*cmp h f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-rHom*:⟦*PolynRg A B Y*; $f \in$ *pHom R S X*, *A B Y*⟧ $\Longrightarrow$
$f \in$ *rHom R A*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-X-Y*:⟦*PolynRg A B Y*; *f* ∈ *pHom R S X*, *A B Y*⟧ ⟹

f X = *Y*

⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-memTr*:⟦*PolynRg A B Y*;
f ∈ *pHom R S X*, *A B Y*⟧ ⟹
∀ *c*. *pol-coeff S* (*n*, *c*) ⟶
f (*polyn-expr R X n* (*n*, *c*)) = *polyn-expr A Y n* (*n*, *cmp f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-mem*:⟦*PolynRg A B Y*;
f ∈ *pHom R S X*, *A B Y*; *pol-coeff S* (*n*, *c*)⟧ ⟹
f (*polyn-expr R X n* (*n*, *c*)) = *polyn-expr A Y n* (*n*, *cmp f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-memc*:⟦*PolynRg A B Y*; *f* ∈ *pHom R S X*, *A B Y*;
pol-coeff S c⟧ ⟹
f (*polyn-expr R X* (*fst c*) *c*) = *polyn-expr A Y* (*fst c*) (*cf-h f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-mem1*:⟦*PolynRg A B Y*; *f* ∈ *pHom R S X*, *A B Y*;
p ∈ *carrier R*⟧ ⟹ *f p* ∈ *carrier A*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-pol-mem*:⟦*PolynRg A B Y*; *f* ∈ *pHom R S X*, *A B Y*;
p ∈ *carrier R*; *p* ≠ **0**⟧ ⟹
f p = *polyn-expr A Y* (*deg-n R S X p*) (*cf-h f* (*s-cf R S X p*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *erh-rHom-coeff*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
pol-coeff S c⟧ ⟹ *erh R S X A B Y h 0 c* = (*cmp h* (*snd c*)) *0*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erh-polyn-exprs*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
pol-coeff S c; *pol-coeff S d*;
polyn-expr R X (*fst c*) *c* = *polyn-expr R X* (*fst d*) *d*⟧ ⟹
erh R S X A B Y h (*fst c*) *c* = *erh R S X A B Y h* (*fst d*) *d*
⟨*proof*⟩

definition
erH :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m1*) *Ring-scheme*, *'a*,
(*'b*, *'n*) *Ring-scheme*, (*'b*, *'n1*) *Ring-scheme*, *'b*, *'a* ⇒ *'b*] ⇒
'a ⇒ *'b* **where**
erH R S X A B Y h = (λ*x*∈*carrier R*. *erh R S X A B Y h*
(*fst* (*s-cf R S X x*)) (*s-cf R S X x*))

lemma (**in** *PolynRg*) *erH-rHom-0*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*⟧ ⟹
(*erH R S X A B Y h*) **0** = $\mathbf{0}_A$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-mem*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*; *p* ∈ *carrier R*⟧
⟹
erH R S X A B Y h p ∈ *carrier A*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-nonzero*:⟦*PolynRg A B Y*; *f* ∈ *rHom S B*;
p ∈ *carrier R*; (*erH R S X A B Y f*) *p* ≠ $\mathbf{0}_A$⟧ ⟹ *p* ≠ **0**
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHomTr2*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*⟧ ⟹
(*erH R S X A B Y h*) (1_r) = (1_{rA})
⟨*proof*⟩

declare *max.absorb1* [*simp*] *max.absorb2* [*simp*]

lemma (**in** *PolynRg*) *erH-multTr*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
pol-coeff S c⟧ ⟹
∀ *f g*. *pol-coeff S* (*m*, *f*) ∧ *pol-coeff S* (((*fst c*) + *m*), *g*) ∧
(*polyn-expr R X* (*fst c*) *c*) $\cdot_r$ (*polyn-expr R X m* (*m*, *f*)) =
(*polyn-expr R X* ((*fst c*) + *m*) ((*fst c*) + *m*, *g*)) ⟶
(*polyn-expr A Y* (*fst c*) (*cf-h h c*)) $\cdot_{rA}$ (*polyn-expr A Y m* (*cf-h h* (*m*, *f*))) =
(*polyn-expr A Y* ((*fst c*) + *m*) (*cf-h h* ((*fst c*)+*m*, *g*)))
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-multTr1*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
pol-coeff S c; *pol-coeff S d*; *pol-coeff S e*; *fst e* = *fst c* + *fst d*;
(*polyn-expr R X* (*fst c*) *c*) $\cdot_r$ (*polyn-expr R X* (*fst d*) *d*) =
polyn-expr R X ((*fst c*) + (*fst d*)) *e* ⟧ ⟹
(*polyn-expr A Y* (*fst c*) (*cf-h h c*)) $\cdot_{rA}$ (*polyn-expr A Y* (*fst d*) (*cf-h h d*))
= (*polyn-expr A Y* (*fst e*) (*cf-h h e*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *erHomTr0*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*; *x* ∈ *carrier R*⟧
⟹ *erH R S X A B Y h* ($-_a$ *x*) = $-_{aA}$ (*erH R S X A B Y h x*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erHomTr1*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
a ∈ *carrier R*; *b* ∈ *carrier R*; *a* ≠ **0**; *b* ≠ **0**; *a* ± *b* ≠ **0**;
deg-n R S X a = *deg-n R S X b*⟧ ⟹
erH R S X A B Y h (*a* ± *b*) = *erH R S X A B Y h a* $\pm_A$
(*erH R S X A B Y h b*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erHomTr2*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
a ∈ *carrier R*; *b* ∈ *carrier R*; *a* ≠ **0**; *b* ≠ **0**; *a* ± *b* ≠ **0**;
deg-n R S X a < *deg-n R S X b*⟧ ⟹
erH R S X A B Y h (*a* ± *b*) = *erH R S X A B Y h a* $\pm_A$
(*erH R S X A B Y h b*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom*:⟦*Idomain S*; *PolynRg A B Y*; *h* ∈ *rHom S B*⟧
⟹ *erH R S X A B Y h* ∈ *pHom R S X*, *A B Y*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-q-rHom*:⟦*Idomain S*; *maximal-ideal S P*;
PolynRg R′ (*S* $/_r$ *P*) *Y*⟧ ⟹
erH R S X R′ (*S* $/_r$ *P*) *Y* (*pj S P*) ∈ *pHom R S X*, *R′* (*S* $/_r$ *P*) *Y*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-add*:⟦*Idomain S*; *PolynRg A B Y*; *h* ∈ *rHom S B*;
p ∈ *carrier R*; *q* ∈ *carrier R*⟧ ⟹
erH R S X A B Y h (*p* ± *q*) =
(*erH R S X A B Y h p*) $\pm_A$ (*erH R S X A B Y h q*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-minus*:⟦*Idomain S*; *PolynRg A B Y*;
h ∈ *rHom S B*; *p* ∈ *carrier R*⟧ ⟹
erH R S X A B Y h ($-_a$ *p*) = $-_{a A}$ (*erH R S X A B Y h p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-mult*:⟦*Idomain S*; *PolynRg A B Y*; *h* ∈ *rHom S B*;
p ∈ *carrier R*; *q* ∈ *carrier R*⟧ ⟹
erH R S X A B Y h (*p* $\cdot_r$ *q*) =
(*erH R S X A B Y h p*) $\cdot_{r A}$ (*erH R S X A B Y h q*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-cf*:⟦*Idomain S*; *PolynRg A B Y*; *h* ∈ *rHom S B*;

s ∈ *carrier S*⟧ ⟹ *erH R S X A B Y h s* = *h s*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-coeff*:⟦*Idomain S*; *PolynRg A B Y*; *h* ∈ *rHom S B*;
pol-coeff S (*n*, *f*)⟧ ⟹ *pol-coeff B* (*n*, *cmp h f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-unique*:⟦*Idomain S*; *PolynRg A B Y*; *h* ∈ *rHom S B*⟧
⟹ ∃!*g*. *g* ∈ *pHom R S X*, *A B Y* ∧ (∀ *x*∈*carrier S*. *h x* = *g x*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-unique1*:⟦*Idomain S*; *PolynRg A B Y*; *h* ∈ *rHom*

S B;
  $f \in pHom\ R\ S\ X,\ A\ B\ Y; \forall x \in carrier\ S.\ f\ x = h\ x]\!] \Longrightarrow$
  $f = (erH\ R\ S\ X\ A\ B\ Y\ h)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-dec-deg*:$[\![PolynRg\ A\ B\ Y; f \in pHom\ R\ S\ X,\ A\ B\ Y;$
  $p \in carrier\ R]\!] \Longrightarrow$
    $deg\ A\ B\ Y\ (f\ p) \leq deg\ R\ S\ X\ p$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-map*:$[\![Idomain\ S;\ PolynRg\ A\ B\ Y;\ h \in rHom\ S\ B;$
  $pol\text{-}coeff\ S\ (n,\ c)]\!] \Longrightarrow$
  $(erH\ R\ S\ X\ A\ B\ Y\ h)\ (polyn\text{-}expr\ R\ X\ n\ (n,\ c)) =$
    $polyn\text{-}expr\ A\ Y\ n\ (n,\ (cmp\ h\ c))$
⟨*proof*⟩

4.16 Relatively prime polynomials

definition
  rel-prime-pols :: $[('a,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'm1)\ Ring\text{-}scheme,\ 'a,$
    $'a,\ 'a\ ] \Rightarrow bool$ **where**
  $rel\text{-}prime\text{-}pols\ R\ S\ X\ p\ q \longleftrightarrow (1_{r\,R}) \in ((Rxa\ R\ p) \mp_R (Rxa\ R\ q))$

definition
  div-condn :: $[('a,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'm1)\ Ring\text{-}scheme,\ 'a,\ nat,$
    $'a,\ 'a\ ] \Rightarrow bool$ **where**
  $div\text{-}condn\ R\ S\ X\ n\ g\ f \longleftrightarrow f \in carrier\ R \wedge n = deg\text{-}n\ R\ S\ X\ f \longrightarrow$
  $(\exists q.\ \ q \in carrier\ R \wedge ((f \pm_R (-_{a\,R}\ (q \cdot_{r\,R}\ g)) = \mathbf{0}_R) \vee (deg\text{-}n\ R\ S\ X$
  $(f \pm_R (-_{a\,R}\ (q \cdot_{r\,R}\ g))) < deg\text{-}n\ R\ S\ X\ g)))$

lemma (**in** *PolynRg*) *divisionTr0*:$[\![Idomain\ S;\ p \in carrier\ R;$
  $c \in carrier\ S;\ c \neq \mathbf{0}_S]\!] \Longrightarrow$
    $lcf\ R\ S\ X\ (c \cdot_r X^{\hat{}R\ n} \cdot_r p) = c \cdot_{r\,S} (lcf\ R\ S\ X\ p)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr1*:$[\![Corps\ S;\ g \in carrier\ R;\ g \neq \mathbf{0};$
  $0 < deg\text{-}n\ R\ S\ X\ g;\ f \in carrier\ R;\ f \neq \mathbf{0};\ deg\text{-}n\ R\ S\ X\ g \leq deg\text{-}n\ R\ S\ X\ f]\!]$
  $\Longrightarrow$
  $f \pm -_a ((lcf\ R\ S\ X\ f) \cdot_{r\,S} ((lcf\ R\ S\ X\ g)^{-\ S}) \cdot_r$
    $(X^{\hat{}R\ ((deg\text{-}n\ R\ S\ X\ f) - (deg\text{-}n\ R\ S\ X\ g))}) \cdot_r g) = \mathbf{0} \vee$
  $deg\text{-}n\ R\ S\ X\ (f \pm -_a ((lcf\ R\ S\ X\ f) \cdot_{r\,S} ((lcf\ R\ S\ X\ g)^{-\ S}) \cdot_r$
    $(X^{\hat{}R\ ((deg\text{-}n\ R\ S\ X\ f) - (deg\text{-}n\ R\ S\ X\ g))}) \cdot_r g)) < deg\text{-}n\ R\ S\ X\ f$
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr2*:$[\![Corps\ S;\ g \in carrier\ R;\ g \neq \mathbf{0};$
    $0 < deg\text{-}n\ R\ S\ X\ g]\!] \Longrightarrow \forall f.\ div\text{-}condn\ R\ S\ X\ n\ g\ f$
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr3*: $[\![$ *Corps S*; $g \in$ *carrier R*; $g \neq \mathbf{0}$;
$0 <$ *deg-n R S X g*; $f \in$ *carrier R* $]\!] \Longrightarrow$
$\exists\, q \in$ *carrier R*. $(f \pm -_a (q \cdot_r g) = \mathbf{0}) \vee (f \pm -_a (q \cdot_r g) \neq \mathbf{0} \wedge$
deg-n R S X $(f \pm -_a (q \cdot_r g)) <$ (*deg-n R S X g*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr4*: $[\![$ *Corps S*; $g \in$ *carrier R*; $g \neq \mathbf{0}$;
$0 <$ *deg-n R S X g*; $f \in$ *carrier R* $]\!] \Longrightarrow$
$\exists\, q \in$ *carrier R*. $(f = q \cdot_r g) \vee (\exists\, r \in$ *carrier R*. $r \neq \mathbf{0} \wedge (f = (q \cdot_r g) \pm r)$
$\wedge$ (*deg-n R S X r*) $<$ (*deg-n R S X g*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr*: $[\![$ *Corps S*; $g \in$ *carrier R*; $0 <$ *deg R S X g*;
$f \in$ *carrier R* $]\!] \Longrightarrow$
$\exists\, q \in$ *carrier R*. $(\exists\, r \in$ *carrier R*. $(f = (q \cdot_r g) \pm r) \wedge$
(*deg R S X r*) $<$ (*deg R S X g*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *rel-prime-equation*: $[\![$ *Corps S*; $f \in$ *carrier R*; $g \in$ *carrier R*;
$0 <$ *deg R S X f*; $0 <$ *deg R S X g*; *rel-prime-pols R S X f g*;
$h \in$ *carrier R* $]\!] \Longrightarrow$
$\exists\, u \in$ *carrier R*. $\exists\, v \in$ *carrier R*.
(*deg R S X u* $\leq$ *amax* ((*deg R S X h*) $-$ (*deg R S X f*)) (*deg R S X g*)) $\wedge$
(*deg R S X v* $\leq$ (*deg R S X f*)) $\wedge$ $(u \cdot_r f \pm (v \cdot_r g) = h)$
⟨*proof*⟩

4.16.1 Polynomial, coeff mod P

definition
P-mod :: [('a, 'm) *Ring-scheme*, ('a, 'm1) *Ring-scheme*, 'a, 'a *set*,
'a] $\Rightarrow$ *bool* **where**
P-mod R S X P p $\longleftrightarrow p = \mathbf{0}_R \vee$
($\forall j \leq$ (*fst* (*s-cf R S X p*)). (*snd* (*s-cf R S X p*) *j*) $\in P$)

lemma (**in** *PolynRg*) *P-mod-whole*: $p \in$ *carrier R* $\Longrightarrow$
P-mod R S X (*carrier S*) *p*
⟨*proof*⟩

lemma (**in** *PolynRg*) *zero-P-mod*: *ideal S I* $\Longrightarrow$ *P-mod R S X I* $\mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mod*: $[\![$ *ideal S I*; $p \in$ *carrier R*; *pol-coeff S c*;
$p =$ *polyn-expr R X* (*fst c*) *c* $]\!] \Longrightarrow$
($\forall j \leq$ (*fst c*). (*snd c*) $j \in I$) $=$ (*P-mod R S X I p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *monomial-P-mod-mod*: $[\![$ *ideal S I*; $c \in$ *carrier S*;
$p = c \cdot_r (X^{\hat{}R\ d})$ $]\!] \Longrightarrow$ $(c \in I) =$ (*P-mod R S X I p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-add*:⟦*ideal S I*; *p* ∈ *carrier R*;
q ∈ *carrier R*; *P-mod R S X I p*; *P-mod R S X I q*⟧ ⟹
P-mod R S X I (*p* ± *q*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-minus*:⟦*ideal S I*; *p* ∈ *carrier R*; *P-mod R S X I p*⟧
⟹
P-mod R S X I ($-_a$ *p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-pre*:⟦*ideal S I*; *pol-coeff S* ((*Suc n*), *f*);
P-mod R S X I (*polyn-expr R X* (*Suc n*) (*Suc n*, *f*))⟧ ⟹
P-mod R S X I (*polyn-expr R X n* (*n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-pre1*:⟦*ideal S I*; *pol-coeff S* ((*Suc n*), *f*);
P-mod R S X I (*polyn-expr R X* (*Suc n*) (*Suc n*, *f*))⟧ ⟹
P-mod R S X I (*polyn-expr R X n* (*Suc n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-coeffTr*:⟦*ideal S I*; *d* ∈ *carrier S*⟧ ⟹
(*P-mod R S X I d*) = (*d* ∈ *I*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult-const*:⟦*ideal S I*; *ideal S J*;
pol-coeff S (*n*, *f*); *P-mod R S X I* (*polyn-expr R X n* (*n*, *f*));
pol-coeff S (*0*, *g*); *P-mod R S X J* (*polyn-expr R X 0* (*0*, *g*))⟧ ⟹
P-mod R S X (*I* $\diamondsuit_{r\,S}$ *J*) ((*polyn-expr R X n* (*n*, *f*)) $\cdot_r$
(*polyn-expr R X 0* (*0*, *g*)))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult-const1*:⟦*ideal S I*; *ideal S J*;
pol-coeff S (*n*, *f*); *P-mod R S X I* (*polyn-expr R X n* (*n*, *f*));
d ∈ *J*⟧ ⟹
P-mod R S X (*I* $\diamondsuit_{r\,S}$ *J*) ((*polyn-expr R X n* (*n*, *f*)) $\cdot_r$ *d*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult-monomial*:⟦*ideal S I*; *p* ∈ *carrier R*⟧ ⟹
(*P-mod R S X I p*) = (*P-mod R S X I* (*p* $\cdot_r$ $X^{\hat{}R\ m}$))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-multTr*:⟦*ideal S I*; *ideal S J*; *pol-coeff S* (*n*, *f*);
P-mod R S X I (*polyn-expr R X n* (*n*, *f*))⟧ ⟹ ∀ *g*. ((*pol-coeff S* (*m*, *g*)
∧ (*P-mod R S X J* (*polyn-expr R X m* (*m*, *g*)))) ⟶
P-mod R S X (*I* $\diamondsuit_{r\,S}$ *J*)
((*polyn-expr R X n* (*n*, *f*)) $\cdot_r$ (*polyn-expr R X m* (*m*, *g*))))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult*:⟦*ideal S I*; *ideal S J*; *pol-coeff S* (*n*, *c*); *pol-coeff S* (*m*, *d*); *P-mod R S X I* (*polyn-expr R X n* (*n*, *c*)); *P-mod R S X J* (*polyn-expr R X m* (*m*, *d*))⟧ $\Longrightarrow$ *P-mod R S X* ($I \diamondsuit_{r\,S}$ *J*) ((*polyn-expr R X n* (*n*, *c*)) $\cdot_r$ (*polyn-expr R X m* (*m*, *d*)))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult1*:⟦*ideal S I*; *ideal S J*; $p \in$ *carrier R*; $q \in$ *carrier R*; *P-mod R S X I p*; *P-mod R S X J q*⟧ $\Longrightarrow$ *P-mod R S X* ($I \diamondsuit_{r\,S}$ *J*) ($p \cdot_r q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult2l*:⟦*ideal S I*; $p \in$ *carrier R*; $q \in$ *carrier R*; *P-mod R S X I p*⟧ $\Longrightarrow$ *P-mod R S X I* ($p \cdot_r q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult2r*:⟦*ideal S I*; $p \in$ *carrier R*; $q \in$ *carrier R*; *P-mod R S X I q*⟧ $\Longrightarrow$ *P-mod R S X I* ($p \cdot_r q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *csrp-fn-pol-coeff*:⟦*ideal S P*; *PolynRg R′* ($S /_r P$) *Y*; *pol-coeff* ($S /_r P$) (*n*, *c′*)⟧ $\Longrightarrow$ *pol-coeff S* (*n*, (*cmp* (*csrp-fn S P*) *c′*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *pj-csrp-mem-coeff*:⟦*ideal S P*; *pol-coeff* ($S /_r P$) (*n*, *c′*)⟧ $\Longrightarrow \forall j \leq n.$ (*pj S P*) ((*csrp-fn S P*) (*c′ j*)) = *c′ j*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-pj-csrp*:⟦*Idomain S*; *ideal S P*; *PolynRg R′* ($S /_r P$) *Y*; *pol-coeff* ($S /_r P$) (*n*, *c′*)⟧ $\Longrightarrow$ *erH R S X R′* ($S /_r P$) *Y* (*pj S P*) (*polyn-expr R X n* (*n*, (*cmp* (*csrp-fn S P*) *c′*))) = *polyn-expr R′ Y n* (*n*, *c′*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-csrp-fn-nonzero*:⟦*Idomain S*; *ideal S P*; *PolynRg R′* ($S /_r P$) *Y*; *g′* $\in$ *carrier R′*; *g′* $\neq \mathbf{0}_{R'}$ ⟧ $\Longrightarrow$ *polyn-expr R X* (*deg-n R′* ($S /_r P$) *Y g′*) ((*deg-n R′* ($S /_r P$) *Y g′*), (*cmp* (*csrp-fn S P*) (*snd* (*s-cf R′* ($S /_r P$) *Y g′*)))) $\neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-inv*:⟦*Idomain S*; *ideal S P*; *Ring R′*; *PolynRg R′* ($S /_r P$) *Y*; *g′* $\in$ *carrier R′*⟧ $\Longrightarrow$ $\exists$ *g*$\in$*carrier R*. *deg R S X g* $\leq$ (*deg R′* ($S /_r P$) *Y g′*) $\wedge$ (*erH R S X R′* ($S /_r P$) *Y* (*pj S P*)) *g* = *g′*
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-0*:⟦*Idomain S*; *ideal S P*; *PolynRg R′* ($S /_r P$) *Y*;

$g \in carrier\ R]\!] \Longrightarrow$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ g = \mathbf{0}_{R'}) = (P\text{-}mod\ R\ S\ X\ P\ g)$
⟨proof⟩

lemma (**in** *PolynRg*) $P\text{-}mod\text{-}I\text{-}J$:$[\![p \in carrier\ R;\ ideal\ S\ I;\ ideal\ S\ J;$
$I \subseteq J;\ \ P\text{-}mod\ R\ S\ X\ I\ p]\!] \Longrightarrow P\text{-}mod\ R\ S\ X\ J\ p$
⟨proof⟩

lemma (**in** *PolynRg*) $P\text{-}mod\text{-}n\text{-}1$:$[\![Idomain\ S;\ t \in carrier\ S;\ g \in carrier\ R;$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\hat{\ }^{S\ (Suc\ n)}))\ g]\!] \Longrightarrow P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ g$
⟨proof⟩

lemma (**in** *PolynRg*) $P\text{-}mod\text{-}n\text{-}m$:$[\![Idomain\ S;\ t \in carrier\ S;\ g \in carrier\ R;$
$m \le n;\ P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\hat{\ }^{S\ (Suc\ n)}))\ g]\!] \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\hat{\ }^{S\ (Suc\ m)}))\ g$
⟨proof⟩

lemma (**in** *PolynRg*) $P\text{-}mod\text{-}diff$:$[\![Idomain\ S;\ ideal\ S\ P;\ PolynRg\ R'\ (S\ /_r\ P)\ Y;$

$g \in carrier\ R;\ h \in carrier\ R]\!] \Longrightarrow$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ g = (erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)$
$h))$
$= (P\text{-}mod\ R\ S\ X\ P\ (g \pm -_a\ h))$
⟨proof⟩

lemma (**in** *PolynRg*) $P\text{-}mod\text{-}erH$:$[\![Idomain\ S;\ ideal\ S\ P;\ PolynRg\ R'\ (S\ /_r\ P)\ Y;$

$g \in carrier\ R;\ v \in carrier\ R;\ t \in P\ ]\!] \Longrightarrow$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ g =$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ (g \pm (t \cdot_r\ v))))$
⟨proof⟩

lemma (**in** *PolynRg*) $coeff\text{-}principalTr$:$[\![t \in carrier\ S]\!] \Longrightarrow$
$\forall f.\ pol\text{-}coeff\ S\ (n, f) \wedge (\forall j \le n.\ f\ j \in S \diamondsuit_p\ t) \longrightarrow$
$(\exists f'.\ pol\text{-}coeff\ S\ (n, f') \wedge (\forall j \le n.\ f\ j = t \cdot_{rS}\ (f'\ j)))$
⟨proof⟩

lemma (**in** *PolynRg*) $coeff\text{-}principal$:$[\![t \in carrier\ S;\ pol\text{-}coeff\ S\ (n, f);$
$\forall j \le n.\ f\ j \in S \diamondsuit_p\ t]\!] \Longrightarrow$
$\exists f'.\ pol\text{-}coeff\ S\ (n, f') \wedge (\forall j \le n.\ f\ j = t \cdot_{rS}\ (f'\ j))$
⟨proof⟩

lemma (**in** *PolynRg*) $Pmod\text{-}0\text{-}principal$:$[\![Idomain\ S;\ t \in carrier\ S;\ g \in carrier\ R;$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ g]\!] \Longrightarrow \exists h \in carrier\ R.\ g = t \cdot_r\ h$
⟨proof⟩

lemma (**in** *PolynRg*) $Pmod0\text{-}principal\text{-}rev$:$[\![Idomain\ S;\ t \in carrier\ S;$
$g \in carrier\ R;\ \exists h \in carrier\ R.\ g = t \cdot_r\ \ h]\!] \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ g$

$\langle proof \rangle$

lemma (**in** *PolynRg*) *Pmod0-principal-rev1*: $[\![Idomain\ S;\ t \in carrier\ S;$
$h \in carrier\ R]\!] \Longrightarrow P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (t \cdot_r h)$
$\langle proof \rangle$

lemma (**in** *PolynRg*) *Pmod0-principal-erH-vanish-t*: $[\![Idomain\ S;\ ideal\ S\ (S \diamondsuit_p\ t);$
$t \in carrier\ S;\ t \neq \mathbf{0}_S;\ PolynRg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ ]\!] \Longrightarrow$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ t = \mathbf{0}_{R'}$
$\langle proof \rangle$

lemma (**in** *PolynRg*) *P-mod-diffxxx1*: $[\![Idomain\ S;\ t \in carrier\ S;\ t \neq \mathbf{0}_S;$
$maximal\text{-}ideal\ S\ (S \diamondsuit_p\ t);\ PolynRg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y;$
$f \in carrier\ R;\ g \in carrier\ R;\ h \in carrier\ R;$
$f \neq \mathbf{0};\ g \neq \mathbf{0};\ h \neq \mathbf{0};\ u \in carrier\ R;\ v \in carrier\ R;$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g \neq \mathbf{0}_{R'};$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ h \neq \mathbf{0}_{R'};$
$ra \in carrier\ R;$
$f \pm -_a (g \cdot_r h) = t\ \hat{}^{S\ m} \cdot_r ra;\ 0 < m;$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ u)$
$\cdot_{r\,R'}\ erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g \pm_{R'}$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ v)$
$\cdot_{r\,R'}\ erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ h =$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ ra]\!]$
$\Longrightarrow P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\ \hat{}^{S\ (Suc\ m)}))$
$(f \pm -_a ((g \pm t\ \hat{}^{S\ m} \cdot_r v) \cdot_r (h \pm t\ \hat{}^{S\ m} \cdot_r u)))$
$\langle proof \rangle$

lemma (**in** *PolynRg*) *P-mod-diffxxx2*: $[\![Idomain\ S;\ t \in carrier\ S;\ t \neq \mathbf{0}_S;$
$maximal\text{-}ideal\ S\ (S \diamondsuit_p\ t);\ PolynRg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y;$
$f \in carrier\ R;\ g \in carrier\ R;\ h \in carrier\ R;$
$deg\ R\ S\ X\ g \leq deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g);$
$deg\ R\ S\ X\ h +$
$deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g)$
$\leq deg\ R\ S\ X\ f;$
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g);$
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ h);$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g)$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ h);$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\ \hat{}^{S\ m}))\ (f \pm -_a (g \cdot_r h));\ 0 < m]\!] \Longrightarrow$
$\exists g1\ h1.\ g1 \in carrier\ R \wedge h1 \in carrier\ R \wedge$
$(deg\ R\ S\ X\ g1 \leq deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g1)) \wedge$

$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\,\hat{}^S\ m))\ (g \pm -_a\ g1) \wedge\ (deg\ R\ S\ X\ h1\ +$
$deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ g1)$
$\le deg\ R\ S\ X\ f) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\,\hat{}^S\ m))\ (h \pm -_a\ h1) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\,\hat{}^S\ (Suc\ m)))\ (f \pm (-_a\ (g1\ \cdot_r\ h1)))$
$\langle proof \rangle$

definition
$Hensel\text{-}next :: [('a,\ 'b)\ Ring\text{-}scheme,\ ('a,\ 'c)\ Ring\text{-}scheme,\ 'a,\ 'a,$
$('a\ set,\ 'm)\ Ring\text{-}scheme,\ 'a\ set, 'a,\ nat] \Rightarrow ('a \times 'a) \Rightarrow ('a \times 'a)$
$((9Hen\ \_\ \_\ \_\ \_\ \_\ \_\ \_\ \_\ \text{-}\ \text{-})\ [67,67,67,67,67,67,67,68]67)$ **where**

$Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh = (SOME\ gh1.$
$gh1 \in carrier\ R \times carrier\ R \wedge$
$(deg\ R\ S\ X\ (fst\ gh1) \le deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh1))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\,\hat{}^S\ m))\ ((fst\ gh) \pm_R -_{aR}\ (fst\ gh1)) \wedge$
$(deg\ R\ S\ X\ (snd\ gh1) + deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh1)) \le deg\ R\ S\ X\ f) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\,\hat{}^S\ m))\ ((snd\ gh) \pm_R -_{aR}\ (snd\ gh1)) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\,\hat{}^S\ (Suc\ m)))\ (f \pm_R (-_{aR}\ ((fst\ gh1)\ \cdot_{rR}\ (snd\ gh1)))))$

lemma $cart\text{-}prod\text{-}fst{:}x \in A \times B \Longrightarrow fst\ x \in A$
$\langle proof \rangle$

lemma $cart\text{-}prod\text{-}snd{:}x \in A \times B \Longrightarrow snd\ x \in B$
$\langle proof \rangle$

lemma $cart\text{-}prod\text{-}split{:}((x,y) \in A \times B) = (x \in A \wedge y \in B)$
$\langle proof \rangle$

lemma (**in** $PolynRg$) $P\text{-}mod\text{-}diffxxx3{:}[\![Idomain\ S;\ t \in carrier\ S;\ t \ne \mathbf{0}_S;$
$maximal\text{-}ideal\ S\ (S \diamondsuit_p\ t);\ PolynRg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y;$
$f \in carrier\ R;\ gh \in carrier\ R \times carrier\ R;$
$deg\ R\ S\ X\ (fst\ gh) \le deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh));$
$deg\ R\ S\ X\ (snd\ gh) + deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh)) \le deg\ R\ S\ X\ f;$
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh));$
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ gh));$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh))$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ gh));$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t\,\hat{}^S\ m))\ (f \pm -_a\ ((fst\ gh)\ \cdot_r\ (snd\ gh)));\ 0 < m]\!] \Longrightarrow$

∃ *gh1*. *gh1* ∈*carrier R* × *carrier R* ∧
(*deg R S X* (*fst gh1*) ≤ *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*fst gh1*))) ∧
P-mod R S X ($S\ \diamondsuit_p\ (t\ \hat{}^S\ m)$) ((*fst gh*) $\pm\ -_a$ (*fst gh1*)) ∧
(*deg R S X* (*snd gh1*) + *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*fst gh1*)) ≤
deg R S X f) ∧
P-mod R S X ($S\ \diamondsuit_p\ (t\ \hat{}^S\ m)$) ((*snd gh*) $\pm\ -_a$ (*snd gh1*)) ∧
P-mod R S X ($S\ \diamondsuit_p\ (t\ \hat{}^S\ (Suc\ m))$) (*f* $\pm$ ($-_a$ ((*fst gh1*) $\cdot_r$ (*snd gh1*))))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx4*:⟦*Idomain S*; *t* ∈ *carrier S*; $t \neq \mathbf{0}_S$;
maximal-ideal S ($S\ \diamondsuit_p\ t$); *PolynRg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*; *f* ∈ *carrier R*;
gh ∈ *carrier R* × *carrier R*;
deg R S X (*fst gh*) ≤ *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*fst gh*));
deg R S X (*snd gh*) + *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*erH R S X R′*
($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*fst gh*)) ≤ *deg R S X f*;
0 < *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*fst gh*));
0 < *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*snd gh*));
rel-prime-pols R′ ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*fst gh*))
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$)) (*snd gh*));
P-mod R S X ($S\ \diamondsuit_p\ (t\ \hat{}^S\ m)$) (*f* $\pm\ -_a$ ((*fst gh*) $\cdot_r$ (*snd gh*))); *0* < *m*⟧ ⟹
($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*) ∈ *carrier R* × *carrier R* ∧ (*deg R S X*
(*fst* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*)) ≤ *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$))
(*fst* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*)))) ∧
P-mod R S X ($S\ \diamondsuit_p\ (t\ \hat{}^S\ m)$) ((*fst gh*) $\pm\ -_a$ (*fst* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*))) ∧

(*deg R S X* (*snd* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*)) + *deg R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y*
(*erH R S X R′* ($S\ /_r\ (S\ \diamondsuit_p\ t)$) *Y* (*pj S* ($S\ \diamondsuit_p\ t$))
(*fst* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*))) ≤ *deg R S X f*) ∧
P-mod R S X ($S\ \diamondsuit_p\ (t\ \hat{}^S\ m)$) ((*snd gh*) $\pm\ -_a$ (*snd* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*)))
∧
P-mod R S X ($S\ \diamondsuit_p\ (t\ \hat{}^S\ (Suc\ m))$) (*f* $\pm$ ($-_a$ ((*fst* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*))
$\cdot_r$
(*snd* ($Hen_{R\ S\ X\ t\ R'\ Y\ f}$ *m gh*)))))
⟨*proof*⟩

primrec
Hensel-pair :: [(′*a*, ′*b*) *Ring-scheme*, (′*a*, ′*c*) *Ring-scheme*, ′*a*, ′*a*,
(′*a set*, ′*m*) *Ring-scheme*, ′*a set*, ′*a*, ′*a*, ′*a*, *nat*] ⇒ (′*a* × ′*a*)

((*10Hpr* - - - - - - - - - - -) [*67,67,67,67,67,67,67,67,67,68*]*67*)
where
Hpr-0: $Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ 0 = (g,\ h)$
| *Hpr-Suc*: $Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m) =$
$Hen_{R\ S\ X\ t\ R'\ Y\ f}\ (Suc\ m)\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)$

lemma (**in** *PolynRg*) *fst-xxx*: $[\![t \in carrier\ S;\ t \neq \mathbf{0}_S;\ ideal\ S\ (S \diamondsuit_p\ t);$
$\forall (n::nat).\ (F\ n) \in carrier\ R \times carrier\ R;$
$\forall m.\ P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (fst\ (F\ m) \pm -_a\ (fst\ (F\ (Suc\ m))))]\!] \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (fst\ (F\ 0) \pm -_a\ (fst\ (F\ n)))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *snd-xxx*: $[\![t \in carrier\ S;\ t \neq \mathbf{0}_S;$
$ideal\ S\ (S \diamondsuit_p\ t);\ \forall (n::nat).\ (F\ n) \in carrier\ R \times carrier\ R;$
$\forall m.\ P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (snd\ (F\ m) \pm -_a\ (snd\ (F\ (Suc\ m))))]\!] \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (snd\ (F\ 0) \pm -_a\ (snd\ (F\ n)))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5*: $[\![Idomain\ S;\ t \in carrier\ S;\ t \neq \mathbf{0}_S;$
$maximal\text{-}ideal\ S\ (S \diamondsuit_p\ t);\ PolynRg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y;$
$f \in carrier\ R;\ (g,\ h) \in carrier\ R \times carrier\ R;$
$deg\ R\ S\ X\ (fst\ (g,\ h)) \leq deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h)));$
$deg\ R\ S\ X\ (snd\ (g,\ h)) + deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h))) \leq deg\ R\ S\ X\ f;$
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h)));$
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S \diamondsuit_p\ t))\ (snd\ (g,\ h)));$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h)))$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ (g,\ h)));$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (f \pm -_a\ (g \cdot_r h))]\!] \Longrightarrow$
$(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)) \in carrier\ R \times carrier\ R\ \wedge$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) =$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h))\ \wedge$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) =$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ (g,\ h))\ \wedge$
$(deg\ R\ S\ X\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) \leq deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\wedge S\ (Suc\ m)}))\ ((fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \pm -_a$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \wedge$
$(deg\ R\ S\ X\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) + deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$

$(Suc\ m)))) \leq\ deg\ R\ S\ X\ f) \wedge$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ (t\,\hat{}^{S\ (Suc\ m)}))\ ((snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m))\ \pm\ -_a\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ (t\,\hat{}^{S\ (Suc\ (Suc\ m))}))\ (f\ \pm\ -_a\ ((fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))\ \cdot_r\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))))$
$\langle proof \rangle$

lemma (**in** *PolynRg*) *P-mod-diffxxx5-1*:$[\![Idomain\ S;\ t \in carrier\ S;\ t \neq \mathbf{0}_S;$
$maximal\text{-}ideal\ S\ (S\ \diamondsuit_p\ t);\ PolynRg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y;$
$f \in carrier\ R;\ g \in carrier\ R;\ h \in carrier\ R;$
$deg\ R\ S\ X\ g \leq deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$deg\ R\ S\ X\ h + deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g) \leq deg\ R\ S\ X\ f;$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g)$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ t)\ (f\ \pm\ -_a\ (g\ \cdot_r\ h))]\!] \Longrightarrow$
$(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)) \in carrier\ R \times carrier\ R\ \wedge$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) =$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ (fst\ (g,\ h)) \wedge$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))$
$(snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) =$
$erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ (snd\ (g,\ h)) \wedge$
$(deg\ R\ S\ X\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) \leq deg\ R'$
$(S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S\ \diamondsuit_p\ t))\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ (t\,\hat{}^{S\ (Suc\ m)}))\ ((fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m))\ \pm\ -_a$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \wedge$
$(deg\ R\ S\ X\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) +$
$deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S\ \diamondsuit_p\ t))\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \leq\ deg\ R\ S\ X\ f) \wedge$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ (t\,\hat{}^{S\ (Suc\ m)}))\ ((snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m))\ \pm\ -_a$
$(snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ (t\,\hat{}^{S\ (Suc\ (Suc\ m))}))\ (f\ \pm\ -_a$
$((fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))\ \cdot_r\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))))$
$\langle proof \rangle$

lemma (**in** *PolynRg*) *P-mod-diffxxx5-2*: $[\![$*Idomain S*; $t \in$ *carrier S*; $t \neq \mathbf{0}_S$;
maximal-ideal S $(S \diamondsuit_p t)$; *PolynRg* R' $(S /_r (S \diamondsuit_p t))$ Y; $f \in$ *carrier R*;
$g \in$ *carrier R*; $h \in$ *carrier R*;
deg R S X g $\leq$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g);
deg R S X h $+$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y (*erH R S X* R'
$(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g) $\leq$ *deg R S X f*;
$0 <$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g);
$0 <$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) h);
rel-prime-pols R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g)
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) h);
P-mod R S X $(S \diamondsuit_p t)$ $(f \pm -_a (g \cdot_r h))]\!] \Longrightarrow$
$(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m) \in$ *carrier R* $\times$ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5-3*: $[\![$*Idomain S*; $t \in$ *carrier S*; $t \neq \mathbf{0}_S$;
maximal-ideal S $(S \diamondsuit_p t)$; *PolynRg* R' $(S /_r (S \diamondsuit_p t))$ Y; $f \in$ *carrier R*;
$g \in$ *carrier R*; $h \in$ *carrier R*;
deg R S X g $\leq$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g);
deg R S X h $+$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y (*erH R S X* R'
$(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g) $\leq$ *deg R S X f*;
$0 <$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g);
$0 <$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) h);
rel-prime-pols R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g)
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) h);
P-mod R S X $(S \diamondsuit_p t)$ $(f \pm -_a (g \cdot_r h))]\!] \Longrightarrow$
P-mod R S X $(S \diamondsuit_p (t\hat{}^{S}\ m))$ ((*fst* $(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)$) $\pm$
$-_a$ (*fst* $(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (m + n))$))) $\wedge$
P-mod R S X $(S \diamondsuit_p (t\hat{}^{S}\ m))$ ((*snd* $(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)$) $\pm$
$-_a$ (*snd* $(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (m + n))$)))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5-4*: $[\![$*Idomain S*; $t \in$ *carrier S*; $t \neq \mathbf{0}_S$;
maximal-ideal S $(S \diamondsuit_p t)$; *PolynRg* R' $(S /_r (S \diamondsuit_p t))$ Y; $f \in$ *carrier R*;
$g \in$ *carrier R*; $h \in$ *carrier R*;
deg R S X g $\leq$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y
(*erH R S X* R' $(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g);
deg R S X h $+$ *deg* R' $(S /_r (S \diamondsuit_p t))$ Y (*erH R S X* R'
$(S /_r (S \diamondsuit_p t))$ Y (*pj S* $(S \diamondsuit_p t)$) g) $\leq$ *deg R S X f*;

$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g)$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ t)\ (f\ \pm\ -_a\ (g\ \cdot_r\ h))] \Longrightarrow$
$deg\ R\ S\ X\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \leq deg\ R\ S\ X\ g\ \wedge$
$deg\ R\ S\ X\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \leq deg\ R\ S\ X\ f$
⟨*proof*⟩

declare *max.absorb1* [*simp del*] *max.absorb2* [*simp del*]

end

theory *Algebra7* **imports** *Algebra6* **begin**

Chapter 5

Modules

5.1 Basic properties of Modules

record ('a, 'b) *Module* = 'a *aGroup* +
sprod :: 'b ⇒ 'a ⇒ 'a (**infixl** $\cdot_{s1}$ *76*)

locale *Module* = *aGroup M* **for** *M* (**structure**) +
fixes *R* (**structure**)
assumes *sc-Ring*: *Ring R*
and *sprod-closed* :
$[\![a \in carrier\ R;\ m \in carrier\ M]\!] \Longrightarrow a \cdot_s m \in carrier\ M$
and *sprod-l-distr*:
$[\![a \in carrier\ R;\ b \in carrier\ R;\ m \in carrier\ M]\!] \Longrightarrow$
$(a \pm_R b) \cdot_s m = a \cdot_s m \pm_M b \cdot_s m$
and *sprod-r-distr*:
$[\![a \in carrier\ R;\ m \in carrier\ M;\ n \in carrier\ M]\!] \Longrightarrow$
$a \cdot_s (m \pm_M n) = a \cdot_s m \pm_M a \cdot_s n$
and *sprod-assoc*:
$[\![a \in carrier\ R;\ b \in carrier\ R;\ m \in carrier\ M]\!] \Longrightarrow$
$(a \cdot_{rR} b) \cdot_s m = a \cdot_s (b \cdot_s m)$
and *sprod-one*:
$m \in carrier\ M \Longrightarrow (1_{rR}) \cdot_s m = m$

definition
submodule :: [('b, 'm) *Ring-scheme*, ('a, 'b, 'c) *Module-scheme*, 'a *set*] ⇒ *bool* **where**
$submodule\ R\ A\ H \longleftrightarrow H \subseteq carrier\ A \wedge A +> H \wedge (\forall a.\ \forall m.$
$(a \in carrier\ R \wedge m \in H) \longrightarrow (sprod\ A\ a\ m) \in H)$

definition
mdl :: [('a, 'b, 'm) *Module-scheme*, 'a *set*] ⇒ ('a, 'b) *Module* **where**
mdl M H = (|*carrier* = *H*, *pop* = *pop M*, *mop* = *mop M*, *zero* = *zero M*,
sprod = λ*a*. λ*x*∈*H*. *sprod M a x*|)

abbreviation

MODULE (**infixl** *module 58*) **where**
R module M == Module M R

lemma (**in** *Module*) *module-is-ag*: *aGroup M* ⟨*proof*⟩

lemma (**in** *Module*) *module-inc-zero*: $\mathbf{0}_M \in$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-subset*:*submodule R M H* $\Longrightarrow$ *H* $\subseteq$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-asubg*:*submodule R M H* $\Longrightarrow$ *M +> H*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-subset1*:⟦*submodule R M H*; *h* ∈ *H*⟧ $\Longrightarrow$
h ∈ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-inc-0*:*submodule R M H* $\Longrightarrow$
$\mathbf{0}_M \in H$
⟨*proof*⟩

lemma (**in** *Module*) *sc-un*: *m* ∈ *carrier M* $\Longrightarrow$ $1_{r R} \cdot_s m = m$
⟨*proof*⟩

lemma (**in** *Module*) *sc-mem*:⟦*a* ∈ *carrier R*; *m* ∈ *carrier M*⟧ $\Longrightarrow$
$a \cdot_s m \in$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-sc-closed*:⟦*submodule R M H*;
a ∈ *carrier R*; *h* ∈ *H*⟧ $\Longrightarrow$ $a \cdot_s h \in H$
⟨*proof*⟩

lemma (**in** *Module*) *sc-assoc*:⟦*a* ∈ *carrier R*; *b* ∈ *carrier R*;
m ∈ *carrier M*⟧ $\Longrightarrow$ $(a \cdot_{r R} b) \cdot_s m = a \cdot_s (b \cdot_s m)$
⟨*proof*⟩

lemma (**in** *Module*) *sc-l-distr*:⟦*a* ∈ *carrier R*; *b* ∈ *carrier R*;
m ∈ *carrier M*⟧ $\Longrightarrow$ $(a \pm_R b) \cdot_s m = a \cdot_s m \pm b \cdot_s m$
⟨*proof*⟩

lemma (**in** *Module*) *sc-r-distr*:⟦*a* ∈ *carrier R*; *m* ∈ *carrier M*; *n* ∈ *carrier M*⟧ $\Longrightarrow$
$a \cdot_s (m \pm n) = a \cdot_s m \pm a \cdot_s n$
⟨*proof*⟩

lemma (**in** *Module*) *sc-0-m*:*m* ∈ *carrier M* $\Longrightarrow$ $\mathbf{0}_R \cdot_s m = \mathbf{0}_M$
⟨*proof*⟩

lemma (**in** *Module*) *sc-a-0*:$a \in carrier\ R \Longrightarrow a \cdot_s \mathbf{0}\ = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Module*) *sc-minus-am*:$[\![a \in carrier\ R;\ m \in carrier\ M]\!]$
$\Longrightarrow -_a\ (a \cdot_s m) = a \cdot_s (-_a\ m)$
⟨*proof*⟩

lemma (**in** *Module*) *sc-minus-am1*:$[\![a \in carrier\ R;\ m \in carrier\ M]\!]$
$\Longrightarrow -_a\ (a \cdot_s m) = (-_{a_R}\ a) \cdot_s m$
⟨*proof*⟩

lemma (**in** *Module*) *submodule-0*:*submodule R M* $\{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Module*) *submodule-whole*:*submodule R M* (*carrier M*)
⟨*proof*⟩

lemma (**in** *Module*) *submodule-pOp-closed*:$[\![submodule\ R\ M\ H;\ h \in H;\ k \in H]\!]$
$\Longrightarrow$
$h \pm k \in H$
⟨*proof*⟩

lemma (**in** *Module*) *submodule-mOp-closed*:$[\![submodule\ R\ M\ H;\ h \in H]\!]$
$\Longrightarrow -_a\ h \in H$
⟨*proof*⟩

definition
mHom :: $[('b, 'm)\ Ring\text{-}scheme, ('a, 'b, 'm1)\ Module\text{-}scheme,$
$('c, 'b, 'm2)\ Module\text{-}scheme] \Rightarrow ('a \Rightarrow 'c)\ set$
where
$mHom\ R\ M\ N = \{f.\ f \in aHom\ M\ N \wedge$
$(\forall a \in carrier\ R.\ \forall m \in carrier\ M.\ f\ (a \cdot_{s_M} m) = a \cdot_{s_N} (f\ m))\}$

definition
mimg :: $[('b, 'm)\ Ring\text{-}scheme, ('a, 'b, 'm1)\ Module\text{-}scheme,$
$('c, 'b, 'm2)\ Module\text{-}scheme, 'a \Rightarrow 'c] \Rightarrow ('c, 'b)\ Module$
((*4mimg- -,-/ -*) [*88,88,88,89*]*88*) **where**
$mimg_{R\ M,N}\ f = mdl\ N\ (f\ `\ (carrier\ M))$

definition
mzeromap :: $[('a, 'b, 'm1)\ Module\text{-}scheme, ('c, 'b, 'm2)\ Module\text{-}scheme]$
$\Rightarrow ('a \Rightarrow 'c)$ **where**
$mzeromap\ M\ N = (\lambda x \in carrier\ M.\ \mathbf{0}_N)$

lemma (**in** *Ring*) *mHom-func*:$f \in mHom\ R\ M\ N \Longrightarrow f \in carrier\ M \rightarrow carrier\ N$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-test*:$[\![R\ module\ N;\ f \in carrier\ M \rightarrow carrier\ N \wedge$
$f \in extensional\ (carrier\ M) \wedge$

$(\forall\, m{\in}carrier\ M.\ \forall\, n{\in}carrier\ M.\ f\ (m \pm_M n) = f\, m \pm_N (f\, n)) \wedge$
$(\forall\, a{\in}carrier\ R.\ \forall\, m{\in}carrier\ M.\ f\ (a \cdot_{s\,M} m) = a \cdot_{s\,N} (f\, m))]\!] \Longrightarrow$
$f \in mHom\ R\ M\ N$
⟨proof⟩

lemma (**in** *Module*) *mHom-mem*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;\ m \in carrier\ M]\!]$
$\Longrightarrow f\, m \in carrier\ N$
⟨proof⟩

lemma (**in** *Module*) *mHom-add*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;\ m \in carrier\ M;$
$n \in carrier\ M]\!] \Longrightarrow f\ (m \pm n) = f\, m \pm_N (f\, n)$
⟨proof⟩

lemma (**in** *Module*) *mHom-0*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N]\!] \Longrightarrow f\ (\mathbf{0}) = \mathbf{0}_N$
⟨proof⟩

lemma (**in** *Module*) *mHom-inv*:$[\![R\ module\ N;\ m \in carrier\ M;\ f \in mHom\ R\ M\ N]\!] \Longrightarrow$
$f\ (-_a\ m) = -_{a\,N} (f\, m)$
⟨proof⟩

lemma (**in** *Module*) *mHom-lin*:$[\![R\ module\ N;\ m \in carrier\ M;\ f \in mHom\ R\ M\ N;$
$a \in carrier\ R]\!] \Longrightarrow f\ (a \cdot_s m) = a \cdot_{s\,N} (f\, m)$
⟨proof⟩

lemma (**in** *Module*) *mker-inc-zero*:
$[\![R\ module\ N;\ f \in mHom\ R\ M\ N\ ]\!] \Longrightarrow \mathbf{0} \in (ker_{M,N}\, f)$
⟨proof⟩

lemma (**in** *Module*) *mHom-eq-ker*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;\ a \in carrier\ M;$
$b{\in}\ carrier\ M;\ a \pm (-_a\ b) \in ker_{M,N}\, f]\!] \Longrightarrow f\, a = f\, b$
⟨proof⟩

lemma (**in** *Module*) *mHom-ker-eq*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;\ a \in carrier\ M;$
$b{\in}\ carrier\ M;\ f\, a = f\, b]\!] \Longrightarrow a \pm (-_a\ b) \in ker_{M,N}\, f$
⟨proof⟩

lemma (**in** *Module*) *mker-submodule*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N]\!] \Longrightarrow$
$submodule\ R\ M\ (ker_{M,N}\, f)$
⟨proof⟩

lemma (**in** *Module*) *mker-mzeromap*:$R\ module\ N \Longrightarrow$
$ker_{M,N}\, (mzeromap\ M\ N) = carrier\ M$
⟨proof⟩

lemma (**in** *Module*) *mdl-carrier*:*submodule R M H* $\Longrightarrow$ *carrier* (*mdl M H*) = *H*
⟨*proof*⟩

lemma (**in** *Module*) *mdl-is-ag*:*submodule R M H* $\Longrightarrow$ *aGroup* (*mdl M H*)
⟨*proof*⟩

lemma (**in** *Module*) *mdl-is-module*:*submodule R M H* $\Longrightarrow$ *R module* (*mdl M H*)
⟨*proof*⟩

lemma (**in** *Module*) *submodule-of-mdl*:⟦*submodule R M H*; *submodule R M N*; *H* ⊆ *N*⟧
$\Longrightarrow$ *submodule R* (*mdl M N*) *H*
⟨*proof*⟩

lemma (**in** *Module*) *img-set-submodule*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ $\Longrightarrow$
submodule R N (*f* ‘ (*carrier M*))
⟨*proof*⟩

lemma (**in** *Module*) *mimg-module*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ $\Longrightarrow$
R module (*mimg R M N f*)
⟨*proof*⟩

lemma (**in** *Module*) *surjec-to-mimg*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ $\Longrightarrow$
$surjec_{M,\ (mimg\ R\ M\ N\ f)}\ f$
⟨*proof*⟩

definition
tOp-mHom :: [('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*] ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) **where**
tOp-mHom R M N f g = (λ*x* ∈ *carrier M*. (*f x* $\pm_N$ (*g x*)))

definition
iOp-mHom :: [('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*] ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) **where**
iOp-mHom R M N f = (λ*x* ∈ *carrier M*. ($-_{aN}$ (*f x*)))

definition
sprod-mHom ::[('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*] ⇒ '*b* ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) **where**
sprod-mHom R M N a f = (λ*x* ∈ *carrier M*. *a* $\cdot_{sN}$ (*f x*))

definition
HOM :: [('*b*, '*more*) *Ring-scheme*, ('*a*, '*b*, '*more1*) *Module-scheme*,
('*c*, '*b*, '*more2*) *Module-scheme*] ⇒ ('*a* ⇒ '*c*, '*b*) *Module*
((*3HOM*- -/ -) [*90*, *90*, *91*] *90*) **where**
HOM_R *M N* = (|*carrier* = *mHom R M N*, *pop* = *tOp-mHom R M N*,
mop = *iOp-mHom R M N*, *zero* = *mzeromap M N*, *sprod* =*sprod-mHom R M N* |)

lemma (**in** *Module*) *zero-HOM*:*R module N* $\Longrightarrow$
mzeromap M N = $\mathbf{0}_{HOM_R\ M\ N}$
⟨*proof*⟩

lemma (**in** *Module*) *tOp-mHom-closed*:⟦*R module N*; *f* ∈ *mHom R M N*; *g* ∈ *mHom R M N*⟧
$\Longrightarrow$ *tOp-mHom R M N f g* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *iOp-mHom-closed*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
$\Longrightarrow$ *iOp-mHom R M N f* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-ex-zero*:*R module N* $\Longrightarrow$ *mzeromap M N* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-eq*:⟦*R module N*; *f* ∈ *mHom R M N*; *g* ∈ *mHom R M N*;
∀ *m*∈*carrier M*. *f m* = *g m*⟧ $\Longrightarrow$ *f* = *g*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-l-zero*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
$\Longrightarrow$ *tOp-mHom R M N* (*mzeromap M N*) *f* = *f*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-l-inv*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
$\Longrightarrow$ *tOp-mHom R M N* (*iOp-mHom R M N f*) *f* = *mzeromap M N*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-tOp-assoc*:⟦*R module N*; *f* ∈ *mHom R M N*; *g* ∈ *mHom R M N*;
h ∈ *mHom R M N*⟧ $\Longrightarrow$ *tOp-mHom R M N* (*tOp-mHom R M N f g*) *h* =
tOp-mHom R M N f (*tOp-mHom R M N g h*)
⟨*proof*⟩

lemma (**in** *Module*) *mHom-tOp-commute*:⟦*R module N*; *f* ∈ *mHom R M N*;
g ∈ *mHom R M N*⟧ $\Longrightarrow$ *tOp-mHom R M N f g* = *tOp-mHom R M N g f*
⟨*proof*⟩

lemma (**in** *Module*) *HOM-is-ag*:*R module N* $\Longrightarrow$ *aGroup* (HOM_R *M N*)
⟨*proof*⟩

lemma (**in** *Module*) *sprod-mHom-closed*:⟦*R module N*; *a* ∈ *carrier R*;
f ∈ *mHom R M N*⟧ $\Longrightarrow$ *sprod-mHom R M N a f* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *HOM-is-module*:*R module N* $\Longrightarrow$ *R module* (HOM_R *M N*)
⟨*proof*⟩

5.2 Injective hom, surjective hom, bijective hom and inverse hom

definition
invmfun :: [('b, 'm) *Ring-scheme*, ('a, 'b, 'm1) *Module-scheme*,
('c, 'b, 'm2) *Module-scheme*, 'a ⇒ 'c] ⇒ 'c ⇒ 'a **where**
invmfun R M N (*f* :: 'a ⇒ 'c) =
(λy∈(*carrier N*). *SOME x*. (*x* ∈ (*carrier M*) ∧ *f x* = *y*))

definition
misomorphic :: [('b, 'm) *Ring-scheme*, ('a, 'b, 'm1) *Module-scheme*,
('c, 'b, 'm2) *Module-scheme*] ⇒ *bool* **where**
misomorphic R M N ⟷ (∃*f*. *f* ∈ *mHom R M N* ∧ $bijec_{M,N}$ *f*)

definition
mId :: ('a, 'b, 'm1) *Module-scheme* ⇒ 'a ⇒ 'a   ((*mId*-/) [89]88) **where**
mId_M = (λm∈*carrier M*. *m*)

definition
mcompose :: [('a, 'r, 'm1) *Module-scheme*, 'b ⇒ 'c, 'a ⇒ 'b] ⇒ 'a ⇒ 'c **where**
mcompose M g f = *compose* (*carrier M*) *g f*

abbreviation
MISOM ((3- ≅- -) [82,82,83]82) **where**
$M \cong_R N$ == *misomorphic R M N*

lemma (**in** *Module*) *minjec-inj*:⟦*R module N*; $injec_{M,N}$ *f*⟧ ⟹
inj-on f (*carrier M*)
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-l-inv*:⟦*R module N*; $bijec_{M,N}$ *f*; *m* ∈ *carrier M*⟧ ⟹
(*invmfun R M N f*) (*f m*) = *m*
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-mHom*:⟦*R module N*; $bijec_{M,N}$ *f*; *f* ∈ *mHom R M N* ⟧ ⟹
invmfun R M N f ∈ *mHom R N M*
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-r-inv*:⟦*R module N*; $bijec_{M,N}$ *f*; *n* ∈ *carrier N*⟧ ⟹
f ((*invmfun R M N f*) *n*) = *n*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-compos*:⟦*R module L*; *R module N*; *f* ∈ *mHom R L M*;
g ∈ *mHom R M N* ⟧ ⟹ *compos L g f* ∈ *mHom R L N*
⟨*proof*⟩

lemma (**in** *Module*) *mcompos-inj-inj*:⟦*R module L*; *R module N*; *f* ∈ *mHom R L*

M;
$g \in mHom\ R\ M\ N;\ injec_{L,M}\ f;\ injec_{M,N}\ g\ \rrbracket \Longrightarrow injec_{L,N}\ (compos\ L\ g\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *mcompos-surj-surj*:$\llbracket R\ module\ L;\ R\ module\ N;\ surjec_{L,M}\ f;$
$surjec_{M,N}\ g;\ f \in mHom\ R\ L\ M;\ g \in mHom\ R\ M\ N\ \rrbracket \Longrightarrow$
$surjec_{L,N}\ (compos\ L\ g\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *mId-mHom*:$mId_M \in mHom\ R\ M\ M$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-mId-bijec*:$\llbracket R\ module\ N;\ f \in mHom\ R\ M\ N;\ g \in mHom$
$R\ N\ M;$
$compose\ (carrier\ M)\ g\ f = mId_M;\ compose\ (carrier\ N)\ f\ g = mId_N\rrbracket \Longrightarrow$
$bijec_{M,N}\ f$
⟨*proof*⟩

definition
sup-sharp :: [('*r*, '*n*) *Ring-scheme*, ('*b*, '*r*, '*m1*) *Module-scheme*,
('*c*, '*r*, '*m2*) *Module-scheme*, ('*a*, '*r*, '*m*) *Module-scheme*, '*b* ⇒ '*c*]
⇒ ('*c* ⇒ '*a*) ⇒ ('*b* ⇒ '*a*) **where**
sup-sharp R M N L u = (λ*f*∈*mHom R N L*. *compos M f u*)

definition
sub-sharp :: [('*r*, '*n*) *Ring-scheme*, ('*a*, '*r*, '*m*) *Module-scheme*,
('*b*, '*r*, '*m1*) *Module-scheme*, ('*c*, '*r*, '*m2*) *Module-scheme*, '*b* ⇒ '*c*]
⇒ ('*a* ⇒ '*b*) ⇒ ('*a* ⇒ '*c*) **where**
sub-sharp R L M N u = (λ*f*∈*mHom R L M*. *compos L u f*)

lemma (**in** *Module*) *sup-sharp-homTr*:$\llbracket R\ module\ N;\ R\ module\ L;\ u \in mHom\ R$
$M\ N;$
$f \in mHom\ R\ N\ L\ \rrbracket \Longrightarrow sup\text{-}sharp\ R\ M\ N\ L\ u\ f \in mHom\ R\ M\ L$
⟨*proof*⟩

lemma (**in** *Module*) *sup-sharp-hom*:$\llbracket R\ module\ N;\ R\ module\ L;\ u \in mHom\ R\ M$
$N\rrbracket \Longrightarrow$
$sup\text{-}sharp\ R\ M\ N\ L\ u \in mHom\ R\ (HOM_R\ N\ L)\ (HOM_R\ M\ L)$
⟨*proof*⟩

lemma (**in** *Module*) *sub-sharp-homTr*:$\llbracket R\ module\ N;\ R\ module\ L;\ u \in mHom\ R\ M$
$N;$
$f \in mHom\ R\ L\ M\rrbracket \Longrightarrow sub\text{-}sharp\ R\ L\ M\ N\ u\ f \in mHom\ R\ L\ N$
⟨*proof*⟩

lemma (**in** *Module*) *sub-sharp-hom*:$\llbracket R\ module\ N;\ R\ module\ L;\ u \in mHom\ R\ M$
$N\rrbracket \Longrightarrow$

sub-sharp R L M N u ∈ *mHom R* (*HOM*$_R$ *L M*) (*HOM*$_R$ *L N*)
⟨*proof*⟩

lemma (**in** *Module*) *mId-bijec*:*bijec*$_{M,M}$ (*mId*$_M$)
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-bijec*:⟦*R module N*; *f* ∈ *mHom R M N*; *bijec*$_{M,N}$ *f*⟧ ⟹
bijec$_{N,M}$ (*invmfun R M N f*)
⟨*proof*⟩

lemma (**in** *Module*) *misom-self*:*M* ≅$_R$ *M*
⟨*proof*⟩

lemma (**in** *Module*) *misom-sym*:⟦*R module N*; *M* ≅$_R$ *N*⟧ ⟹ *N* ≅$_R$ *M*
⟨*proof*⟩

lemma (**in** *Module*) *misom-trans*:⟦*R module L*; *R module N*; *L* ≅$_R$ *M*; *M* ≅$_R$ *N*⟧ ⟹
L ≅$_R$ *N*
⟨*proof*⟩

definition
mr-coset :: [′*a*, (′*a*, ′*b*, ′*more*) *Module-scheme*, ′*a set*] ⇒ ′*a set* **where**
mr-coset a M H = *a* ⊎$_M$ *H*

definition
set-mr-cos :: [(′*a*, ′*b*, ′*more*) *Module-scheme*, ′*a set*] ⇒ ′*a set set* **where**
set-mr-cos M H = {*X*. ∃ *a*∈*carrier M*. *X* = *a* ⊎$_M$ *H*}

definition
mr-cos-sprod :: [(′*a*, ′*b*, ′*more*) *Module-scheme*, ′*a set*] ⇒
′*b* ⇒ ′*a set* ⇒ ′*a set* **where**
mr-cos-sprod M H a X = {*z*. ∃*x*∈*X*. ∃*h*∈*H*. *z* = *h* ±$_M$ (*a* ·$_{s M}$ *x*)}

definition
mr-cospOp :: [(′*a*, ′*b*, ′*more*) *Module-scheme*, ′*a set*] ⇒
′*a set* ⇒ ′*a set* ⇒ ′*a set* **where**
mr-cospOp M H = (λ*X*. λ*Y*. *c-top* (*b-ag M*) *H X Y*)

definition
mr-cosmOp :: [(′*a*, ′*b*, ′*more*) *Module-scheme*, ′*a set*] ⇒
′*a set* ⇒ ′*a set* **where**
mr-cosmOp M H = (λ*X*. *c-iop* (*b-ag M*) *H X*)

definition
qmodule :: [(′*a*, ′*r*, ′*more*) *Module-scheme*, ′*a set*] ⇒
(′*a set*, ′*r*) *Module* **where**
qmodule M H = (| *carrier* = *set-mr-cos M H*, *pop* = *mr-cospOp M H*,

mop = *mr-cosmOp M H*, *zero* = *H*, *sprod* = *mr-cos-sprod M H*⦈

definition
sub-mr-set-cos :: [('*a*, '*r*, '*more*) *Module-scheme*, '*a set*, '*a set*] ⇒
'*a set set* **where**
sub-mr-set-cos M H N = {*X*. ∃ *n*∈*N*. *X* = *n* $\uplus_M$ *H*}

abbreviation
QMODULE (**infixl** '/'$_m$ *200*) **where**
M $/_m$ *H* == *qmodule M H*

abbreviation
SUBMRSET ((*3*-/ $_s$'/'-/ -) [*82*,*82*,*83*]*82*) **where**
N $_s/_M$ *H* == *sub-mr-set-cos M H N*

lemma (**in** *Module*) *qmodule-carr*:*submodule R M H* ⟹
carrier (*qmodule M H*) = *set-mr-cos M H*
⟨*proof*⟩

lemma (**in** *Module*) *set-mr-cos-mem*:⟦*submodule R M H*; *m* ∈ *carrier M*⟧ ⟹
m $\uplus_M$ *H* ∈ *set-mr-cos M H*
⟨*proof*⟩

lemma (**in** *Module*) *mem-set-mr-cos*:⟦*submodule R M N*; *x* ∈ *set-mr-cos M N*⟧
⟹
∃ *m* ∈ *carrier M*. *x* = *m* $\uplus_M$ *N*
⟨*proof*⟩

lemma (**in** *Module*) *m-in-mr-coset*:⟦*submodule R M H*; *m* ∈ *carrier M*⟧ ⟹
m ∈ *m* $\uplus_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-h-stable*:⟦*submodule R M H*; *h* ∈ *H*⟧ ⟹
H = *h* $\uplus_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-h-stable1*:⟦*submodule R M H*; *m* ∈ *carrier M*; *h* ∈ *H*⟧
⟹ (*m* ± *h*) $\uplus_M$ *H* = *m* $\uplus_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *x-in-mr-coset*:⟦*submodule R M H*; *m* ∈ *carrier M*; *x* ∈ *m* $\uplus_M$
H⟧
⟹ ∃ *h*∈*H*. *m* ± *h* = *x*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprodTr*:⟦*submodule R M H*; *a* ∈ *carrier R*;
m ∈ *carrier M*⟧ ⟹ *mr-cos-sprod M H a* (*m* $\uplus_M$ *H*) = (*a* $\cdot_s$ *m*) $\uplus_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-mem*:⟦*submodule R M H*; $a \in$ *carrier R*;
$X \in$ *set-mr-cos M H*⟧ $\Longrightarrow$ *mr-cos-sprod M H a X* $\in$ *set-mr-cos M H*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-assoc*:⟦*submodule R M H*; $a \in$ *carrier R*;
$b \in$ *carrier R*; $X \in$ *set-mr-cos M H*⟧ $\Longrightarrow$ *mr-cos-sprod M H* $(a \cdot_{r R} b)$ X =
mr-cos-sprod M H a (*mr-cos-sprod M H b X*)
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-one*:⟦*submodule R M H*; $X \in$ *set-mr-cos M H*⟧
$\Longrightarrow$
mr-cos-sprod M H $(1_{r R})$ $X = X$
⟨*proof*⟩

lemma (**in** *Module*) *mr-cospOpTr*:⟦*submodule R M H*; $m \in$ *carrier M*; $n \in$ *carrier M*⟧
$\Longrightarrow$ *mr-cospOp M H* $(m \uplus_M H)$ $(n \uplus_M H) = (m \pm n) \uplus_M H$
⟨*proof*⟩

lemma(**in** *Module*) *mr-cos-sprod-distrib1*:⟦*submodule R M H*; $a \in$ *carrier R*;
$b \in$ *carrier R*; $X \in$ *set-mr-cos M H*⟧ $\Longrightarrow$
mr-cos-sprod M H $(a \pm_R b)$ X =
mr-cospOp M H (*mr-cos-sprod M H a X*) (*mr-cos-sprod M H b X*)
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-distrib2*:⟦*submodule R M H*;
$a \in$ *carrier R*; $X \in$ *set-mr-cos M H*; $Y \in$ *set-mr-cos M H*⟧ $\Longrightarrow$
mr-cos-sprod M H a (*mr-cospOp M H X Y*) =
mr-cospOp M H (*mr-cos-sprod M H a X*) (*mr-cos-sprod M H a Y*)
⟨*proof*⟩

lemma (**in** *Module*) *mr-cosmOpTr*:⟦*submodule R M H*; $m \in$ *carrier M*⟧ $\Longrightarrow$
mr-cosmOp M H $(m \uplus_M H) = (-_a m) \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-oneTr*:*submodule R M H* $\Longrightarrow H = \mathbf{0} \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-oneTr1*:⟦*submodule R M H*; $m \in$ *carrier M*⟧ $\Longrightarrow$
mr-cospOp M H H $(m \uplus_M H) = m \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *qmodule-is-ag*:*submodule R M H* $\Longrightarrow$ *aGroup* $(M /_m H)$
⟨*proof*⟩

lemma (**in** *Module*) *qmodule-module*:*submodule R M H* $\Longrightarrow$ *R module* $(M /_m H)$
⟨*proof*⟩

definition
indmhom :: [('b, 'm) *Ring-scheme*, ('a, 'b, 'm1) *Module-scheme*,
('c, 'b, 'm2) *Module-scheme*, 'a ⇒ 'c] ⇒ 'a *set* ⇒ 'c **where**
indmhom R M N f = (λX∈ (*set-mr-cos M* ($ker_{M,N}$ *f*)). *f* (*SOME x. x* ∈ *X*))

abbreviation
INDMHOM ((4-$^\flat$- -, -) [92,92,92,93]92) **where**
$f^\flat{}_{R\ M,N}$ == *indmhom R M N f*

lemma (**in** *Module*) *indmhom-someTr*:⟦*R module N*; *f* ∈ *mHom R M N*;
X ∈ *set-mr-cos M* ($ker_{M,N}$ *f*)⟧ ⟹ *f* (*SOME xa. xa* ∈ *X*) ∈ *f* '(*carrier M*)
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-someTr1*:⟦*R module N*; *f* ∈ *mHom R M N*; *m* ∈ *carrier M*⟧
⟹ *f* (*SOME xa. xa* ∈ (*ar-coset m M* ($ker_{M,N}$ *f*))) = *f m*
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-someTr2*:⟦*R module N*; *f* ∈ *mHom R M N*;
submodule R M H; *m* ∈ *carrier M*; *H* ⊆ $ker_{M,N}$ *f*⟧ ⟹
f (*SOME xa. xa* ∈ *m* $\uplus_M$ *H*) = *f m*
⟨*proof*⟩

lemma (**in** *Module*) *indmhomTr1*:⟦*R module N*; *f* ∈ *mHom R M N*; *m* ∈ *carrier M*⟧ ⟹
($f^\flat{}_{R\ M,N}$) (*m* $\uplus_M$ ($ker_{M,N}$ *f*)) = *f m*
⟨*proof*⟩

lemma (**in** *Module*) *indmhomTr2*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
⟹ ($f^\flat{}_{R\ M,N}$) ∈ *set-mr-cos M* ($ker_{M,N}$ *f*) → *carrier N*
⟨*proof*⟩

lemma (**in** *Module*) *indmhom*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
⟹ ($f^\flat{}_{R\ M,N}$) ∈ *mHom R* (*M* $/_m$ ($ker_{M,N}$ *f*)) *N*
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-injec*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ ⟹
$injec_{(M\ /_m\ (ker_{M,N}\ f)),N}$ ($f^\flat{}_{R\ M,N}$)
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-surjec1*:⟦*R module N*; $surjec_{M,N}$ *f*;
f ∈ *mHom R M N*⟧ ⟹ $surjec_{(M\ /_m\ (ker_{M,N}\ f)),N}$ ($f^\flat{}_{R\ M,N}$)
⟨*proof*⟩

lemma (**in** *Module*) *module-homTr*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ ⟹
f ∈ *mHom R M* ($mimg_{R\ M,N}$ *f*)
⟨*proof*⟩

lemma (**in** *Module*) *ker-to-mimg*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N]\!] \Longrightarrow$
$ker_{M,mimg_{R\ M,N}\ f}\ f = ker_{M,N}\ f$
⟨*proof*⟩

lemma (**in** *Module*) *module-homTr1*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N]\!] \Longrightarrow$
$(mimg_{R\ (M\ /_m\ (ker_{M,N}\ f)),N}\ (f^{\flat}{}_{R\ M,N})) = mimg_{R\ M,N}\ f$ ⟨*proof*⟩

lemma (**in** *Module*) *module-Homth-1*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N]\!] \Longrightarrow$
$M\ /_m\ (ker_{M,N}\ f) \cong_R mimg_{R\ M,N}\ f$
⟨*proof*⟩

definition
$mpj :: [('a, 'r, 'm)\ Module\text{-}scheme, 'a\ set] \Rightarrow\ ('a => 'a\ set)$ **where**
$mpj\ M\ H = (\lambda x{\in}carrier\ M.\ x \uplus_M H)$

lemma (**in** *Module*) *elem-mpj*:$[\![m \in carrier\ M;\ submodule\ R\ M\ H]\!] \Longrightarrow$
$mpj\ M\ H\ m = m \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *mpj-mHom*:$submodule\ R\ M\ H \Longrightarrow mpj\ M\ H \in mHom\ R\ M$
$(M\ /_m\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *mpj-mem*:$[\![submodule\ R\ M\ H;\ m \in carrier\ M]\!] \Longrightarrow$
$mpj\ M\ H\ m \in carrier\ (M\ /_m\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *mpj-surjec*:$submodule\ R\ M\ H \Longrightarrow$
$surjec_{M,(M\ /_m\ H)}\ (mpj\ M\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *mpj-0*:$[\![submodule\ R\ M\ H;\ h \in H]\!] \Longrightarrow$
$mpj\ M\ H\ h\ = \mathbf{0}_{(M\ /_m\ H)}$
⟨*proof*⟩

lemma (**in** *Module*) *mker-of-mpj*:$submodule\ R\ M\ H \Longrightarrow$
$ker_{M,(M\ /_m\ H)}\ (mpj\ M\ H) = H$
⟨*proof*⟩

lemma (**in** *Module*) *indmhom1*:$[\![submodule\ R\ M\ H;\ R\ module\ N;\ f \in mHom\ R\ M$
$N;\ \ H \subseteq ker_{M,N}\ f]\!] \Longrightarrow \exists!g.\ g \in (mHom\ R\ (M\ /_m\ H)\ N) \wedge (compos\ M\ g\ (mpj$
$M\ H)) = f$
⟨*proof*⟩

definition
$mQmp :: [('a, 'r, 'm)\ Module\text{-}scheme, 'a\ set, 'a\ set] \Rightarrow$
$('a\ set \Rightarrow 'a\ set)$ **where**

mQmp M H N = ($\lambda X \in$ *set-mr-cos M H*. {*z*. $\exists$ *x* $\in$ *X*. $\exists$ *y* $\in$ *N*. ($y \pm_M x$ = *z*)})

abbreviation
MQP ((*3Mp*_ _,_) [*82*,*82*,*83*]*82*) **where**
$Mp_{M\ H,N}$ == *mQmp M H N*

lemma (**in** *Module*) *mQmpTr0*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$;
$m \in$ *carrier M*⟧ $\Longrightarrow$ *mQmp M H N* ($m \uplus_M H$) = $m \uplus_M N$
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr1*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$;
$m \in$ *carrier M*; $n \in$ *carrier M*; $m \uplus_M H = n \uplus_M H$⟧ $\Longrightarrow$ $m \uplus_M N = n \uplus_M N$
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr2*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$;
$X \in$ *carrier* ($M /_m H$)⟧ $\Longrightarrow$ (*mQmp M H N*) $X \in$ *carrier* ($M /_m N$)
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr2-1*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$
⟧
$\Longrightarrow$ *mQmp M H N* $\in$ *carrier* ($M /_m H$) $\rightarrow$ *carrier* ($M /_m N$)
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr3*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$;
$X \in$ *carrier* ($M /_m H$); $Y \in$ *carrier* ($M /_m H$)⟧ $\Longrightarrow$ (*mQmp M H N*) (*mr-cospOp M H X Y*) = *mr-cospOp M N* ((*mQmp M H N*) *X*) ((*mQmp M H N*) *Y*)
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr4*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$;
$a \in N$⟧ $\Longrightarrow$ *mr-coset a* (*mdl M N*) *H* = *mr-coset a M H*
⟨*proof*⟩

lemma (**in** *Module*) *mQmp-mHom*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$⟧ $\Longrightarrow$
($Mp_{M\ H,N}$) $\in$ *mHom R* ($M /_m H$) ($M /_m N$)
⟨*proof*⟩

lemma (**in** *Module*) *Mp-surjec*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$⟧
$\Longrightarrow$
$surjec_{(M\ /_m\ H),(M\ /_m\ N)}$ ($Mp_{M\ H,N}$)
⟨*proof*⟩

lemma (**in** *Module*) *kerQmp*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$⟧
$\Longrightarrow$ $ker_{(M\ /_m\ H),(M\ /_m\ N)}$ ($Mp_{M\ H,N}$) = *carrier* ((*mdl M N*) $/_m$ *H*)

⟨*proof*⟩

lemma (**in** *Module*) *misom2Tr*:$[\![$*submodule R M H*; *submodule R M N*; $H \subseteq N]\!]$ $\Longrightarrow$

$(M\ /_m\ H)\ /_m\ (carrier\ ((mdl\ M\ N)\ /_m\ H)) \cong_R (M\ /_m\ N)$

⟨*proof*⟩

lemma (**in** *Module*) *eq-class-of-Submodule*:$[\![$*submodule R M H*; *submodule R M N*;

$H \subseteq N]\!] \Longrightarrow carrier\ ((mdl\ M\ N)\ /_m\ H) = N\ {}_s/_M\ H$

⟨*proof*⟩

theorem (**in** *Module*) *misom2*:$[\![$*submodule R M H*; *submodule R M N*; $H \subseteq N]\!]$ $\Longrightarrow$

$(M\ /_m\ H)\ /_m\ (N\ {}_s/_M\ H) \cong_R (M\ /_m\ N)$

⟨*proof*⟩

primrec *natm* :: ('*a*, '*m*) *aGroup-scheme* => *nat* ⇒ '*a* => '*a*
where
natm-0: $natm\ M\ 0\ x = \mathbf{0}_M$
| *natm-Suc*: $natm\ M\ (Suc\ n)\ x = (natm\ M\ n\ x) \pm_M x$

definition
finitesum-base :: [('*a*, '*r*, '*m*) *Module-scheme*, '*b set*, '*b* ⇒ '*a set*]
⇒ '*a set* **where**
$finitesum\text{-}base\ M\ I\ f = \bigcup\{f\ i \mid i.\ i \in I\}$

definition
finitesum :: [('*a*, '*r*, '*m*) *Module-scheme*, '*b set*, '*b* ⇒ '*a set*]
⇒ '*a set* **where**
$finitesum\ M\ I\ f = \{x.\ \exists n.\ \exists g.\ g \in \{j.\ j \leq (n::nat)\} \rightarrow finitesum\text{-}base\ M\ I\ f$
$\wedge\ x = nsum\ M\ g\ n\}$

lemma (**in** *Module*) *finitesumbase-sub-carrier*:$f \in I \rightarrow \{X.\ submodule\ R\ M\ X\}$ $\Longrightarrow$

$finitesum\text{-}base\ M\ I\ f \subseteq carrier\ M$

⟨*proof*⟩

lemma (**in** *Module*) *finitesum-sub-carrier*:$f \in I \rightarrow \{X.\ submodule\ R\ M\ X\} \Longrightarrow$
$finitesum\ M\ I\ f \subseteq carrier\ M$

⟨*proof*⟩

lemma (**in** *Module*) *finitesum-inc-zero*:$[\![f \in I \rightarrow \{X.\ submodule\ R\ M\ X\};\ I \neq \{\}]\!]$
$\Longrightarrow\ \mathbf{0} \in finitesum\ M\ I\ f$

⟨*proof*⟩

lemma (**in** *Module*) *finitesum-mOp-closed*:
$[\![f \in I \rightarrow \{X.\ submodule\ R\ M\ X\};\ I \neq \{\};\ a \in finitesum\ M\ I\ f]\!] \Longrightarrow$

$-_a\ a \in$ *finitesum M I f*
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-pOp-closed*:
$[\![f \in I \to \{X.\ submodule\ R\ M\ X\};\ a \in finitesum\ M\ I\ f;\ \ b \in finitesum\ M\ I\ f]\!]$
$\Longrightarrow\ a \pm b \in finitesum\ M\ I\ f$
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-sprodTr*:$[\![f \in I \to \{X.\ submodule\ R\ M\ X\};\ I \neq \{\};$
$r \in carrier\ R]\!]\ \Longrightarrow g \in \{j.\ j \leq (n::nat)\} \to (finitesum\text{-}base\ M\ I\ f)$
$\longrightarrow r \cdot_s (nsum\ M\ g\ n) =\ nsum\ M\ (\lambda x.\ r \cdot_s (g\ x))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-sprod*:$[\![f \in I \to \{X.\ submodule\ R\ M\ X\};\ I \neq \{\};$
$r \in carrier\ R;\ g \in \{j.\ j \leq (n::nat)\} \to (finitesum\text{-}base\ M\ I\ f)\ ]\!] \Longrightarrow$
$r \cdot_s (nsum\ M\ g\ n) =\ nsum\ M\ (\lambda x.\ r \cdot_s (g\ x))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-subModule*:$[\![f \in I \to \{X.\ submodule\ R\ M\ X\};\ I \neq$
$\{\}]\!]$
$\Longrightarrow submodule\ R\ M\ (finitesum\ M\ I\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-cont-H*:$[\![submodule\ R\ M\ H;\ submodule\ R\ M\ K]\!] \Longrightarrow$
$H \subseteq\ H \mp K$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-commute*:$[\![submodule\ R\ M\ H;\ submodule\ R\ M\ K]\!] \Longrightarrow$
$H \mp K =\ K \mp H$
⟨*proof*⟩

lemma (**in** *Module*) *Sum-of-SubmodulesTr*:$[\![submodule\ R\ M\ H;\ submodule\ R\ M$
$K]\!] \Longrightarrow$
$g \in \{j.\ j \leq (n::nat)\} \to H \cup K \longrightarrow \Sigma_e\ M\ g\ n \in H \mp K$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-two-Submodules*:$[\![submodule\ R\ M\ H;\ submodule\ R\ M$
$K]\!] \Longrightarrow$
$submodule\ R\ M\ (H \mp K)$
⟨*proof*⟩

definition
iotam :: $[('a, 'r, 'm)\ Module\text{-}scheme, 'a\ set, 'a\ set] \Rightarrow ('a \Rightarrow 'a)$
$((3\iota m_{-\ -,-})\ [82, 82, 83]82)$ **where**
$\iota m_{M\ H,K} = (\lambda x{\in}H.\ (x \pm_M \mathbf{0}_M))$

lemma (**in** *Module*) *iotam-mHom*:$[\![submodule\ R\ M\ H;\ submodule\ R\ M\ K]\!]$
$\Longrightarrow \iota m_{M\ H,K} \in mHom\ R\ (mdl\ M\ H)\ (mdl\ M\ (H \mp K))$
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr*:⟦*submodule R M H*; *submodule R M K*⟧ $\Longrightarrow$
submodule R (*mdl M* ($H \mp K$)) *K*
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr0*:⟦*submodule R M H*; *submodule R M K*⟧
$\Longrightarrow$ *compos* (*mdl M H*) (*mpj* (*mdl M* ($H \mp K$)) *K*) ($\iota m_{M\ H,K}$)
$\in$ *mHom R* (*mdl M H*) (*mdl M* ($H \mp K$) $/_m$ *K*)
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr1*:⟦*submodule R M H*; *submodule R M K*⟧ $\Longrightarrow$
$surjec_{(mdl\ M\ H),((mdl\ M\ (H \mp K))/_m\ K)}$
(*compos* (*mdl M H*) (*mpj* (*mdl M* ($H \mp K$)) *K*) ($\iota m_{M\ H,K}$))
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr2*:⟦*submodule R M H*; *submodule R M K*⟧ $\Longrightarrow$
$ker_{(mdl\ M\ H),((mdl\ M\ (H \mp K))\ /_m\ K)}$
(*compos* (*mdl M H*) (*mpj* (*mdl M* ($H \mp K$)) *K*) ($\iota m_{M\ H,K}$)) = $H \cap K$
⟨*proof*⟩

lemma (**in** *Module*) *mhomom-3*:⟦*submodule R M H*; *submodule R M K*⟧ $\Longrightarrow$
(*mdl M H*) $/_m$ ($H \cap K$) $\cong_R$ (*mdl M* ($H \mp K$)) $/_m$ *K*
⟨*proof*⟩

definition
l-comb :: [(′*r*, ′*m*) *Ring-scheme*, (′*a*, ′*r*, ′*m1*) *Module-scheme*, *nat*] ⇒
(*nat* ⇒ ′*r*) ⇒ (*nat* ⇒ ′*a*) ⇒ ′*a* **where**
l-comb R M n s m = *nsum M* ($\lambda j.$ (*s j*) $\cdot_{s\,M}$ (*m j*)) *n*

definition
linear-span :: [(′*r*, ′*m*) *Ring-scheme*, (′*a*, ′*r*, ′*m1*) *Module-scheme*, ′*r set*,
′*a set*] ⇒ ′*a set* **where**
linear-span R M A H = (*if H* = {} *then* {$\mathbf{0}_M$} *else*
{*x*. ∃ *n*. ∃ *f* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *H*.
∃ *s*∈{*j*. *j* ≤ (*n*::*nat*)} → *A*. *x* = *l-comb R M n s f*})

definition
coefficient :: [(′*r*, ′*m*) *Ring-scheme*, (′*a*, ′*r*, ′*m1*) *Module-scheme*,
nat, *nat* ⇒ ′*r*, *nat* ⇒ ′*a*] ⇒ *nat* ⇒ ′*r* **where**
coefficient R M n s m j = *s j*

definition
body :: [(′*r*, ′*m*) *Ring-scheme*, (′*a*, ′*r*, ′*m1*) *Module-scheme*, *nat*, *nat* ⇒ ′*r*,
nat ⇒ ′*a*] ⇒ *nat* ⇒ ′*a* **where**
body R M n s m j = *m j*

lemma (**in** *Module*) *l-comb-mem-linear-span*:⟦*ideal R A*; *H* ⊆ *carrier M*;
s ∈ {*j*. *j* ≤ (*n*::*nat*)} → *A*; *f* ∈ {*j*. *j* ≤ *n*} → *H*⟧ $\Longrightarrow$
l-comb R M n s f ∈ *linear-span R M A H*

⟨*proof*⟩

lemma (**in** *Module*) *linear-comb-eqTr*:$H \subseteq carrier\ M \Longrightarrow$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow carrier\ R\ \wedge$
$f \in \{j.\ j \leq n\} \rightarrow H\ \wedge$
$g \in \{j.\ j \leq n\} \rightarrow H\ \wedge$
$(\forall j \in \{j.\ j \leq n\}.\ f\ j = g\ j) \longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ g$
⟨*proof*⟩

lemma (**in** *Module*) *linear-comb-eq*:$[\![H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow carrier\ R;\ f \in \{j.\ j \leq n\} \rightarrow H;$
$g \in \{j.\ j \leq n\} \rightarrow H;\ \forall j \in \{j.\ j \leq n\}.\ f\ j = g\ j]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ g$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-Suc*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (Suc\ n)\} \rightarrow carrier\ R;\ f \in \{j.\ j \leq (Suc\ n)\} \rightarrow H]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ (Suc\ n)\ s\ f = l\text{-}comb\ R\ M\ n\ s\ f \pm s\ (Suc\ n) \cdot_s f\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jj*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A;\ f \in \{j.\ j \leq (n::nat)\} \rightarrow H;$
$t \in \{j.\ j \leq (m::nat)\} \rightarrow A;\ g \in \{j.\ j \leq (m::nat)\} \rightarrow H]\!] \Longrightarrow$
$nsum\ M\ (\lambda j.\ (jointfun\ n\ s\ m\ t)\ j \cdot_s (jointfun\ n\ f\ m\ g)\ j)\ n =$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s f\ j)\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jj1*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A;\ f \in \{j.\ j \leq (n::nat)\} \rightarrow H;$
$t \in \{j.\ j \leq (m::nat)\} \rightarrow A;\ g \in \{j.\ j \leq (m::nat)\} \rightarrow H]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ (jointfun\ n\ s\ m\ t)\ (jointfun\ n\ f\ m\ g) =$
$l\text{-}comb\ R\ M\ n\ s\ f$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jf*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A;\ f \in \{j.\ j \leq Suc\ (n + m)\} \rightarrow H;$
$t \in \{j.\ j \leq (m::nat)\} \rightarrow A]\!] \Longrightarrow$
$nsum\ M\ (\lambda j.\ (jointfun\ n\ s\ m\ t)\ j \cdot_s f\ j)\ n =$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s f\ j)\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jf1*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A;\ f \in \{j.\ j \leq Suc\ (n + m)\} \rightarrow H;$
$t \in \{j.\ j \leq (m::nat)\} \rightarrow A]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ (jointfun\ n\ s\ m\ t)\ f = l\text{-}comb\ R\ M\ n\ s\ f$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-fj*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$

$s \in \{j.\ j \leq Suc\ (n + m)\} \rightarrow A;\ f \in \{j.\ j \leq (n::nat)\} \rightarrow H;$
$g \in \{j.\ j \leq (m::nat)\} \rightarrow H]\!] \Longrightarrow$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s (jointfun\ n\ f\ m\ g)\ j)\ n =$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s f\ j)\ n$
$\langle proof \rangle$

lemma (**in** *Module*) *l-comb-jointfun-fj1*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq Suc\ (n + m)\} \rightarrow A;\ f \in \{j.\ j \leq (n::nat)\} \rightarrow H;$
$g \in \{j.\ j \leq (m::nat)\} \rightarrow H]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ (jointfun\ n\ f\ m\ g) = l\text{-}comb\ R\ M\ n\ s\ f$
$\langle proof \rangle$

lemma (**in** *Module*) *linear-comb0-1Tr*:$H \subseteq carrier\ M \Longrightarrow$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow \{\mathbf{0}_R\} \wedge$
$m \in \{j.\ j \leq n\} \rightarrow H \longrightarrow l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
$\langle proof \rangle$

lemma (**in** *Module*) *linear-comb0-1*:$[\![H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow \{\mathbf{0}_R\};\ m \in \{j.\ j \leq n\} \rightarrow H\ ]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
$\langle proof \rangle$

lemma (**in** *Module*) *linear-comb0-2Tr*:$ideal\ R\ A \Longrightarrow s \in \{j.\ j \leq (n::nat)\} \rightarrow A$
$\wedge\ \ m \in \{j.\ j \leq n\} \rightarrow \{\mathbf{0}_M\} \longrightarrow l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
$\langle proof \rangle$

lemma (**in** *Module*) *linear-comb0-2*:$[\![ideal\ R\ A;\ \ s \in \{j.\ j \leq (n::nat)\} \rightarrow A;$
$m \in \{j.\ j \leq n\} \rightarrow \{\mathbf{0}_M\}\ ]\!] \Longrightarrow\ l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
$\langle proof \rangle$

lemma (**in** *Module*) *liear-comb-memTr*:$[\![ideal\ R\ A;\ H \subseteq carrier\ M]\!] \Longrightarrow$
$\forall s.\ \forall m.\ s \in \{j.\ j \leq (n::nat)\} \rightarrow A \wedge$
$m \in \{j.\ j \leq n\} \rightarrow H \longrightarrow l\text{-}comb\ R\ M\ n\ s\ m \in carrier\ M$
$\langle proof \rangle$

lemma (**in** *Module*) *l-comb-mem*:$[\![ideal\ R\ A;\ H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A;\ m \in \{j.\ j \leq n\} \rightarrow H]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ m \in carrier\ M$
$\langle proof \rangle$

lemma (**in** *Module*) *l-comb-transpos*: $[\![ideal\ R\ A;\ H \subseteq carrier\ M;$
$s \in \{l.\ l \leq Suc\ n\} \rightarrow A;\ f \in \{l.\ l \leq Suc\ n\} \rightarrow H;$
$j < Suc\ n\ ]\!] \Longrightarrow$
$\Sigma_e\ M\ (cmp\ (\lambda k.\ s\ k \cdot_s f\ k)\ (transpos\ j\ (Suc\ n)))\ (Suc\ n) =$
$\Sigma_e\ M\ (\lambda k.\ (cmp\ s\ (transpos\ j\ (Suc\ n)))\ k \cdot_s$
$(cmp\ f\ (transpos\ j\ (Suc\ n)))\ k)\ (Suc\ n)$
$\langle proof \rangle$

lemma (**in** *Module*) *l-comb-transpos1*: $[\![ideal\ R\ A;\ H \subseteq carrier\ M;$

$s \in \{l.\ l \leq Suc\ n\} \rightarrow A;\ f \in \{l.\ l \leq Suc\ n\} \rightarrow H;\ j < Suc\ n\ \rrbracket \Longrightarrow$
l-comb $R\ M\ (Suc\ n)\ s\ f =$
l-comb $R\ M\ (Suc\ n)\ (cmp\ s\ (transpos\ j\ (Suc\ n)))\ (cmp\ f\ (transpos\ j\ (Suc\ n)))$
⟨*proof*⟩

lemma (**in** *Module*) *sc-linear-span*:$\llbracket ideal\ R\ A;\ H \subseteq carrier\ M;\ a \in A;$
$h \in H\rrbracket \Longrightarrow a \cdot_s h \in linear\text{-}span\ R\ M\ A\ H$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-cont-H*:$H \subseteq carrier\ M \Longrightarrow$
$H \subseteq linear\text{-}span\ R\ M\ (carrier\ R)\ H$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-inc-0*:$\llbracket ideal\ R\ A;\ H \subseteq carrier\ M\rrbracket \Longrightarrow$
$\mathbf{0} \in linear\text{-}span\ R\ M\ A\ H$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-iOp-closedTr1*:$\llbracket ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A\rrbracket \Longrightarrow$
$(\lambda x \in \{j.\ j \leq n\}.\ -_{a R}\ (s\ x)) \in \{j.\ j \leq n\} \rightarrow A$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-gen-mono*:$\llbracket K \subseteq H;\ H \subseteq carrier\ M;\ ideal\ R\ A\rrbracket \Longrightarrow$
$linear\text{-}span\ R\ M\ A\ K \subseteq linear\text{-}span\ R\ M\ A\ H$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-add*:$\llbracket ideal\ R\ A;\ H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A;\ f \in \{j.\ j \leq n\} \rightarrow H;$
$t \in \{j.\ j \leq (m::nat)\} \rightarrow A;\ g \in \{j.\ j \leq m\} \rightarrow H\rrbracket \Longrightarrow$
l-comb $R\ M\ (Suc\ (n + m))\ (jointfun\ n\ s\ m\ t)\ (jointfun\ n\ f\ m\ g) =$
l-comb $R\ M\ n\ s\ f \pm$ *l-comb* $R\ M\ m\ t\ g$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-add1Tr*:$\llbracket ideal\ R\ A;\ H \subseteq carrier\ M\rrbracket \Longrightarrow$
$f \in \{j.\ j \leq (n::nat)\} \rightarrow H \wedge s \in \{j.\ j \leq n\} \rightarrow A \wedge t \in \{j.\ j \leq n\} \rightarrow A \longrightarrow$
l-comb $R\ M\ n\ (\lambda x \in \{j.\ j \leq n\}.\ (s\ x) \pm_R (t\ x))\ f =$
l-comb $R\ M\ n\ s\ f \pm$ *l-comb* $R\ M\ n\ t\ f$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-add1*:$\llbracket ideal\ R\ A;\ H \subseteq carrier\ M;$
$f \in \{j.\ j \leq (n::nat)\} \rightarrow H;\ s \in \{j.\ j \leq n\} \rightarrow A;\ t \in \{j.\ j \leq n\} \rightarrow A\ \rrbracket \Longrightarrow$
l-comb $R\ M\ n\ (\lambda x \in \{j.\ j \leq n\}.\ (s\ x) \pm_R (t\ x))\ f =$
l-comb $R\ M\ n\ s\ f \pm$ *l-comb* $R\ M\ n\ t\ f$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-iOp-closedTr2*:$\llbracket ideal\ R\ A;\ H \subseteq carrier\ M;$
$f \in \{j.\ j \leq (n::nat)\} \rightarrow H;\ s \in \{j.\ j \leq n\} \rightarrow A\rrbracket \Longrightarrow$
$-_a$ (*l-comb* $R\ M\ n\ s\ f) =$
l-comb $R\ M\ n\ (\lambda x \in \{j.\ j \leq n\}.\ -_{a R}\ (s\ x))\ f$

⟨proof⟩

lemma (**in** *Module*) *linear-span-iOp-closed*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*;
$a \in$ *linear-span R M A H*$]\!] \Longrightarrow -_a\ a \in$ *linear-span R M A H*
⟨proof⟩

lemma (**in** *Module*) *linear-span-pOp-closed*:
$[\![$*ideal R A*; $H \subseteq$ *carrier M*; $a \in$ *linear-span R M A H*; $b \in$ *linear-span R M A H*$]\!]$
$\Longrightarrow a \pm b \in$ *linear-span R M A H*
⟨proof⟩

lemma (**in** *Module*) *l-comb-scTr*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*;
$r \in$ *carrier R*; $H \neq \{\}]\!] \Longrightarrow s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$g \in \{j.\ j \leq n\} \to H \longrightarrow r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ n) =$
$nsum\ M\ (\lambda k.\ r \cdot_s ((s\ k) \cdot_s (g\ k)))\ n$
⟨proof⟩

lemma (**in** *Module*) *l-comb-sc1Tr*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*;
$r \in$ *carrier R*; $H \neq \{\}]\!] \Longrightarrow s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$g \in \{j.\ j \leq n\} \to H \longrightarrow r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ n) =$
$nsum\ M\ (\lambda k.\ (r \cdot_{rR} (s\ k)) \cdot_s (g\ k))\ n$
⟨proof⟩

lemma (**in** *Module*) *l-comb-sc*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*; $r \in$ *carrier R*;
$s \in \{j.\ j \leq (n::nat)\} \to A;\ g \in \{j.\ j \leq n\} \to H]\!] \Longrightarrow$
$r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ n) = nsum\ M\ (\lambda k.\ r \cdot_s ((s\ k) \cdot_s (g\ k)))\ n$
⟨proof⟩

lemma (**in** *Module*) *l-comb-sc1*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*; $r \in$ *carrier R*;
$s \in \{j.\ j \leq (n::nat)\} \to A;\ g \in \{j.\ j \leq n\} \to H]\!] \Longrightarrow$
$r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ n) = nsum\ M\ (\lambda k.\ (r \cdot_{rR} (s\ k)) \cdot_s (g\ k))\ n$
⟨proof⟩

lemma (**in** *Module*) *linear-span-sc-closed*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*;
$r \in$ *carrier R*; $x \in$ *linear-span R M A H*$]\!] \Longrightarrow r \cdot_s x \in$ *linear-span R M A H*
⟨proof⟩

lemma (**in** *Module*) *mem-single-l-spanTr*:$[\![$*ideal R A*; $h \in$ *carrier M*$]\!] \Longrightarrow$
$s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$f \in \{j.\ j \leq n\} \to \{h\} \wedge$ *l-comb R M n s f* $\in$ *linear-span R M A* $\{h\}$
$\longrightarrow (\exists\, a \in A.$ *l-comb R M n s f* $= a \cdot_s h)$
⟨proof⟩

lemma (**in** *Module*) *mem-single-l-span*:$[\![$*ideal R A*; $h \in$ *carrier M*;
$s \in \{j.\ j \leq (n::nat)\} \to A;\ f \in \{j.\ j \leq n\} \to \{h\}$;
l-comb R M n s f $\in$ *linear-span R M A* $\{h\}]\!] \Longrightarrow$
$\exists\, a \in A.$ *l-comb R M n s f* $= a \cdot_s h$
⟨proof⟩

lemma (**in** *Module*) *mem-single-l-span1*: $[\![$*ideal R A*; $h \in$ *carrier M*;
$x \in$ *linear-span R M A* $\{h\}]\!] \Longrightarrow \exists\, a \in A.\ x = a \cdot_s h$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-subModule*: $[\![$*ideal R A*; $H \subseteq$ *carrier M*$]\!] \Longrightarrow$
submodule R M (*linear-span R M A H*)
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-mem-submoduleTr*: $[\![$*ideal R A*; *submodule R M N*$]\!] \Longrightarrow$
$(s \in \{j.\ j \le (n{::}nat)\} \to A \wedge f \in \{j.\ j \le n\} \to$ *carrier M* $\wedge$
$(\forall j \le n.(s\ j) \cdot_s (f\ j) \in N)) \longrightarrow$ *l-comb R M n s f* $\in N$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-sub-submodule*: $[\![$*ideal R A*; *submodule R M N*; $H \subseteq N]\!]$
$\Longrightarrow$
linear-span R M A H $\subseteq N$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-sub*: $[\![$*ideal R A*; $H \subseteq$ *carrier M*$]\!] \Longrightarrow$
(*linear-span R M A H*) $\subseteq$ *carrier M*
⟨*proof*⟩

definition
smodule-ideal-coeff :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*,
'r set] $\Rightarrow$ *'a set* **where**
smodule-ideal-coeff R M A = *linear-span R M A* (*carrier M*)

abbreviation
SMLIDEALCOEFF ((*3-/* $\odot_{-}$ *-*) [*64*,*64*,*65*]*64*) **where**
$A \odot_R M$ == *smodule-ideal-coeff R M A*

lemma (**in** *Module*) *smodule-ideal-coeff-is-Submodule*: *ideal R A* $\Longrightarrow$
submodule R M $(A \odot_R M)$
⟨*proof*⟩

lemma (**in** *Module*) *mem-smodule-ideal-coeff*: $[\![$*ideal R A*; $x \in A \odot_R M]\!] \Longrightarrow$
$\exists\, n.\ \exists\, s \in \{j.\ j \le n\} \to A.\ \exists\, g \in \{j.\ j \le n\} \to$ *carrier M*.
$x =$ *l-comb R M n s g*
⟨*proof*⟩

definition
quotient-of-submodules :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*,
'a set, *'a set*] $\Rightarrow$ *'r set* **where**
quotient-of-submodules R M N P = $\{x \mid x.\ x \in$ *carrier R* $\wedge$
(*linear-span R M* (*Rxa R x*)   *P*) $\subseteq N\}$

definition
Annihilator :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*]

$\Rightarrow$ *'r set* ((*Ann*$_{-}$ -) [*82*,*83*]*82*) **where**
Ann$_R$ *M* = *quotient-of-submodules R M* $\{\mathbf{0}_M\}$ (*carrier M*)

abbreviation
QOFSUBMDS ((*4*- $_{-\ddagger-}$ -) [*82*,*82*,*82*,*83*]*82*) **where**
N $_{R\ddagger M}$ *P* == *quotient-of-submodules R M N P*

lemma (**in** *Module*) *quotient-of-submodules-inc-0*:
⟦*submodule R M P*; *submodule R M Q*⟧ $\Longrightarrow$ $\mathbf{0}_R \in$ (*P* $_{R\ddagger M}$ *Q*)
⟨*proof*⟩

lemma (**in** *Module*) *quotient-of-submodules-is-ideal*:
⟦*submodule R M P*; *submodule R M Q*⟧ $\Longrightarrow$ *ideal R* (*P* $_{R\ddagger M}$ *Q*)
⟨*proof*⟩

lemma (**in** *Module*) *Ann-is-ideal*:*ideal R* (*Ann*$_R$ *M*)
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-of-lincombTr*:⟦*ideal R A*; *R module N*;
f $\in$ *mHom R M N*; *H* $\subseteq$ *carrier M*⟧ $\Longrightarrow$
s $\in$ {*j*. *j* $\le$ (*n*::*nat*)} $\rightarrow$ *A* $\wedge$ *g* $\in$ {*j*. *j* $\le$ *n*} $\rightarrow$ *H* $\longrightarrow$
f (*l-comb R M n s g*) = *l-comb R N n s* (*cmp f g*)
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-lincomb*:⟦*ideal R A*; *R module N*; *f* $\in$ *mHom R M N*;
H $\subseteq$ *carrier M*; *s* $\in$ {*j*. *j* $\le$ (*n*::*nat*)} $\rightarrow$ *A*; *g* $\in$ {*j*. *j* $\le$ *n*} $\rightarrow$ *H* ⟧ $\Longrightarrow$
f (*l-comb R M n s g*) = *l-comb R N n s* (*cmp f g*)
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-linspan*:⟦*ideal R A*; *R module N*; *f* $\in$ *mHom R M N*;
H $\subseteq$ *carrier M*; *s* $\in$ {*j*. *j* $\le$ (*n*::*nat*)} $\rightarrow$ *A*; *g* $\in$ {*j*. *j* $\le$ *n*} $\rightarrow$ *H* ⟧ $\Longrightarrow$
f (*l-comb R M n s g*) $\in$ *linear-span R N A* (*f* ‘ *H*)
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-linspan1*:⟦*ideal R A*; *R module N*; *f* $\in$ *mHom R M N*;
H $\subseteq$ *carrier M*; *h* $\in$ *linear-span R M A H*⟧ $\Longrightarrow$
f h $\in$ *linear-span R N A* (*f* ‘ *H*)
⟨*proof*⟩

definition
faithful :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*]
$\Rightarrow$ *bool* **where**
faithful R M $\longleftrightarrow$ *Ann*$_R$ *M* = $\{\mathbf{0}_R\}$

5.3 nsum and Generators

definition
generator :: $[('r, 'm)$ *Ring-scheme*, $('a, 'r, 'm1)$ *Module-scheme*,
$'a$ *set*$] \Rightarrow$ *bool* **where**
generator R M H $==$ $H \subseteq$ *carrier M* $\wedge$
linear-span R M (*carrier R*) $H =$ *carrier M*

definition
finite-generator :: $[('r, 'm)$ *Ring-scheme*, $('a, 'r, 'm1)$ *Module-scheme*,
$'a$ *set*$] \Rightarrow$ *bool* **where**
finite-generator R M H $\longleftrightarrow$ *finite H* $\wedge$ *generator R M H*

definition
fGOver :: $[('a, 'r, 'm1)$ *Module-scheme*, $('r, 'm)$ *Ring-scheme*$] \Rightarrow$ *bool*
where
fGOver M R $\longleftrightarrow$ ($\exists H$. *finite-generator R M H*)

abbreviation
FGENOVER (**infixl** *fgover 70*) **where**
M fgover R $==$ *fGOver M R*

lemma (**in** *Module*) *h-in-linear-span*:$[\![H \subseteq$ *carrier M*; $h \in H]\!] \Longrightarrow$
$h \in$ *linear-span R M* (*carrier R*) *H*
⟨*proof*⟩

lemma (**in** *Module*) *generator-sub-carrier*:*generator R M H* $\Longrightarrow$
$H \subseteq$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *lin-span-sub-carrier*:$[\![$*ideal R A*;
$H \subseteq$ *carrier M*$]\!] \Longrightarrow$ *linear-span R M A H* $\subseteq$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *lin-span-coeff-mono*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*$]\!] \Longrightarrow$
linear-span R M A H $\subseteq$ *linear-span R M* (*carrier R*) *H*
⟨*proof*⟩

lemma (**in** *Module*) *l-span-sum-closedTr*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*$]\!] \Longrightarrow$
$\forall s. \forall f. s \in \{j. j \leq (n::nat)\} \rightarrow A \wedge$
$f \in \{j. j \leq n\} \rightarrow$ *linear-span R M A H* $\longrightarrow$
(*nsum M* ($\lambda j.\ s\ j \cdot_s (f\ j)$) $n \in$ *linear-span R M A H*)
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed*:$[\![$*ideal R A*; $H \subseteq$ *carrier M*;
$s \in \{j. j \leq (n::nat)\} \rightarrow A$; $f \in \{j. j \leq n\} \rightarrow$ *linear-span R M A H* $]\!] \Longrightarrow$
l-comb R M n s f $\in$ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed1*:⟦$H \subseteq$ *carrier* M;
$s \in \{j.\ j \leq (n{::}nat)\} \rightarrow$ *carrier* R;
$f \in \{j.\ j \leq n\} \rightarrow$ *linear-span* R M (*carrier* R) H ⟧ $\Longrightarrow$
Σ_e M $(\lambda j.\ \ s\ j \cdot_s (f\,j))$ $n \in$ *linear-span* R M (*carrier* R) H
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed2Tr0*:⟦*ideal* R A; $H \subseteq$ *carrier* M; *Ring* R; $s \in A$;
$f \in$ *linear-span* R M (*carrier* R) H ⟧ $\Longrightarrow$ $s \cdot_s f \in$ *linear-span* R M A H
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed2Tr*:⟦*ideal* R A; $H \subseteq$ *carrier* M⟧ $\Longrightarrow$
$s \in \{j.\ j \leq (n{::}nat)\} \rightarrow A\ \wedge$
$f \in \{j.\ j \leq n\} \rightarrow$ *linear-span* R M (*carrier* R) H $\longrightarrow$
l-comb R M n s f $\in$ *linear-span* R M A H
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed2*:⟦*ideal* R A; $H \subseteq$ *carrier* M;
$s \in \{j.\ j \leq (n{::}nat)\} \rightarrow A$;
$f \in \{j.\ j \leq n\} \rightarrow$ *linear-span* R M (*carrier* R) H⟧ $\Longrightarrow$
l-comb R M n s f $\in$ *linear-span* R M A H
⟨*proof*⟩

lemma (**in** *Module*) *l-span-l-span*:$H \subseteq$ *carrier* M $\Longrightarrow$
linear-span R M (*carrier* R) (*linear-span* R M (*carrier* R) H) $=$
linear-span R M (*carrier* R) H
⟨*proof*⟩

lemma (**in** *Module*) *l-spanA-l-span*:⟦*ideal* R A; $H \subseteq$ *carrier* M⟧ $\Longrightarrow$
linear-span R M A (*linear-span* R M (*carrier* R) H) $=$
linear-span R M A H
⟨*proof*⟩

lemma (**in** *Module*) *l-span-zero*:*ideal* R A $\Longrightarrow$ *linear-span* R M A $\{\mathbf{0}\} = \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed3*:⟦*ideal* R A; *generator* R M H;
$A \odot_R M =$ *carrier* M⟧ $\Longrightarrow$ *linear-span* R M A H $=$ *carrier* M
⟨*proof*⟩

lemma (**in** *Module*) *generator-generator*:⟦*generator* R M H; $H1 \subseteq$ *carrier* M;
$H \subseteq$ *linear-span* R M (*carrier* R) $H1$⟧ $\Longrightarrow$ *generator* R M $H1$
⟨*proof*⟩

lemma (**in** *Module*) *generator-elimTr*:
$f \in \{j.\ j \leq (n{::}nat)\} \rightarrow$ *carrier* M $\wedge$ *generator* R M $(f\ \text{`}\ \{j.\ j \leq n\})$ $\wedge$
$(\forall\, i{\in}nset\ (Suc\ 0)\ n.\ f\ i \in$
linear-span R M (*carrier* R) $(f\ \text{`}\ \{j.\ j \leq (i - Suc\ 0)\}))$ $\longrightarrow$
linear-span R M (*carrier* R) $\{f\ 0\}$ $=$ *carrier* M
⟨*proof*⟩

lemma (**in** *Module*) *generator-generator-elim*:
$[\![f \in \{j.\ j \leq (n::nat)\} \rightarrow carrier\ M;\ generator\ R\ M\ (f\ `\ \{j.\ j \leq n\});$
$(\forall i \in nset\ (Suc\ 0)\ n.\ f\ i \in linear\text{-}span\ R\ M\ (carrier\ R)$
$(f\ `\ \{j.\ j \leq (i - Suc\ 0)\}))]\!] \Longrightarrow$
$linear\text{-}span\ R\ M\ (carrier\ R)\ \{f\ 0\} = carrier\ M$
⟨*proof*⟩

lemma (**in** *Module*) *surjec-generator*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;$
$surjec_{M,N}\ f;\ generator\ R\ M\ H]\!] \Longrightarrow generator\ R\ N\ (f\ `\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *surjec-finitely-gen*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;$
$surjec_{M,N}\ f;\ M\ fgover\ R]\!] \Longrightarrow N\ fgover\ R$
⟨*proof*⟩

5.3.1 Sum up coefficients

Symbolic calculation.

lemma (**in** *Module*) *similar-termTr*:$[\![ideal\ R\ A;\ a \in A]\!] \Longrightarrow$
$\forall s.\ \forall f.\ s \in \{j.\ j \leq (n::nat)\} \rightarrow A\ \wedge$
$f \in \{j.\ j \leq n\} \rightarrow carrier\ M\ \wedge$
$m \in f\ `\ \{j.\ j \leq n\} \longrightarrow$
$(\exists t \in \{j.\ j \leq n\} \rightarrow A.\ nsum\ M\ (\lambda j.\ s\ j \cdot_s (f\ j))\ n \pm a \cdot_s m =$
$nsum\ M\ (\lambda j.\ t\ j \cdot_s (f\ j))\ n\ )$
⟨*proof*⟩

lemma (**in** *Module*) *similar-term1*:$[\![ideal\ R\ A;\ a \in A;\ s \in \{j.\ j \leq (n::nat)\} \rightarrow A;$
$f \in \{j.\ j \leq n\} \rightarrow carrier\ M;\ m \in f\ `\ \{j.\ j \leq n\}]\!] \Longrightarrow$
$\exists t \in \{j.\ j \leq n\} \rightarrow A.\ \Sigma_e\ M\ (\lambda j.\ s\ j \cdot_s (f\ j))\ n \pm a \cdot_s m =$
$\Sigma_e\ M\ (\lambda j.\ t\ j \cdot_s (f\ j))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *same-togetherTr*:$[\![ideal\ R\ A;\ H \subseteq carrier\ M\ ]\!] \Longrightarrow$
$\forall s.\ \forall f.\ s \in \{j.\ j \leq (n::nat)\} \rightarrow A\ \wedge f \in \{j.\ j \leq n\} \rightarrow H \longrightarrow$
$(\exists t \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\} \rightarrow A.$
$\exists g \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\} \rightarrow f\ `\ \{j.\ j \leq n\}.$
$surj\text{-}to\ g\ \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\}\ (f\ `\ \{j.\ j \leq n\})\ \wedge$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s (f\ j))\ n = nsum\ M\ (\lambda k.\ t\ k \cdot_s (g\ k))$
$(card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0))$
⟨*proof*⟩

lemma (**in** *Module*) *same-together*:$[\![ideal\ R\ A;\ H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (n::nat)\} \rightarrow A;\ f \in \{j.\ j \leq n\} \rightarrow H]\!] \Longrightarrow$
$\exists t \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq (n::nat)\}) - Suc\ 0)\} \rightarrow A.$
$\exists g \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\} \rightarrow f\ `\ \{j.\ j \leq n\}.$
$surj\text{-}to\ g\ \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\}\ (f\ `\ \{j.\ j \leq n\})\ \wedge$

$\Sigma_e\ M\ (\lambda j.\ s\ j\ \cdot_s\ (f\ j))\ n =$
$\Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)$
⟨proof⟩

lemma (**in** *Module*) *one-last*:$[\![ideal\ R\ A;\ H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (Suc\ n)\} \rightarrow A;\ f \in \{j.\ j \leq (Suc\ n)\} \rightarrow H;$
$bij\text{-}to\ f\ \{j.\ j \leq (Suc\ n)\}\ H;\ j \leq (Suc\ n);\ j \neq (Suc\ n)]\!] \Longrightarrow$
$\exists\, t \in \{j.\ j \leq (Suc\ n)\} \rightarrow A.\ \exists\, g \in \{j.\ j \leq (Suc\ n)\} \rightarrow H.$
$\Sigma_e\ M\ (\lambda k.\ s\ k\ \cdot_s\ (f\ k))\ (Suc\ n) = \Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (Suc\ n) \wedge$
$g\ (Suc\ n) = f\ j \wedge t\ (Suc\ n) = s\ j \wedge bij\text{-}to\ g\ \{j.\ j \leq (Suc\ n)\}\ H$
⟨proof⟩

lemma (**in** *Module*) *finite-lin-spanTr1*:$[\![ideal\ R\ A;\ z \in carrier\ M]\!] \Longrightarrow$
$h \in \{j.\ j \leq (n::nat)\} \rightarrow \{z\} \wedge t \in \{j.\ j \leq n\} \rightarrow A \longrightarrow$
$(\exists\, s \in \{0::nat\} \rightarrow A.\ \Sigma_e\ M\ (\lambda j.\ t\ j\ \cdot_s\ (h\ j))\ n = s\ 0\ \cdot_s\ z)$
⟨proof⟩

lemma (**in** *Module*) *single-span*:$[\![ideal\ R\ A;\ z \in carrier\ M;$
$h \in \{j.\ j \leq (n::nat)\} \rightarrow \{z\};\ t \in \{j.\ j \leq n\} \rightarrow A]\!] \Longrightarrow$
$\exists\, s \in \{0::nat\} \rightarrow A.\ \Sigma_e\ M\ (\lambda j.\ t\ j\ \cdot_s\ (h\ j))\ n = s\ 0\ \cdot_s\ z$
⟨proof⟩

definition
$coeff\text{-}at\text{-}k :: [('r,\ 'm)\ Ring\text{-}scheme,\ 'r,\ nat] \Rightarrow (nat \Rightarrow 'r)$ **where**
$coeff\text{-}at\text{-}k\ R\ a\ k = (\lambda j.\ if\ j = k\ then\ a\ else\ (\mathbf{0}_R))$

lemma *card-Nset-im*:$f \in \{j.\ j \leq (n::nat)\} \rightarrow A \Longrightarrow$
$(Suc\ 0) \leq card\ (f\ `\{j.\ j \leq n\})$
⟨proof⟩

lemma (**in** *Module*) *eSum-changeTr1*:$[\![ideal\ R\ A;$
$t \in \{k.\ k \leq (card\ (f\ `\ \{j.\ j \leq (n1::nat)\}) - Suc\ 0)\} \rightarrow A;$
$g \in \{k.\ k \leq (card\ (f\ `\ \{j.\ j \leq n1\}) - Suc\ 0)\} \rightarrow f\ `\{j.\ j \leq n1\};$
$Suc\ 0 < card\ (f\ `\{j.\ j \leq n1\});\ g\ x = h\ (Suc\ n);\ x = Suc\ n;$
$card\ (f\ `\{j.\ j \leq n1\}) - Suc\ 0 = Suc\ (card\ (f\ `\ \{j.\ j \leq n1\}) - Suc\ 0 - Suc\ 0)]\!]$
$\Longrightarrow$
$\Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (card\ (f\ `\ \{j.\ j \leq n1\}) - Suc\ 0) =$
$\Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (card\ (f\ `\ \{j.\ j \leq n1\}) - Suc\ 0 - Suc\ 0) \pm$
$(t\ (Suc\ (card\ (f\ `\ \{j.\ j \leq n1\}) - Suc\ 0 - Suc\ 0))\ \cdot_s$
$(g\ (\ Suc\ (card\ (f\ `\ \{j.\ j \leq n1\}) - Suc\ 0 - Suc\ 0))))$
⟨proof⟩

definition
$zeroi :: [('r,\ 'm)\ Ring\text{-}scheme] \Rightarrow nat \Rightarrow 'r$ **where**
$zeroi\ R = (\lambda j.\ \mathbf{0}_R)$

lemma *zeroi-func*:$[\![Ring\ R;\ ideal\ R\ A]\!] \Longrightarrow zeroi\ R \in \{j.\ j \leq 0\} \rightarrow A$
⟨proof⟩

lemma (**in** *Module*) *prep-arrTr1*: $\llbracket$*ideal R A*; $h \in \{j.\ j \le (Suc\ n)\} \to carrier\ M$;
$f \in \{j.\ j \le (n1::nat)\} \to h\ `\ \{j.\ j \le (Suc\ n)\}$; $s \in \{j.\ j \le n1\} \to A$;
$m = l\text{-}comb\ R\ M\ n1\ s\ f\rrbracket \Longrightarrow$
$\exists l \in \{j.\ j \le (Suc\ n)\}.\ (\exists s \in \{j.\ j \le (l::nat)\} \to A.$
$\exists g \in \{j.\ j \le l\} \to h\ `\{j.\ j \le (Suc\ n)\}.\ m = l\text{-}comb\ R\ M\ l\ s\ g \wedge$
$bij\text{-}to\ g\ \{j.\ j \le l\}\ (f\ `\ \{j.\ j \le n1\}))$
⟨*proof*⟩

lemma *two-func-imageTr*: $\llbracket\ h \in \{j.\ j \le Suc\ n\} \to B$;
$f \in \{j.\ j \le (m::nat)\} \to h\ `\ \{j.\ j \le Suc\ n\}$; $h\ (Suc\ n) \notin f\ `\ \{j.\ j \le m\}\rrbracket$
$\Longrightarrow f \in \{j.\ j \le m\} \to h\ `\ \{j.\ j \le n\}$
⟨*proof*⟩

lemma (**in** *Module*) *finite-lin-spanTr3-0*: $\llbracket bij\text{-}to\ g\ \{j.\ j \le l\}\ (g\ `\{j.\ j \le l\})$;
ideal R A;
$\forall na.\ \forall s \in \{j.\ j \le na\} \to A.$
$\forall f \in \{j.\ j \le na\} \to h\ `\ \{j.\ j \le n\}.$
$\exists t \in \{j.\ j \le n\} \to A.\ l\text{-}comb\ R\ M\ na\ s\ f = l\text{-}comb\ R\ M\ n\ t\ h$;
$h \in \{j.\ j \le Suc\ n\} \to carrier\ M$; $s \in \{j.\ j \le m\} \to A$;
$f \in \{j.\ j \le m\} \to h\ `\ \{j.\ j \le Suc\ n\}$;
$l \le Suc\ n$; $sa \in \{j.\ j \le l\} \to A$; $g \in \{j.\ j \le l\} \to h\ `\ \{j.\ j \le Suc\ n\}$;
$0 < l$; $f\ `\ \{j.\ j \le m\} = g\ `\ \{j.\ j \le l\}$; $h\ (Suc\ n) = g\ l\rrbracket$
$\Longrightarrow \exists t \in \{j.\ j \le Suc\ n\} \to A.\ l\text{-}comb\ R\ M\ l\ sa\ g = l\text{-}comb\ R\ M\ (Suc\ n)\ t\ h$
⟨*proof*⟩

lemma (**in** *Module*) *finite-lin-spanTr3*: *ideal R A* $\Longrightarrow$
$h \in \{j.\ j \le (n::nat)\} \to carrier\ M \longrightarrow$
$(\forall na.\ \forall s \in \{j.\ j \le (na::nat)\} \to A.$
$\forall f \in \{j.\ j \le na\} \to (h\ `\ \{j.\ j \le n\}).\ (\exists t \in \{j.\ j \le n\} \to A.$
$l\text{-}comb\ R\ M\ na\ s\ f = l\text{-}comb\ R\ M\ n\ t\ h))$
⟨*proof*⟩

lemma (**in** *Module*) *finite-lin-span*:
$\llbracket$*ideal R A*; $h \in \{j.\ j \le (n::nat)\} \to carrier\ M$; $s \in \{j.\ j \le (n1::nat)\} \to A$;
$f \in \{j.\ j \le n1\} \to h\ `\ \{j.\ j \le n\}\rrbracket \Longrightarrow \exists t \in \{j.\ j \le n\} \to A.$
$l\text{-}comb\ R\ M\ n1\ s\ f = l\text{-}comb\ R\ M\ n\ t\ h$
⟨*proof*⟩

5.3.2 Free generators

definition
free-generator :: $[('r, 'm)\ Ring\text{-}scheme, ('a, 'r, 'm1)\ Module\text{-}scheme, 'a\ set]$
$\Rightarrow$ *bool* **where**
$free\text{-}generator\ R\ M\ H \longleftrightarrow generator\ R\ M\ H \wedge$
$(\forall n.\ (\forall s\ f.\ (s \in \{j.\ j \le (n::nat)\} \to carrier\ R \wedge$
$f \in \{j.\ j \le n\} \to H \wedge inj\text{-}on\ f\ \{j.\ j \le n\} \wedge$
$l\text{-}comb\ R\ M\ n\ s\ f = \mathbf{0}_M) \longrightarrow s \in \{j.\ j \le n\} \to \{\mathbf{0}_R\}))$

lemma (**in** *Module*) *free-generator-generator*:*free-generator R M H* $\Longrightarrow$
generator R M H
⟨*proof*⟩

lemma (**in** *Module*) *free-generator-sub*:*free-generator R M H* $\Longrightarrow$
$H \subseteq$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *free-generator-nonzero*:$[\![\neg$ (*zeroring R*);
free-generator R M H; $h \in H]\!] \Longrightarrow h \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Module*) *has-free-generator-nonzeroring*: $[\![$*free-generator R M H*;
$\exists\, p \in$ *linear-span R M* (*carrier R*) *H*. $p \neq \mathbf{0}$ $]\!]$ $\Longrightarrow \neg$ *zeroring R*
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression1*:$[\![H \subseteq$ *carrier M*; *free-generator R M H*;
$s \in \{j.\ j \leq (n::nat)\} \to$ *carrier R*; $m \in \{j.\ j \leq n\} \to H$;
inj-on m $\{j.\ j \leq n\}$; *l-comb R M n s m* $= \mathbf{0}]\!] \Longrightarrow$
$\forall j{\in}\{j.\ j \leq n\}.\ s\ j = \mathbf{0}_R$
⟨*proof*⟩

lemma (**in** *Module*) *free-gen-coeff-zero*:$[\![H \subseteq$ *carrier M*; *free-generator R M H*;
$h \in H$; $a \in$ *carrier R*; $a \cdot_s h = \mathbf{0}]\!] \Longrightarrow a = \mathbf{0}_R$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression2*:$[\![H \subseteq$ *carrier M*;
$f \in \{j.\ j \leq (n::nat)\} \to H$; $s \in \{j.\ j \leq n\} \to$ *carrier R*$]\!] \Longrightarrow$
$\exists\, m\ g\ t.\ g \in (\{j.\ j \leq (m::nat)\} \to H) \wedge$
bij-to g $\{j.\ j \leq (m::nat)\}$ $(f$ ‘ $\{j.\ j \leq n\}) \wedge$
$t \in \{j.\ j \leq m\} \to$ *carrier R* $\wedge$
l-comb R M n s f $=$ *l-comb R M m t g*
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression3-1*:$[\![H \subseteq$ *carrier M*;
$f \in \{l.\ l \leq (Suc\ n)\} \to H$; $s \in \{l.\ l \leq (Suc\ n)\} \to$ *carrier R*;
$(f\ (Suc\ n)) \notin f$ ‘$(\{l.\ l \leq (Suc\ n)\} - \{Suc\ n\})]\!] \Longrightarrow$
$\exists\, g\ m\ t.\ g \in \{l.\ l \leq (m::nat)\} \to H \wedge$
inj-on g $\{l.\ l \leq (m::nat)\} \wedge$
$t \in \{l.\ l \leq (m::nat)\} \to$ *carrier R* $\wedge$
l-comb R M $(Suc\ n)$ $s\ f =$
l-comb R M m t g $\wedge\ t\ m = s\ (Suc\ n) \wedge g\ m = f\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression3-2*:$[\![H \subseteq$ *carrier M*;
$f \in \{k.\ k \leq (Suc\ n)\} \to H$; $s \in \{k.\ k \leq (Suc\ n)\} \to$ *carrier R*;
$l \leq (Suc\ n)$; $(f\ l) \notin f$ ‘ $(\{k.\ k \leq (Suc\ n)\} - \{l\})$; $l \neq Suc\ n]\!] \Longrightarrow$
$\exists\, g\ m\ t.\ g \in \{l.\ l \leq (m::nat)\} \to H \wedge$ *inj-on g* $\{l.\ l \leq (m::nat)\} \wedge$

$t \in \{l.\ l \leq m\} \rightarrow carrier\ R\ \wedge$
$l\text{-}comb\ R\ M\ (Suc\ n)\ s\ f = l\text{-}comb\ R\ M\ m\ t\ g\ \wedge$
$t\ m = s\ l \wedge g\ m = f\ l$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression3*:
$[\![H \subseteq carrier\ M;\ f \in \{k.\ k \leq (Suc\ n)\} \rightarrow H;$
$s \in \{k.\ k \leq (Suc\ n)\} \rightarrow carrier\ R;\ l \leq (Suc\ n);$
$(f\ l) \notin f\ `\ (\{k.\ k \leq (Suc\ n)\} - \{l\})]\!] \Longrightarrow$
$\exists\, g\ m\ t.\ g \in \{k.\ k \leq (m::nat)\} \rightarrow H\ \wedge$
$inj\text{-}on\ g\ \{k.\ k \leq m\}\ \wedge$
$t \in \{k.\ k \leq m\} \rightarrow carrier\ R\ \wedge$
$l\text{-}comb\ R\ M\ (Suc\ n)\ s\ f = l\text{-}comb\ R\ M\ m\ t\ g \wedge t\ m = s\ l \wedge g\ m = f\ l$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression4*:$free\text{-}generator\ R\ M\ H \Longrightarrow$
$f \in \{k.\ k \leq (n::nat)\} \rightarrow H \wedge inj\text{-}on\ f\ \{k.\ k \leq n\}\ \wedge$
$s \in \{k.\ k \leq n\} \rightarrow carrier\ R \wedge l\text{-}comb\ R\ M\ n\ s\ f \neq \mathbf{0} \longrightarrow$
$(\exists\, m\ g\ t.\ (g \in \{k.\ k \leq m\} \rightarrow H) \wedge inj\text{-}on\ g\ \{k.\ k \leq m\}\ \wedge$
$(g\ `\ \{k.\ k \leq m\} \subseteq f\ `\ \{k.\ k \leq n\}) \wedge (t \in \{k.\ k \leq m\} \rightarrow carrier\ R)\ \wedge$
$(\forall\, l \in \{k.\ k \leq m\}.\ t\ l \neq \mathbf{0}_R) \wedge l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ m\ t\ g)$
⟨*proof*⟩

lemma (**in** *Module*) *unique-prepression5-0*:$[\![free\text{-}generator\ R\ M\ H;$
$f \in \{j.\ j \leq n\} \rightarrow H;\ inj\text{-}on\ f\ \{j.\ j \leq n\};$
$s \in \{j.\ j \leq n\} \rightarrow carrier\ R;\ g \in \{j.\ j \leq m\} \rightarrow H;$
$inj\text{-}on\ g\ \{j.\ j \leq m\};\ t \in \{j.\ j \leq m\} \rightarrow carrier\ R;$
$l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ m\ t\ g; \forall\, j{\leq}n.\ s\ j \neq \mathbf{0}_R;\ \forall\, k{\leq}m.\ t\ k \neq \mathbf{0}_R;$
$f\ n \notin g\ `\ \{j.\ j \leq m\};\ 0 < n]\!] \Longrightarrow False$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression5*:$[\![free\text{-}generator\ R\ M\ H;$
$f \in \{j.\ j \leq (n::nat)\} \rightarrow H;\ inj\text{-}on\ f\ \{j.\ j \leq n\};$
$s \in \{j.\ j \leq n\} \rightarrow carrier\ R;\ g \in \{j.\ j \leq (m::nat)\} \rightarrow H;$
$inj\text{-}on\ g\ \{j.\ j \leq m\};\ t \in \{j.\ j \leq m\} \rightarrow carrier\ R;$
$l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ m\ t\ g;$
$\forall\, j \in \{j.\ j \leq n\}.\ s\ j \neq \mathbf{0}_R;\ \forall\, k \in \{j.\ j \leq m\}.\ t\ k \neq \mathbf{0}_R]\!] \Longrightarrow$
$f\ `\ \{j.\ j \leq n\} \subseteq g\ `\ \{j.\ j \leq m\}$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression6*:$[\![free\text{-}generator\ R\ M\ H;$
$f \in \{j.\ j \leq (n::nat)\} \rightarrow H;\ inj\text{-}on\ f\ \{j.\ j \leq n\};$
$s \in \{j.\ j \leq n\} \rightarrow carrier\ R;$
$g \in \{j.\ j \leq (m::nat)\} \rightarrow H;\ inj\text{-}on\ g\ \{j.\ j \leq m\};$
$t \in \{j.\ j \leq m\} \rightarrow carrier\ R;$
$l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ m\ t\ g;$
$\forall\, j{\in}\{j.\ j \leq n\}.\ s\ j \neq \mathbf{0}_R;\ \forall\, k{\in}\ \{j.\ j \leq m\}.\ t\ k \neq \mathbf{0}_R]\!] \Longrightarrow$

f ‘$\{j.\ j \le n\} = g$ ‘ $\{j.\ j \le m\}$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression7-1*:⟦*free-generator R M H*;
$f \in \{j.\ j \le (n{::}nat)\} \to H$; *inj-on* f $\{j.\ j \le n\}$;
$s \in \{j.\ j \le n\} \to carrier\ R$;
$g \in \{j.\ j \le (m{::}nat)\} \to H$; *inj-on* g $\{j.\ j \le m\}$;
$t \in \{j.\ j \le m\} \to carrier\ R$;
l-comb $R\ M\ n\ s\ f$ = *l-comb* $R\ M\ m\ t\ g$;
$\forall j \in \{j.\ j \le n\}.\ s\ j \ne \mathbf{0}_R$; $\forall k{\in}\{j.\ j \le m\}.\ t\ k \ne \mathbf{0}_R$⟧ $\Longrightarrow n = m$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression7-2*:⟦*free-generator R M H*;
$f \in \{j.\ j \le (n{::}nat)\} \to H$; *inj-on* f $\{j.\ j \le n\}$;
$s \in \{j.\ j \le n\} \to carrier\ R$; $t \in \{j.\ j \le n\} \to carrier\ R$;
l-comb $R\ M\ n\ s\ f$ = *l-comb* $R\ M\ n\ t\ f$⟧ $\Longrightarrow (\forall l \in \{j.\ j \le n\}.\ s\ l = t\ l)$
⟨*proof*⟩

end

theory *Algebra8* **imports** *Algebra7* **begin**

5.4 nsum and Generators (continued)

lemma (**in** *Module*) *unique-expression-last*: ⟦*free-generator R M H*;
$f \in \{j.\ j \le Suc\ n\} \to H$; $s \in \{j.\ j \le Suc\ n\} \to carrier\ R$;
$g \in \{j.\ j \le Suc\ n\} \to H$; $t \in \{j.\ j \le Suc\ n\} \to carrier\ R$;
l-comb $R\ M\ (Suc\ n)\ s\ f$ = *l-comb* $R\ M\ (Suc\ n)\ t\ g$;
inj-on f $\{j.\ j \le Suc\ n\}$; *inj-on* g $\{j.\ j \le Suc\ n\}$;
$f\ (Suc\ n) = g\ (Suc\ n)$⟧ $\Longrightarrow s\ (Suc\ n) = t\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Module*) *unique-exprTr7p1*: ⟦*free-generator R M H*;
$\forall f\ s\ g\ t\ m.$
$f \in \{j.\ j \le n\} \to H \land$ *inj-on* f $\{j.\ j \le n\} \land s \in \{j.\ j \le n\} \to carrier\ R \land$
$g \in \{j.\ j \le m\} \to H \land$ *inj-on* g $\{j.\ j \le m\} \land t \in \{j.\ j \le m\} \to carrier\ R \land$
l-comb $R\ M\ n\ s\ f$ = *l-comb* $R\ M\ m\ t\ g \land$
$(\forall j{\le}n.\ s\ j \ne \mathbf{0}_R) \land (\forall k{\le}m.\ t\ k \ne \mathbf{0}_R) \longrightarrow$
$n = m \land$
$(\exists h.\ h \in \{j.\ j \le n\} \to \{j.\ j \le n\} \land$
$(\forall l{\le}n.\ cmp\ f\ h\ l = g\ l \land cmp\ s\ h\ l = t\ l))$;
$f \in \{j.\ j \le Suc\ n\} \to H$; $s \in \{j.\ j \le Suc\ n\} \to carrier\ R$;
$g \in \{j.\ j \le Suc\ n\} \to H$; $t \in \{j.\ j \le Suc\ n\} \to carrier\ R$;
l-comb $R\ M\ (Suc\ n)\ s\ f$ = *l-comb* $R\ M\ (Suc\ n)\ t\ g$; $\forall j{\le}Suc\ n.\ s\ j \ne \mathbf{0}_R$;
$\forall k{\le}Suc\ n.\ t\ k \ne \mathbf{0}_R$; *inj-on* f $\{j.\ j \le Suc\ n\}$; *inj-on* g $\{j.\ j \le Suc\ n\}$;
$f\ (Suc\ n) = g\ (Suc\ n)$⟧ $\Longrightarrow \exists h.\ h \in \{j.\ j \le Suc\ n\} \to \{j.\ j \le Suc\ n\} \land$
$(\forall l{\le}Suc\ n.\ cmp\ f\ h\ l = g\ l \land cmp\ s\ h\ l = t\ l)$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression7*:*free-generator R M H* $\Longrightarrow$
$\forall f\ s\ g\ t\ m.\ f \in \{j.\ j \le (n{::}nat)\} \rightarrow H \wedge inj\text{-}on\ f\ \{j.\ j \le n\} \wedge$
$s \in \{j.\ j \le n\} \rightarrow carrier\ R \wedge$
$g \in \{j.\ j \le (m{::}nat)\} \rightarrow H \wedge inj\text{-}on\ g\ \{j.\ j \le m\} \wedge$
$t \in \{j.\ j \le m\} \rightarrow carrier\ R \wedge l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ m\ t\ g \wedge$
$(\forall j \in \{j.\ j \le n\}.\ s\ j \ne \mathbf{0}_R) \wedge (\forall k \in \{j.\ j \le m\}.\ t\ k \ne \mathbf{0}_R\ ) \longrightarrow n = m \wedge$
$(\exists h.\ h \in \{j.\ j \le n\} \rightarrow \{j.\ j \le n\} \wedge (\forall l \in \{j.\ j \le n\}.\ (cmp\ f\ h)\ l = g\ l$
$\wedge\ (cmp\ s\ h)\ l = t\ l))$
⟨*proof*⟩

lemma (**in** *Module*) *gen-mHom-eq*:$[\![R\ module\ N;\ generator\ R\ M\ H;\ f \in mHom\ R$
$M\ N;$
$g \in mHom\ R\ M\ N;\ \forall h{\in}H.\ f\ h = g\ h\ ]\!] \Longrightarrow f = g$
⟨*proof*⟩

5.5 Existence of homomorphism

definition
fgs :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*, *'a set*] $\Rightarrow$
'a set **where**

fgs R M A = *linear-span R M* (*carrier R*) *A*

definition
fsp :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*,
(*'b*, *'r*, *'m2*) *Module-scheme*, *'a* $\Rightarrow$ *'b*, *'a set*, *'a set*, *'a* $\Rightarrow$ *'b*] $\Rightarrow$ *bool* **where**
$fsp\ R\ M\ N\ f\ H\ A\ g \longleftrightarrow g \in mHom\ R\ (mdl\ M\ (fgs\ R\ M\ A))\ N \wedge (\forall z{\in}A.\ f\ z =$
$g\ z) \wedge A \subseteq H$

definition
fsps :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*,
(*'b*, *'r*, *'m2*) *Module-scheme*, *'a* $\Rightarrow$ *'b*, *'a set*] $\Rightarrow$
((*'a set*) * (*'a* $\Rightarrow$ *'b*)) *set* **where**
$fsps\ R\ M\ N\ f\ H = \{Z.\ \exists A\ g.\ Z = (A,\ g) \wedge fsp\ R\ M\ N\ f\ H\ A\ g\}$

definition
od-fm-fun :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*,
(*'b*, *'r*, *'m2*) *Module-scheme*, *'a* $\Rightarrow$ *'b*, *'a set*] $\Rightarrow$
((*'a set*) * (*'a* $\Rightarrow$ *'b*)) *Order* **where**
od-fm-fun R M N f H = (|*carrier* = *fsps R M N f H*,
$rel = \{Y.\ Y \in (fsps\ R\ M\ N\ f\ H) \times (fsps\ R\ M\ N\ f\ H) \wedge$
$(fst\ (fst\ Y)) \subseteq (fst\ (snd\ Y))\}$ |)

lemma (**in** *Module*) *od-fm-fun-carrier*:*carrier* (*od-fm-fun R M N f H*) =
fsps R M N f H

⟨proof⟩

lemma (**in** *Module*) *fgs-submodule*: $a \subseteq carrier\ M \Longrightarrow$
$submodule\ R\ M\ (fgs\ R\ M\ a)$
⟨proof⟩

lemma (**in** *Module*) *fgs-sub-carrier*: $a \subseteq carrier\ M \Longrightarrow (fgs\ R\ M\ a) \subseteq carrier\ M$
⟨proof⟩

lemma (**in** *Module*) *elem-fgs*: $[\![a \subseteq carrier\ M;\ x \in a]\!] \Longrightarrow x \in fgs\ R\ M\ a$
⟨proof⟩

lemma (**in** *Module*) *fst-chain-subset*: $[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ C \subseteq fsps\ R\ M\ N\ f\ H;\ (a,\ b) \in C]\!] \Longrightarrow a \subseteq carrier\ M$
⟨proof⟩

lemma (**in** *Module*) *empty-fsp*: $[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N]\!] \Longrightarrow (\{\},\ (\lambda x{\in}\{\mathbf{0}_M\}.\ \mathbf{0}_N)) \in fsps\ R\ M\ N\ f\ H$
⟨proof⟩

lemma (**in** *Module*) *mem-fgs-l-comb*: $[\![K \neq \{\};\ K \subseteq carrier\ M;\ x \in fgs\ R\ M\ K]\!]$
$\Longrightarrow$
$\exists (n::nat).$
$\exists f \in \{j.\ j \leq (n::nat)\} \to K.\ \exists s \in \{j.\ j \leq n\} \to carrier\ R.$
$x = l\text{-}comb\ R\ M\ n\ s\ f$
⟨proof⟩

lemma *PairE-lemma*: $\exists x\ y.\ p = (x,\ y)$ ⟨proof⟩

lemma (**in** *Module*) *fsps-chain-boundTr1*: $[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ C \subseteq fsps\ R\ M\ N\ f\ H;$
$\forall a{\in}C.\ \forall b{\in}C.\ fst\ a \subseteq fst\ b \vee fst\ b \subseteq fst\ a;\ \forall a\ b.\ (a,\ b) \in C \longrightarrow$
$(a,\ b) \in fsps\ R\ M\ N\ f\ H;\ \exists x.\ (\exists b.\ (x,\ b) \in C) \wedge x \neq \{\}]\!] \Longrightarrow$
$fa \in \{j.\ j \leq (n::nat)\} \to \bigcup\{a.\ \exists b.\ (a,\ b) \in C\}$
$\longrightarrow (\exists (c,\ d) \in C.\ fa\ \text{`}\ \{j.\ j \leq n\} \subseteq c)$
⟨proof⟩

lemma (**in** *Module*) *fsps-chain-boundTr1-1*: $[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ C \subseteq fsps\ R\ M\ N\ f\ H;$
$\forall a{\in}C.\ \forall b{\in}C.\ fst\ a \subseteq fst\ b \vee fst\ b \subseteq fst\ a;$
$\exists x.\ (\exists b.\ (x,\ b) \in C) \wedge x \neq \{\};$
$fa \in \{j.\ j \leq (n::nat)\} \to \bigcup\{a.\ \exists b.\ (a,\ b) \in C\}]\!] \Longrightarrow$
$\exists (c,\ d) \in C.\ fa\ \text{`}\ \{j.\ j \leq n\} \subseteq c$
⟨proof⟩

lemma (**in** *Module*) *fsps-chain-boundTr1-2*: $[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ C \subseteq fsps\ R\ M\ N\ f\ H;$
$\forall a{\in}C.\ \forall b{\in}C.\ fst\ a \subseteq fst\ b \vee fst\ b \subseteq fst\ a;$
$\exists x.\ (\exists b.\ (x,\ b) \in C) \wedge x \neq \{\};$

$fa \in \{j.\ j \leq (n::nat)\} \to \bigcup\{a.\ \exists b.\ (a,\ b) \in C\}\rrbracket \Longrightarrow$
$\exists P \in C.\ fa\ `\ \{j.\ j \leq n\} \subseteq fst\ P$
⟨proof⟩

lemma (**in** *Module*) *eSum-in-SubmoduleTr*:$\llbracket H \subseteq carrier\ M;\ K \subseteq H\rrbracket \Longrightarrow$
$f \in \{j.\ j \leq (n::nat)\} \to K \wedge s \in \{j.\ j \leq n\} \to carrier\ R \longrightarrow$
$l\text{-}comb\ R\ (mdl\ M\ (fgs\ R\ M\ K))\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ f$
⟨proof⟩

lemma (**in** *Module*) *eSum-in-Submodule*:$\llbracket H \subseteq carrier\ M;\ K \subseteq H;$
$f \in \{j.\ j \leq (n::nat)\} \to K;\ s \in \{j.\ j \leq n\} \to carrier\ R\rrbracket \Longrightarrow$
$l\text{-}comb\ R\ (mdl\ M\ (fgs\ R\ M\ K))\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ f$
⟨proof⟩

lemma (**in** *Module*) *fgs-generator*:$H \subseteq carrier\ M \Longrightarrow$
$generator\ R\ (mdl\ M\ (fgs\ R\ M\ H))\ H$
⟨proof⟩

lemma (**in** *Module*) *fgs-mono*:$\llbracket free\text{-}generator\ R\ M\ H;\ J \subseteq K;\ K \subseteq H\rrbracket$
$\Longrightarrow fgs\ R\ M\ J \subseteq fgs\ R\ M\ K$
⟨proof⟩

lemma (**in** *Module*) *empty-fgs*:$fgs\ R\ M\ \{\} = \{\mathbf{0}\}$
⟨proof⟩

lemma (**in** *Module*) *mem-fsps-snd-mHom*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ (a,\ b) \in fsps\ R\ M\ N\ f\ H\rrbracket \Longrightarrow$
$b \in mHom\ R\ (mdl\ M\ (fgs\ R\ M\ a))\ N$
⟨proof⟩

lemma (**in** *Module*) *mem-fsps-fst-sub*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ (a,\ b) \in fsps\ R\ M\ N\ f\ H\rrbracket \Longrightarrow a \subseteq H$
⟨proof⟩

lemma (**in** *Module*) *mem-fsps-fst-sub1*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ (a,\ b) \in fsps\ R\ M\ N\ f\ H\rrbracket \Longrightarrow a \subseteq carrier\ M$
⟨proof⟩

lemma (**in** *Module*) *Order-od-fm-fun*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N\rrbracket \Longrightarrow Order\ (od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H)$
⟨proof⟩

lemma (**in** *Module*) *fsps-chain-boundTr2-1*:$\llbracket R\ module\ N;$
$free\text{-}generator\ R\ M\ H;\ f \in H \to carrier\ N;\ C \subseteq fsps\ R\ M\ N\ f\ H;$
$(a,\ b) \in C;\ (aa,\ ba) \in C;\ x \in fgs\ R\ M\ a;\ x \in fgs\ R\ M\ aa;\ a \subseteq aa\rrbracket$
$\Longrightarrow b\ x = ba\ x$
⟨proof⟩

lemma (**in** *Module*) *fsps-chain-boundTr2-2*:⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*; *C* ⊆ *fsps R M N f H*;
∀ *a*∈*C*. ∀ *b*∈*C*. *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*; *C* ≠ {}; (*a*, *b*) ∈ *C*;
x ∈ *fgs R M a*; (*a1*, *b1*) ∈ *C*; *x* ∈ *fgs R M a1*⟧ ⟹ *b x* = *b1 x*
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-boundTr2*:⋀*x*. ⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*; *C* ⊆ *fsps R M N f H*;
∀ *a*∈*C*. ∀ *b*∈*C*. *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*;
x ∈ (*fgs R M* (⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*})); *C* ≠ {}⟧ ⟹
(*THE y*. *y* ∈ *carrier N* ∧ (∃ *a b*. (*a*, *b*) ∈ *C* ∧ *x* ∈ *fgs R M a* ∧ *y* = *b x*)) ∈ (*carrier N*) ∧
(∃ *a1 b1*. (*a1*, *b1*) ∈ *C* ∧ *x* ∈ *fgs R M a1* ∧
(*THE y*. *y* ∈ *carrier N* ∧ (∃ *a b*. (*a*, *b*) ∈ *C* ∧ *x* ∈ *fgs R M a* ∧ *y* = *b x*)) = *b1 x*)
⟨*proof*⟩

lemma (**in** *Module*) *Un-C-submodule*:⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*; *C* ⊆ *fsps R M N f H*; *C* ≠ {};
∀ *a*∈*C*. ∀ *b*∈*C*. *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*⟧ ⟹
submodule R M (*fgs R M* (⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*}))
⟨*proof*⟩

lemma (**in** *Module*) *Un-C-fgs-sub*:⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*; *C* ⊆ *fsps R M N f H*; *C* ≠ {};
∀ *a*∈*C*. ∀ *b*∈*C*. *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*⟧ ⟹
⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*} ⊆ *fgs R M* (⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*})
⟨*proof*⟩

lemma (**in** *Module*) *Chain-3-supset*:⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*; *C* ⊆ *fsps R M N f H*; *C* ≠ {};
∀ *a*∈*C*. ∀ *b*∈*C*. *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*; (*a1*, *b1*) ∈ *C*; (*a2*, *b2*) ∈ *C*;
(*a3*, *b3*) ∈ *C*⟧ ⟹ ∃(*g*, *h*)∈*C*. *a1* ⊆ *g* ∧ *a2* ⊆ *g* ∧ *a3* ⊆ *g*
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-bound1*:⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*; *C* ⊆ *fsps R M N f H*;
∀ *a*∈*C*. ∀ *b*∈*C*. *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*; *C* ≠ {}⟧ ⟹
(λ*x*∈(*fgs R M* (⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*})). (*THE y*. *y* ∈ *carrier N* ∧
(∃ *a b*. (*a*, *b*) ∈ *C* ∧ *x* ∈ *fgs R M a* ∧ *y* = *b x*))) ∈
mHom R (*mdl M* (*fgs R M* (⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*}))) *N*
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-bound2*:⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*; *C* ⊆ *fsps R M N f H*; *C* ≠ {};
∀ *a*∈*C*. ∀ *b*∈*C*. *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*⟧ ⟹
∀ *y*∈(⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*}). (λ*x*∈(*fgs R M* (⋃{*a*. ∃ *b*. (*a*, *b*) ∈ *C*})).
(*THE y*. *y* ∈ *carrier N* ∧ (∃ *a b*. (*a*, *b*) ∈ *C* ∧ *x* ∈ *fgs R M a* ∧ *y* = *b x*))) *y* =

$f\ y$
⟨proof⟩

lemma (**in** *Module*) *od-fm-fun-Chain*:$[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ Algebra2.Chain\ (od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H)\ C;\ C \neq \{\}]\!] \Longrightarrow$

$\forall a{\in}C.\ \forall b{\in}C.\ fst\ a \subseteq fst\ b \vee fst\ b \subseteq fst\ a$
⟨proof⟩

lemma (**in** *Module*) *od-fm-fun-inPr0*:$[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ Algebra2.Chain\ (od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H)\ C;\ C \neq \{\};$
$\exists b.\ (y,\ b) \in C;\ z \in y]\!] \Longrightarrow z \in fgs\ R\ M\ (\bigcup\{a.\ \exists b.\ (a,\ b) \in C\})$
⟨proof⟩

lemma (**in** *Module*) *od-fm-fun-indPr1*: $[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ Algebra2.Chain\ (od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H)\ C;\ C \neq \{\}]\!] \Longrightarrow$
$(\bigcup\{a.\ \exists b.\ (a,\ b) \in C\},$
$\lambda x \in fgs\ R\ M\ (\bigcup\{a.\ \exists b.\ (a,\ b) \in C\}).\ THE\ y.\ y \in carrier\ N \wedge$
$(\exists a\ b.\ (a,\ b) \in C \wedge x \in fgs\ R\ M\ a \wedge y = b\ x)) \in fsps\ R\ M\ N\ f\ H$
⟨proof⟩

lemma (**in** *Module*) *od-fm-fun-indPr2*: $[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ Algebra2.Chain\ (od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H)\ C;\ C \neq \{\}]\!] \Longrightarrow$
$ub_{od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H}\ C\ (\bigcup\{a.\ \exists b.\ (a,\ b) \in C\},$
$\lambda x \in fgs\ R\ M\ (\bigcup\{a.\ \exists b.\ (a,\ b) \in C\}).\ THE\ y.\ y \in carrier\ N \wedge$
$(\exists a\ b.\ (a,\ b) \in C \wedge x \in fgs\ R\ M\ a \wedge y = b\ x))$
⟨proof⟩

lemma (**in** *Module*) *od-fm-fun-inductive*:$[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N]\!] \Longrightarrow inductive\text{-}set\ (od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H)$
⟨proof⟩

lemma (**in** *Module*) *sSum-eq*:$[\![R\ module\ N;\ free\text{-}generator\ R\ M\ H;\ H1 \subseteq H;$
$h \in H - H1]\!] \Longrightarrow (fgs\ R\ M\ H1) \mp (fgs\ R\ M\ \{h\}) = fgs\ R\ M\ (H1 \cup \{h\})$
⟨proof⟩

definition
monofun :: $[('r,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'r,\ 'm1)\ Module\text{-}scheme,$
$('b,\ 'r,\ 'm2)\ Module\text{-}scheme,\ 'a \Rightarrow 'b,\ 'a\ set,\ 'a] \Rightarrow ('a \Rightarrow 'b)$ **where**
$monofun\ R\ M\ N\ f\ H\ h = (\lambda x{\in}fgs\ R\ M\ \{h\}.$
$(THE\ y.\ (\exists r{\in}carrier\ R.\ x = r \cdot_{s M} h \wedge\ y = r \cdot_{s N} (f\ h))))$

lemma (**in** *Module*) *fgs-single-span*:$[\![h \in carrier\ M;\ x \in (fgs\ R\ M\ \{h\})]\!] \Longrightarrow$
$\exists a{\in}carrier\ R.\ x = a \cdot_s h$
⟨proof⟩

lemma (**in** *Module*) *monofun-mHomTr*:$[\![h \in H;\ free\text{-}generator\ R\ M\ H;$
$a \in carrier\ R;\ r \in carrier\ R;\ a \cdot_s h = r \cdot_s h]\!] \Longrightarrow a = r$
⟨proof⟩

lemma (**in** *Module*) *monofun-mhomTr1*:$[\![R$ *module* N; $h \in H$; *free-generator* $R\ M$ H;
$f \in H \to$ *carrier* N; $a \in$ *carrier* $R]\!] \Longrightarrow$
monofun $R\ M\ N\ f\ H\ h\ (a \cdot_s h) = a \cdot_{s N} (f\ h)$
⟨*proof*⟩

lemma (**in** *Module*) *monofun-mem*:$[\![R$ *module* N; $h \in H$; *free-generator* $R\ M\ H$;
$x \in$ *fgs* $R\ M\ \{h\}$; $f \in H \to$ *carrier* $N]\!] \Longrightarrow$
monofun $R\ M\ N\ f\ H\ h\ x \in$ *carrier* N
⟨*proof*⟩

lemma (**in** *Module*) *monofun-eq-f*:$[\![R$ *module* N; $h \in H$; *free-generator* $R\ M\ H$;
$f \in H \to$ *carrier* $N]\!] \Longrightarrow$ *monofun* $R\ M\ N\ f\ H\ h\ h = f\ h$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-unique*:$[\![R$ *module* N; *free-generator* $R\ M\ H$; $H1 \subseteq H$;
$h \in H - H1$; $x1 \in$(*fgs* $R\ M\ H1$); $x2 \in$ (*fgs* $R\ M\ H1$);
$y1 \in$ (*fgs* $R\ M\ \{h\}$); $y2 \in$ (*fgs* $R\ M\ \{h\}$); $x1 \pm y1 = x2 \pm y2]\!] \Longrightarrow$
$x1 = x2 \wedge y1 = y2$
⟨*proof*⟩

lemma (**in** *Module*) *ex-extensionTr*:$[\![R$ *module* N; *free-generator* $R\ M\ H$;
$f \in H \to$ *carrier* N; $H1 \subseteq H$; $h \in H$; $h \notin H1$;
$g \in$ *mHom* R (*mdl* M (*fgs* $R\ M\ H1$)) N;
$x \in$ *fgs* $R\ M\ H1 \mp$ (*fgs* $R\ M\ \{h\}$)$]\!] \Longrightarrow$
$\exists k1\in$ *fgs* $R\ M\ H1$. $\exists k2\in$*fgs* $R\ M\ \{h\}$. $x = k1 \pm k2 \wedge$
(*THE* y. $\exists h1\in$*fgs* $R\ M\ H1$. $\exists h2\in$*fgs* $R\ M\ \{h\}$. $x = h1 \pm h2 \wedge y = g\ h1 \pm_N$
(*monofun* $R\ M\ N\ f\ H\ h\ h2$)) $= g\ k1\ \pm_N$ (*monofun* $R\ M\ N\ f\ H\ h\ k2$)
⟨*proof*⟩

lemma (**in** *Module*) *monofun-add*:$[\![R$ *module* N; *free-generator* $R\ M\ H$;
$f \in H \to$ *carrier* N; $h \in H$; $x \in$ *fgs* $R\ M\ \{h\}$; $y \in$ *fgs* $R\ M\ \{h\}]\!] \Longrightarrow$
monofun $R\ M\ N\ f\ H\ h\ (x \pm y) =$
monofun $R\ M\ N\ f\ H\ h\ x \pm_N$ (*monofun* $R\ M\ N\ f\ H\ h\ y$)
⟨*proof*⟩

lemma (**in** *Module*) *monofun-sprod*:$[\![R$ *module* N; *free-generator* $R\ M\ H$;
$f \in H \to$ *carrier* N; $h \in H$; $x \in$ *fgs* $R\ M\ \{h\}$; $a \in$ *carrier* $R]\!] \Longrightarrow$
monofun $R\ M\ N\ f\ H\ h\ (a \cdot_s x) = a \cdot_{s N}$ (*monofun* $R\ M\ N\ f\ H\ h\ x$)
⟨*proof*⟩

lemma (**in** *Module*) *monofun-0*:$[\![R$ *module* N; *free-generator* $R\ M\ H$;
$f \in H \to$ *carrier* N; $h \in H]\!] \Longrightarrow$ *monofun* $R\ M\ N\ f\ H\ h\ \mathbf{0} = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Module*) *ex-extension*:$[\![R$ *module* N; *free-generator* $R\ M\ H$;
$f \in H \to$ *carrier* N; $H1 \subseteq H$; $h \in H - H1$; $(H1, g) \in$ *fsps* $R\ M\ N\ f\ H]\!] \Longrightarrow$
$\exists k.\ ((H1 \cup \{h\}), k) \in$ *fsps* $R\ M\ N\ f\ H$

⟨*proof*⟩

lemma (**in** *Module*) *mHom-mHom*:⟦*R module N*; *g* ∈ *mHom R* (*mdl M* (*carrier M*)) *N*⟧
⟹ *g* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *exist-extension-mhom*:⟦*R module N*; *free-generator R M H*;
f ∈ *H* → *carrier N*⟧ ⟹ ∃ *g*∈*mHom R M N*. ∀ *x*∈*H*. *g x* = *f x*
⟨*proof*⟩

5.6 Nakayama lemma

definition
Lcg :: [(*'r*, *'m*) *Ring-scheme*, (*'a*, *'r*, *'m1*) *Module-scheme*, *nat*] ⇒ *bool* **where**
Lcg R M j ⟷ (∃ *H*. *finite-generator R M H* ∧ *j* = *card H*)

lemma (**in** *Module*) *NAKTr1*:*M fgover R* ⟹
∃ *H*. *finite-generator R M H* ∧ (*LEAST j*.
∃ *L*. *finite-generator R M L* ∧ *j* = *card L*) = *card H*
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr2*:⟦*Lcg R M j*; *k* < (*LEAST j*. *Lcg R M j*)⟧ ⟹
¬ *Lcg R M k*
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr3*:⟦*M fgover R*; *H* ⊆ *carrier M*; *finite H*;
card H < (*LEAST j*. ∃ *L*. *finite-generator R M L* ∧ *j* = *card L*)⟧ ⟹
¬ *finite-generator R M H*
⟨*proof*⟩

lemma (**in** *Module*) *finite-gen-over-ideal*:⟦*ideal R A*; *h* ∈ {*j*. *j* ≤ (*n*::*nat*)} →
carrier M; *generator R M* (*h* ‘ {*j*. *j* ≤ *n*}); *A* $\odot_R$ *M* = *carrier M*;
m ∈ *carrier M* ⟧ ⟹ ∃ *s*∈{*j*. *j* ≤ *n*} → *A*. *m* = *l-comb R M n s h*
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr4*:⟦*ideal R A*; *h* ∈ {*j*. *j* ≤ (*k*::*nat*)} → *carrier M*;
0 < *k*; *h* ‘ {*j*. *j* ≤ *k*} ⊆ *carrier M*; *s* ∈ {*j*. *j* ≤ *k*} → *A*;
h k = Σ_e *M* (λ*j*. *s j* $\cdot_s$ (*h j*)) (*k* − *Suc 0*) ± (*s k* $\cdot_s$ (*h k*))⟧ ⟹
(1_{rR} $\pm_R$ ($-_{aR}$ (*s k*))) $\cdot_s$ (*h k*) = Σ_e *M* (λ*j*. *s j* $\cdot_s$ (*h j*)) (*k* − *Suc 0*)
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr5*:⟦¬ *zeroring R*; *ideal R A*; *A* ⊆ *J-rad R*;
A $\odot_R$ *M* = *carrier M*; *finite-generator R M H*; *card H* = *Suc k*; *0* < *k*⟧ ⟹
∃ *h* ∈ {*j*. *j* ≤ *k*} → *carrier M*. *H* = *h* ‘ {*j*. *j* ≤ *k*} ∧
h k ∈ *linear-span R M A* (*h* ‘ {*j*. *j* ≤ (*k* − *Suc 0*)})
⟨*proof*⟩

lemma (**in** *Module*) *NAK*:⟦¬ *zeroring R*; *M fgover R*; *ideal R A*; $A \subseteq$ *J-rad R*; $A \odot_R M$ = *carrier M* ⟧ ⟹ *carrier M* = {**0**}
⟨*proof*⟩

lemma (**in** *Module*) *fg-qmodule*:⟦ *M fgover R*; *submodule R M N*⟧ ⟹ $(M /_m N)$ *fgover R*
⟨*proof*⟩

lemma (**in** *Module*) *NAK1*:⟦¬ *zeroring R*; *M fgover R*; *submodule R M N*; *ideal R A*; $A \subseteq$ *J-rad R*; *carrier M* = $A \odot_R M \mp N$ ⟧ ⟹ *carrier M* = *N*
⟨*proof*⟩

5.7 Direct sum and direct products of modules

definition
prodM-sprod :: [('r, 'm) *Ring-scheme*, 'i *set*,
'i ⇒ ('a, 'r, 'm1) *Module-scheme*] ⇒ 'r ⇒ ('i ⇒ 'a) ⇒ ('i ⇒ 'a) **where**
prodM-sprod R I A = (λa∈*carrier R*. λg∈*carr-prodag I A*.
(λj∈I. $a \cdot_{s(A\ j)}$ (g j)))

definition
prodM :: [('r, 'm) *Ring-scheme*, 'i *set*, 'i ⇒ ('a, 'r, 'm1) *Module-scheme*] ⇒
(| *carrier*:: ('i ⇒ 'a) *set*,
pop::['i ⇒ 'a, 'i ⇒ 'a] ⇒ ('i ⇒ 'a),
mop:: ('i ⇒ 'a) ⇒ ('i ⇒ 'a), *zero*::('i ⇒ 'a),
sprod :: ['r, 'i ⇒ 'a] ⇒ ('i ⇒ 'a) |) **where**
prodM R I A = (|*carrier* = *carr-prodag I A*,
pop = *prod-pOp I A*, *mop* = *prod-mOp I A*,
zero = *prod-zero I A*, *sprod* = *prodM-sprod R I A* |)

definition
mProject :: [('r, 'm) *Ring-scheme*, 'i *set*,
'i ⇒ ('a, 'r, 'more) *Module-scheme*, 'i] ⇒ ('i ⇒ 'a) ⇒ 'a **where**
mProject R I A j = (λf∈*carr-prodag I A*. *f j*)

abbreviation
PRODMODULES ((*3m*Π- - -) [*72*,*72*,*73*]*72*) **where**
$m\Pi_{R\ I}$ *A* == *prodM R I A*

lemma (**in** *Ring*) *prodM-carr*:⟦∀ i∈I. (*R module* (*M i*))⟧ ⟹
carrier (*prodM R I M*) = *carr-prodag I M*
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-mem-eq*:⟦∀ i∈I. (*R module* (*M i*));
m1 ∈ *carrier* (*prodM R I M*); *m2* ∈ *carrier* (*prodM R I M*);
∀ i∈I. *m1 i* = *m2 i* ⟧ ⟹ *m1* = *m2*
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-sprod-mem*:$[\![\forall\, i{\in}I.\ (R\ module\ (M\ i));\ a \in carrier\ R;$
$m \in carr\text{-}prodag\ I\ M]\!] \Longrightarrow prodM\text{-}sprod\ R\ I\ M\ a\ m \in carr\text{-}prodag\ I\ M$
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-sprod-val*:$[\![\forall\, i{\in}I.\ (R\ module\ (M\ i));\ a \in carrier\ R;$
$m \in carr\text{-}prodag\ I\ M;\ j \in I]\!] \Longrightarrow (prodM\text{-}sprod\ R\ I\ M\ a\ m)\ j = a \cdot_{s(M\ j)} (m\ j)$
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-Module*:$\forall\, i{\in}I.\ (R\ module\ (M\ i)) \Longrightarrow$
$R\ module\ (prodM\ R\ I\ M)$
⟨*proof*⟩

definition
dsumM :: [('*r*, '*m*) *Ring-scheme*, '*i set*, '*i* ⇒ ('*a*, '*r*, '*more*) *Module-scheme*]
⇒ (| *carrier*:: ('*i* ⇒ '*a*) *set*,
pop::['*i* ⇒ '*a*, '*i* ⇒ '*a*] ⇒ ('*i* ⇒ '*a*),
mop:: ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*),
zero::('*i* ⇒ '*a*),
sprod :: ['*r*, '*i* ⇒ '*a*] ⇒ ('*i* ⇒ '*a*) |) **where**

dsumM R I A = (| *carrier* = *carr-dsumag I A*,
pop = *prod-pOp I A*, *mop* = *prod-mOp I A*,
zero = *prod-zero I A*, *sprod* = *prodM-sprod R I A*|)

abbreviation
DSUMMOD $((3\_\Sigma_{d\_}\ \text{-})\ [72,72,73]72)$ **where**
$_R\Sigma_{d\,I}\ A == dsumM\ R\ I\ A$

lemma (**in** *Ring*) *dsumM-carr*:$carrier\ (dsumM\ R\ I\ M) = carr\text{-}dsumag\ I\ M$
⟨*proof*⟩

lemma (**in** *Ring*) *dsum-sprod-mem*:$[\![\forall\, i{\in}I.\ R\ module\ M\ i;\ a \in carrier\ R;$
$b \in carr\text{-}dsumag\ I\ M]\!] \Longrightarrow prodM\text{-}sprod\ R\ I\ M\ a\ b \in carr\text{-}dsumag\ I\ M$
⟨*proof*⟩

lemma (**in** *Ring*) *carr-dsum-prod*:$carr\text{-}dsumag\ I\ M \subseteq carr\text{-}prodag\ I\ M$
⟨*proof*⟩

lemma (**in** *Ring*) *carr-dsum-prod1*:
$\forall\, x.\ x \in carr\text{-}dsumag\ I\ M \longrightarrow x \in carr\text{-}prodag\ I\ M$
⟨*proof*⟩

lemma (**in** *Ring*) *carr-dsumM-mem-eq*:$[\![\forall\, i{\in}I.\ R\ module\ M\ i;\ x \in carr\text{-}dsumag\ I$
$M;$
$y \in carr\text{-}dsumag\ I\ M;\ \forall\, j{\in}I.\ x\ j = y\ j]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Ring*) *dsumM-Module*:$\forall\, i{\in}I.\ R\ module\ (M\ i) \Longrightarrow R\ module\ (_R\Sigma_{d\,I}\ M)$
⟨*proof*⟩

definition
ringModule :: ('*r*, '*b*) *Ring-scheme* ⇒ ('*r*, '*r*) *Module*
((*M*$_{\_}$) [*998*]*999*) **where**
M_R = (|*carrier* = *carrier R*, *pop* = *pop R*, *mop* = *mop R*,
zero = *zero R*, *sprod* = *tp R*|)

lemma (**in** *Ring*) *ringModule-Module*:*R module* M_R
⟨*proof*⟩

definition
dsumMhom :: ['*i set*, '*i* ⇒ ('*a*, '*r*, '*m*) *Module-scheme*,
'*i* ⇒ ('*b*, '*r*, '*m1*) *Module-scheme*, '*i* ⇒ ('*a* ⇒ '*b*)] ⇒ ('*i* ⇒ '*a*) ⇒
('*i* ⇒ '*b*) **where**

dsumMhom I A B S = (λ*f*∈*carr-dsumag I A*. (λ*k*∈*I*. (*S k*) (*f k*)))

lemma (**in** *Ring*) *dsumMhom-mem*:⟦∀ *i*∈*I*. *R module M i*; ∀ *i*∈*I*. *R module N i*;
∀ *i*∈*I*. *S i* ∈ *mHom R* (*M i*) (*N i*); *x* ∈ *carr-dsumag I M*⟧
⟹ *dsumMhom I M N S x* ∈ *carr-dsumag I N*
⟨*proof*⟩

lemma (**in** *Ring*) *dsumMhom-mHom*:⟦∀ *i*∈*I*. (*R module* (*M i*));
∀ *i*∈*I*. (*R module* (*N i*)); ∀ *i*∈*I*. ((*S i*) ∈ *mHom R* (*M i*) (*N i*))⟧ ⟹
dsumMhom I M N S ∈ *mHom R* (*dsumM R I M*) (*dsumM R I N*)
⟨*proof*⟩

end

theory *Algebra9* **imports** *Algebra8* **begin**

5.8 Exact sequence

definition
Zm :: [('*r*, '*m*) *Ring-scheme*, '*a*] ⇒ ('*a*, '*r*) *Module* **where**
Zm R e = (| *carrier* = {*e*}, *pop* = λ*x*∈{*e*}. λ*y*∈{*e*}. *e*, *mop* =
λ*x*∈{*e*}. *e*, *zero* = *e*, *sprod* = λ*r*∈*carrier R*. λ*x*∈{*e*}. *e*|)

lemma (**in** *Ring*) *Zm-Module*:*R module* (*Zm R e*)
⟨*proof*⟩

lemma (**in** *Ring*) *Zm-carrier*:*carrier* (*Zm R e*) = {*e*}
⟨*proof*⟩

lemma (**in** *Ring*) *Zm-to-M-0*:⟦*R module M*; *f* ∈ *mHom R* (*Zm R e*) *M*⟧ ⟹

$f\ e = \mathbf{0}_M$
⟨proof⟩

lemma (**in** *Ring*) *Z-to-M*:⟦*R module M*; $f \in$ *mHom R* (*Zm R e*) *M*;
$g \in$ *mHom R* (*Zm R e*) *M* ⟧ $\Longrightarrow f = g$
⟨proof⟩

lemma (**in** *Ring*) *mzeromap-mHom*:⟦*R module M*; *R module N*⟧ $\Longrightarrow$
mzeromap M N $\in$ *mHom R M N*
⟨proof⟩

lemma (**in** *Ring*) *HOM-carrier*:*carrier* (HOM_R *M N*) = *mHom R M N*
⟨proof⟩

lemma (**in** *Ring*) *mHom-Z-M*:*R module M* $\Longrightarrow$
mHom R (*Zm R e*) *M* = {*mzeromap* (*Zm R e*) *M*}
⟨proof⟩

lemma (**in** *Module*) *Modules-single-carrier-isom*:⟦*R module N*; *carrier M* = {**0**};
carrier N = {$\mathbf{0}_N$}⟧ $\Longrightarrow M \cong_R N$
⟨proof⟩

lemma (**in** *Ring*) *Zm-isom*:(*Zm R* (*e*::′*a*)) $\cong_R$ (*Zm R* (*u*::′*b*))
⟨proof⟩

lemma (**in** *Ring*) *HOM-Z-M-0*:*R module M* $\Longrightarrow$ HOM_R (*Zm R e*) *M* $\cong_R$ (*Zm R*
e)
⟨proof⟩

lemma (**in** *Ring*) *M-to-Z*:⟦*R module M*; $f \in$ *mHom R M* (*Zm R e*);
$g \in$ *mHom R M* (*Zm R e*)⟧ $\Longrightarrow f = g$
⟨proof⟩

lemma (**in** *Ring*) *mHom-to-zero*:*R module M* $\Longrightarrow$ *mHom R M* (*Zm R e*) =
{*mzeromap M* (*Zm R e*)}
⟨proof⟩

lemma (**in** *Ring*) *carrier-HOM-M-Z*:*R module M* $\Longrightarrow$
carrier (HOM_R *M* (*Zm R e*)) = {*mzeromap M* (*Zm R e*)}
⟨proof⟩

lemma (**in** *Ring*) *HOM-M-Z-0*:*R module M* $\Longrightarrow$ HOM_R *M* (*Zm R e*) $\cong_R$ (*Zm R*
e)
⟨proof⟩

lemma (**in** *Ring*) *M-to-Z-0*:⟦*R module M*; $f \in$ *mHom R M* (*Zm R e*)⟧ $\Longrightarrow$
$ker_{M,(Zm\ R\ e)}\ f$ = *carrier M*
⟨proof⟩

definition
exact3 :: $[('r, 'm)$ *Ring-scheme*, $('a, 'r, 'm1)$ *Module-scheme*, $'a \Rightarrow 'b$,
$('b, 'r, 'm1)$ *Module-scheme*, $'b \Rightarrow 'c$, $('c, 'r, 'm1)$ *Module-scheme*$] \Rightarrow$ *bool*
where
exact3 R L0 h0 L1 h1 L2 $==$ *h0* ‘ (*carrier L0*) $=$ $ker_{(L1),(L2)}$ *h1*

definition
exact4 :: $[('r, 'm)$ *Ring-scheme*, $('a0, 'r, 'm1)$ *Module-scheme*, $'a0 \Rightarrow 'a1$,
$('a1, 'r, 'm1)$ *Module-scheme*, $'a1 \Rightarrow 'a2$, $('a2, 'r, 'm1)$ *Module-scheme*,
$'a2 \Rightarrow 'a3$, $('a3, 'r, 'm1)$ *Module-scheme*$] \Rightarrow$ *bool* **where**
exact4 R L0 h0 L1 h1 L2 h2 L3 $\longleftrightarrow$ *h0* ‘ (*carrier L0*) $=$ $ker_{(L1),(L2)}$ *h1* $\wedge$
h1 ‘ (*carrier L1*) $=$ $ker_{(L2),(L3)}$ *h2*

definition
exact5 :: $[('r, 'm)$ *Ring-scheme*, $('a0, 'r, 'm1)$ *Module-scheme*, $'a0 \Rightarrow 'a1$,
$('a1, 'r, 'm1)$ *Module-scheme*, $'a1 \Rightarrow 'a2$, $('a2, 'r, 'm1)$ *Module-scheme*,
$'a2 \Rightarrow 'a3$, $('a3, 'r, 'm1)$ *Module-scheme*, $'a3 \Rightarrow 'a4$,
$('a4, 'r, 'm1)$ *Module-scheme*$] \Rightarrow$ *bool* **where**
exact5 R L0 h0 L1 h1 L2 h2 L3 h3 L4 $==$ *h0* ‘ (*carrier L0*) $=$ $ker_{(L1),(L2)}$ *h1*
$\wedge$
h1 ‘ (*carrier L1*) $=$ $ker_{(L2),(L3)}$ *h2* $\wedge$ *h2* ‘(*carrier L2*) $=$ $ker_{(L3),(L4)}$ *h3*

definition
exact8 :: $[('r, 'm)$ *Ring-scheme*, $('a0, 'r, 'm1)$ *Module-scheme*, $'a0 \Rightarrow 'a1$,
$('a1, 'r, 'm1)$ *Module-scheme*, $'a1 \Rightarrow 'a2$, $('a2, 'r, 'm1)$ *Module-scheme*,
$'a2 \Rightarrow 'a3$, $('a3, 'r, 'm1)$ *Module-scheme*, $'a3 \Rightarrow 'a4$,
$('a4, 'r, 'm1)$ *Module-scheme*, $'a4 \Rightarrow 'a5$, $('a5, 'r, 'm1)$ *Module-scheme*,
$'a5 \Rightarrow 'a6$, $('a6, 'r, 'm1)$ *Module-scheme*, $'a6 \Rightarrow 'a7$,
$('a7, 'r, 'm1)$ *Module-scheme*$] \Rightarrow$ *bool* **where**
exact8 R L0 h0 L1 h1 L2 h2 L3 h3 L4 h4 L5 h5 L6 h6 L7 $\longleftrightarrow$
h0 ‘ (*carrier L0*) $=$ $ker_{(L1),(L2)}$ *h1* $\wedge$ *h1* ‘ (*carrier L1*) $=$ $ker_{(L2),(L3)}$ *h2* $\wedge$
h2 ‘ (*carrier L2*) $=$ $ker_{(L3),(L4)}$ *h3* $\wedge$ *h3* ‘ (*carrier L3*) $=$ $ker_{(L4),(L5)}$ *h4* $\wedge$
h4 ‘ (*carrier L4*) $=$ $ker_{(L5),(L6)}$ *h5* $\wedge$ *h5* ‘ (*carrier L5*) $=$ $ker_{(L6),(L7)}$ *h6*

lemma (**in** *Ring*) *exact3-comp-0*:⟦*R module L*; *R module M*; *R module N*;
f $\in$ *mHom R L M*; *g* $\in$ *mHom R M N*; *exact3 R L f M g N*⟧ $\Longrightarrow$
compos L g f $=$ *mzeromap L N*
⟨*proof*⟩

lemma (**in** *Ring*) *exact-im-sub-kern*:⟦*R module L*; *R module M*; *R module N*;
f $\in$ *mHom R L M*; *g* $\in$ *mHom R M N*; *exact3 R L f M g N*⟧ $\Longrightarrow$
f ‘ (*carrier L*) $\subseteq$ $ker_{M,N}$ *g*
⟨*proof*⟩

lemma (**in** *Ring*) *mzero-im-sub-ker*:⟦*R module L*; *R module M*; *R module N*;
f $\in$ *mHom R L M*; *g* $\in$ *mHom R M N*; *compos L g f* $=$ *mzeromap L N*⟧ $\Longrightarrow$

f ‘ (*carrier L*) $\subseteq$ $ker_{M,N}$ *g*
⟨*proof*⟩

lemma (**in** *Ring*) *left-exact-injec*:⟦*R module M*; *R module N*;
z ∈ *mHom R* (*Zm R e*) *M*; *f* ∈ *mHom R M N*; *exact3 R* (*Zm R e*) *z M f N*⟧
⟹
$injec_{M,N}$ *f*
⟨*proof*⟩

lemma (**in** *Ring*) *injec-left-exact*:⟦*R module M*; *R module N*;
z ∈ *mHom R* (*Zm R e*) *M*; *f* ∈ *mHom R M N*; $injec_{M,N}$ *f*⟧ ⟹
exact3 R (*Zm R e*) *z M f N*
⟨*proof*⟩

lemma (**in** *Ring*) *injec-mHom-image*:⟦*R module N*; *R module M1*; *R module M2*;

x ∈ *mHom R N M2*; *f* ∈ *mHom R M1 M2*; *x* ‘ (*carrier N*) ⊆ *f* ‘ (*carrier M1*);
$injec_{M1,M2}$ *f*⟧⟹
(λ*n* ∈(*carrier N*). (*SOME m*. (*m* ∈ *carrier M1* ∧ *x n* = *f m*))) ∈ *mHom R N M1* ∧
compos N f (λ*n* ∈ (*carrier N*). (*SOME m*. *m* ∈ *carrier M1* ∧ *x n* = *f m*)) = *x*
⟨*proof*⟩

lemma (**in** *Ring*) *right-exact-surjec*:⟦*R module M*; *R module N*; *f* ∈ *mHom R M N*;
p ∈ *mHom R N* (*Zm R e*); *exact3 R M f N p* (*Zm R e*)⟧ ⟹ $surjec_{M,N}$ *f*
⟨*proof*⟩

lemma (**in** *Ring*) *surjec-right-exact*:⟦*R module M*; *R module N*; *f* ∈ *mHom R M N*;
p ∈ *mHom R N* (*Zm R e*); $surjec_{M,N}$ *f*⟧ ⟹ *exact3 R M f N p* (*Zm R e*)
⟨*proof*⟩

lemma (**in** *Ring*) *exact4-exact3*:⟦*R module M*; *R module N*; *z* ∈ *mHom R* (*Zm R e*) *M*;
f ∈ *mHom R M N*; *z1* ∈ *mHom R N* (*Zm R e*);
exact4 R (*Zm R e*) *z M f N z1* (*Zm R e*) ⟧ ⟹
exact3 R (*Zm R e*) *z M f N* ∧ *exact3 R M f N z1* (*Zm R e*)
⟨*proof*⟩

lemma (**in** *Ring*) *exact4-bijec*:⟦*R module M*; *R module N*; *z* ∈ *mHom R* (*Zm R e*) *M*;
f ∈ *mHom R M N*; *z1* ∈ *mHom R N* (*Zm R e*);
exact4 R (*Zm R e*) *z M f N z1* (*Zm R e*)⟧ ⟹ $bijec_{M,N}$ *f*
⟨*proof*⟩

lemma (**in** *Ring*) *exact-im-sub-ker*:⟦*R module L*; *R module M*; *R module N*;
f ∈ *mHom R L M*; *g* ∈ *mHom R M N*; *z1* ∈ *mHom R N* (*Zm R e*); *R module Z*;

exact4 R L f M g N z1 (Zm R e); x ∈ mHom R M Z; compos L x f = mzeromap L Z⟧
⟹ (λz∈(carrier N). x (SOME y. y ∈ carrier M ∧ g y = z)) ∈ mHom R N Z
⟨proof⟩

lemma (**in** Ring) exact-im-sub-ker1:⟦R module L; R module M; R module N;
f ∈ mHom R L M; g ∈ mHom R M N; z1 ∈ mHom R N (Zm R e); R module Z;
exact4 R L f M g N z1 (Zm R e); x ∈ mHom R M Z;
compos L x f = mzeromap L Z ⟧ ⟹
compos M (λz∈(carrier N). x (SOME y. y ∈ carrier M ∧ g y = z)) g = x
⟨proof⟩

definition
module-iota :: [('r, 'm) Ring-scheme, ('a, 'r, 'm1) Module-scheme] ⇒
'a ⇒ 'a ((mι₋ -) [92, 93]92) **where**
$m\iota_R$ M = (λx∈carrier M. x)

lemma (**in** Ring) short-exact-sequence:⟦R module M; submodule R M N;
z ∈ mHom R (Zm R e) (mdl M N); z1 ∈ mHom R (M $/_m$ N) (Zm R e)⟧ ⟹
exact5 R (Zm R e) z (mdl M N)($m\iota_R$ (mdl M N)) M (mpj M N) (M $/_m$ N) z1 (Zm R e)
⟨proof⟩

lemma (**in** Ring) rexact4-lexact4-HOM:⟦R module M1; R module M2; R module M3;
f ∈ mHom R M1 M2; g ∈ mHom R M2 M3; z1 ∈ mHom R M3 (Zm R e);
exact4 R M1 f M2 g M3 z1 (Zm R e)⟧ ⟹
∀N. R module N ⟶
exact4 R (HOM_R (Zm R e) N) (sup-sharp R M3 (Zm R e) N z1) (HOM_R M3 N)
(sup-sharp R M2 M3 N g) (HOM_R M2 N) (sup-sharp R M1 M2 N f) (HOM_R M1 N)

⟨proof⟩

lemma exact-HOM-exactTr:⟦Ring (R::('r, 'm1) Ring-scheme); f ∈ mHom R M1 M2;
g ∈ mHom R M2 M3; z1 ∈ mHom R M3 (Zm R e); R module NV;
∀(N::('a, 'r, 'm) Module-scheme). R module N ⟶
exact4 R (HOM_R (Zm R e) N)(sup-sharp R M3 (Zm R e) N z1)
(HOM_R M3 N) (sup-sharp R M2 M3 N g) (HOM_R M2 N) (sup-sharp R M1 M2 N f)
(HOM_R M1 N); R module (L::('a, 'r, 'm) Module-scheme)⟧ ⟹
exact4 R (HOM_R (Zm R e) L) (sup-sharp R M3 (Zm R e) L z1)
(HOM_R M3 L) (sup-sharp R M2 M3 L g) (HOM_R M2 L) (sup-sharp R M1 M2

L f)
(*HOM*$_R$ *M1 L*)
⟨*proof*⟩

lemma *lexact4-rexact4-HOM*:⟦*Ring R*; *R module M1*; *R module M2*; *R module M3*;
f ∈ *mHom R M1 M2*; *g* ∈ *mHom R M2 M3*; *z* ∈ *mHom R* (*Zm R e*) *M1*;
exact4 R (*Zm R e*) *z M1 f M2 g M3* ⟧ ⟹
∀ *N*. *R module N* ⟶ *exact4 R* (*HOM*$_R$ *N* (*Zm R e*)) (*sub-sharp R N* (*Zm R e*) *M1 z*)
(*HOM*$_R$ *N M1*) (*sub-sharp R N M1 M2 f*) (*HOM*$_R$ *N M2*) (*sub-sharp R N M2 M3 g*)
(*HOM*$_R$ *N M3*)

⟨*proof*⟩

5.9 Tensor product

definition
prod-carr :: [(′*a*, ′*r*, ′*m*) *Module-scheme*, (′*b*, ′*r*, ′*m*) *Module-scheme*]
⇒ (′*a* ∗ ′*b*) *set* (**infixl** $\times_c$ *100*) **where**
M $\times_c$ *N* = *carrier M* × *carrier N*

definition
bilinear-map :: [′*a* ∗ ′*b* ⇒ ′*c*, (′*r*, ′*m*) *Ring-scheme*,
(′*a*, ′*r*, ′*m1*) *Module-scheme*, (′*b*, ′*r*, ′*m1*) *Module-scheme*,
(′*c*, ′*r*, ′*m1*) *Module-scheme*] ⇒ *bool* **where**
bilinear-map f R M1 M2 N ⟷ *f* ∈ *M1* $\times_c$ *M2* → *carrier N* ∧
f ∈ *extensional* (*M1* $\times_c$ *M2*) ∧
(∀ *x1* ∈ *carrier M1*. ∀ *x2* ∈ *carrier M1*.
∀ *y*∈*carrier M2*.(*f* (*x1* $\pm_{M1}$ *x2*, *y*) = *f* (*x1*, *y*) $\pm_N$ (*f* (*x2*, *y*)))) ∧
(∀ *x*∈*carrier M1*. ∀ *y1*∈*carrier M2*.
∀ *y2*∈*carrier M2*. *f* (*x*, *y1* $\pm_{M2}$ *y2*) = *f* (*x*, *y1*) $\pm_N$ (*f* (*x*, *y2*))) ∧
(∀ *x*∈*carrier M1*. ∀ *y*∈*carrier M2*.
∀ *r*∈*carrier R*. *f* (*r* $\cdot_{s\,M1}$ *x*, *y*) = *r* $\cdot_{s\,N}$ (*f* (*x*, *y*)) ∧
f (*x*, *r* $\cdot_{s\,M2}$ *y*) = *r* $\cdot_{s\,N}$ (*f* (*x*, *y*)))

lemma (**in** *Ring*) *prod-carr-mem*:⟦*R module M*; *R module N*; *m* ∈ *carrier M*;
n ∈ *carrier N*⟧ ⟹ (*m*, *n*) ∈ *M* $\times_c$ *N*
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-func*:*bilinear-map f R M N Z* ⟹
f ∈ *M* $\times_c$ *N* → *carrier Z*
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-mem*:⟦*R module M1*; *R module M2*; *R module N*;

$m1 \in carrier\ M1;\ m2 \in carrier\ M2;\ bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow$
$f\ (m1,\ m2) \in carrier\ N$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-l-add*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m11 \in carrier\ M1;\ m12 \in carrier\ M1;\ m2 \in carrier\ M2;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow$
$f\ (m11 \pm_{M1} m12,\ m2) = f\ (m11,\ m2) \pm_N (f\ (m12,\ m2))$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-l-add1*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m11 \in carrier\ M1;\ m12 \in carrier\ M1;\ m2 \in carrier\ M2;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow$
$f\ (m11 \pm_{M1} m12,\ m2) \pm_N -_{aN} (f\ (m11,\ m2) \pm_N (f\ (m12,\ m2))) = \mathbf{0}_N$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-r-add*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m \in carrier\ M1;\ m21 \in carrier\ M2;\ m22 \in carrier\ M2;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow$
$f\ (m,\ m21 \pm_{M2} m22) = f\ (m,\ m21) \pm_N (f\ (m,\ m22))$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-r-add1*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m \in carrier\ M1;\ m21 \in carrier\ M2;\ m22 \in carrier\ M2;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow$
$f\ (m,\ m21 \pm_{M2} m22) \pm_N -_{aN} (f\ (m,\ m21) \pm_N (f\ (m,\ m22))) = \mathbf{0}_N$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-l-lin*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m1 \in carrier\ M1;\ m2 \in carrier\ M2;\ r \in carrier\ R;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow f\ (r \cdot_{sM1} m1,\ m2) = r \cdot_{sN} (f\ (m1,\ m2))$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-l-lin1*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m1 \in carrier\ M1;\ m2 \in carrier\ M2;\ r \in carrier\ R;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow$
$f\ (r \cdot_{sM1} m1,\ m2) \pm_N -_{aN} (r \cdot_{sN} (f\ (m1,\ m2))) = \mathbf{0}_N$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-r-lin*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m1 \in carrier\ M1;\ m2 \in carrier\ M2;\ r \in carrier\ R;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow f\ (m1,\ r \cdot_{sM2} m2) = r \cdot_{sN} (f\ (m1,\ m2))$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-r-lin1*: $[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ N;$
$m1 \in carrier\ M1;\ m2 \in carrier\ M2;\ r \in carrier\ R;$
$bilinear\text{-}map\ f\ R\ M1\ M2\ N]\!] \Longrightarrow$
$f\ (m1,\ r \cdot_{sM2} m2)\ \pm_N -_{aN} (r \cdot_{sN} (f\ (m1,\ m2))) = \mathbf{0}_N$
⟨proof⟩

lemma (**in** *Ring*) *bilinear-l-0*:⟦*R module M1*; *R module M2*; *R module N*;
m2 ∈ *carrier M2*; *bilinear-map f R M1 M2 N*⟧ ⟹ *f* ($\mathbf{0}_{M1}$, *m2*) = $\mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-r-0*:⟦*R module M1*; *R module M2*; *R module N*;
m1 ∈ *carrier M1*; *bilinear-map f R M1 M2 N*⟧ ⟹ *f* (*m1*, $\mathbf{0}_{M2}$) = $\mathbf{0}_N$
⟨*proof*⟩

definition
universal-property :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('b, 'r, 'm1) *Module-scheme*, ('c, 'r, 'm1) *Module-scheme*,
'a ∗ 'b ⇒'c] ⇒ *bool* **where**
universal-property (*R*::('r, 'm) *Ring-scheme*) (*M*::('a, 'r, 'm1) *Module-scheme*)
(*N*:: ('b, 'r, 'm1) *Module-scheme*) (*MN*::('c, 'r, 'm1) *Module-scheme*)
(*f*:: 'a ∗ 'b ⇒ 'c) ⟷ (*bilinear-map f R M N MN*) ∧
(∀(*Z* :: ('c, 'r, 'm1) *Module-scheme*). ∀(*g* :: 'a ∗ 'b ⇒ 'c). (*R module Z*) ∧
(*bilinear-map g R M N Z*) ⟶ ((∃!*h*. (*h* ∈ *mHom R MN Z*) ∧
(*compose* (*M* $\times_c$ *N*) *h f* = *g*))))

lemma *tensor-prod-uniqueTr*:⟦*Ring R*; *R module* (*M*::('a, 'r, 'm1) *Module-scheme*);

R module (*N*:: ('b, 'r, 'm1) *Module-scheme*);
R module (*MN*:: ('c, 'r, 'm1) *Module-scheme*);
R module (*MN1*::('c, 'r, 'm1) *Module-scheme*);
universal-property R M N MN f; *universal-property R M N MN1 g*⟧ ⟹
∃!*k*. *k* ∈ *mHom R MN1 MN* ∧ *compose* (*M* $\times_c$ *N*) *k g* = *f*
⟨*proof*⟩

lemma *tensor-prod-unique*:⟦*Ring* (*R*:: ('r, 'm) *Ring-scheme*);
R module (*M* :: ('a, 'r, 'm1) *Module-scheme*);
R module (*N*:: ('b, 'r, 'm1) *Module-scheme*);
R module (*MN*:: ('c, 'r, 'm1) *Module-scheme*);
R module (*MN1*::('c, 'r, 'm1) *Module-scheme*);
universal-property R M N MN f; *universal-property R M N MN1 g*⟧ ⟹
MN $\cong_R$ *MN1*
⟨*proof*⟩

Chapter 6

Construction of an abelian group

6.1 Free generated abelian group I, direct sum and direct product 2

definition
bpp :: $['a \Rightarrow 'a \Rightarrow 'a, 'a, 'a] \Rightarrow 'a$ **where**
bpp f a b $= f\ a\ b$

definition
ipp :: $['a \Rightarrow 'a, 'a] \Rightarrow 'a$ ((_−/ -) [64,65]64) **where**
$_{i}-\ a == i\ a$

definition
sop :: $['r \Rightarrow 'a \Rightarrow 'a, 'r, 'a] \Rightarrow 'a$ **where**
sop s r a $= s\ r\ a$

abbreviation
BOP :: $['a, 'a \Rightarrow 'a \Rightarrow 'a, 'a] \Rightarrow 'a$
((3-/ _+/ -) [62,62,63]62) **where**
$a\ _{f}+\ b ==$ *bpp f a b*

abbreviation
SOP :: $['r, 'r \Rightarrow 'a \Rightarrow 'a, 'a] \Rightarrow 'a$
((3-/ _· -) [68,68,69]68) **where**
$r\ _{s}\cdot\ a ==$ *sop s r a*

definition
minus-set :: $['a \Rightarrow 'a, 'a\ set] \Rightarrow 'a\ set$ **where**
minus-set i A $= \{x.\ \exists y \in A.\ x = {_i}-\ y\}$

definition
pm-set :: $['a \Rightarrow 'a, 'a\ set] \Rightarrow 'a\ set$ **where**

pm-set i A = *A* ∪ (*minus-set i A*)

definition
s-set :: [(′*r*, ′*m*) *Ring-scheme*, ′*r* ⇒ ′*a* ⇒ ′*a*, ′*a set*] ⇒ ′*a set* **where**
s-set R s A = {*x*. ∃ *r*∈*carrier R*. ∃ *a*∈*A*. *x* = *r* $_s$· *a*} ∪ *A*

primrec *add-set* :: [′*a* ⇒ ′*a* ⇒ ′*a*, ′*a set*] ⇒ *nat* ⇒ ′*a set*
where
add-set-0 : *add-set f A 0* = *A*
| *add-set-Suc*: *add-set f A* (*Suc n*) =
{*x*. ∃ *s*∈ (*add-set f A n*). ∃ *t*∈ *A*. *x* = *s* $_f$+ *t*}

definition
aug-pm-set :: [′*a*, ′*a* ⇒ ′*a*, ′*a set*] ⇒ ′*a set* **where**
aug-pm-set z i A = {*z*} ∪ *A* ∪ (*minus-set i A*)

definition
addition-set :: [′*a* ⇒ ′*a* ⇒ ′*a*, ′*a set*] ⇒ ′*a set* **where**
addition-set f A = ⋃{*add-set f A n* | *n*. (*0*::*nat*)≤ *n*}

definition
assoc-bpp :: [′*a set*, ′*a* ⇒ ′*a* ⇒ ′*a*] ⇒ *bool* **where**
assoc-bpp A f ⟷
(∀ *a*∈(*addition-set f A*). ∀ *b*∈(*addition-set f A*). ∀ *c*∈(*addition-set f A*). (*a* $_f$+ *b*) $_f$+ *c* = *a* $_f$+ (*b* $_f$+ *c*))

definition
commute-bpp :: [′*a* ⇒ ′*a* ⇒ ′*a*, ′*a set*] ⇒ *bool* **where**
commute-bpp f A ⟷ (∀ *x*∈*addition-set f A*. ∀ *y*∈*addition-set f A*. *x* $_f$+ *y* = *y* $_f$+ *x*)

definition
zeroA :: [′*a*, ′*a* ⇒ ′*a*, ′*a* ⇒ ′*a* ⇒ ′*a*, ′*a set*] ⇒ ′*a* ⇒ *bool* **where**
zeroA z i f A z1 ⟷ (∀ *x* ∈ *addition-set f* (*aug-pm-set z i A*). *z1* $_f$+ *x* = *x*)

definition
inv-ipp :: [′*a*, ′*a* ⇒ ′*a*, ′*a* ⇒ ′*a* ⇒ ′*a*, ′*a set*] ⇒ *bool* **where**
inv-ipp z i f A ⟷ (∀ *a*∈*addition-set f* (*aug-pm-set z i A*). *zeroA z i f A* (($_i$− *a*) $_f$+ *a*))

definition
ipp-cond1 :: [′*a set*, ′*a* ⇒ ′*a*] ⇒ *bool* **where**
ipp-cond1 A i ⟷ (∀ *x*∈*A*. $_i$− ($_i$− *x*) = *x*)

definition
ipp-cond2 :: [′*a*, ′*a set*, ′*a* ⇒ ′*a*, ′*a* ⇒ ′*a* ⇒ ′*a*] ⇒ *bool* **where**
ipp-cond2 z A i f == ∀ *x*∈(*addition-set f* (*aug-pm-set z i A*)).
∀ *y*∈ (*addition-set f* (*aug-pm-set z i A*)). $_i$−(*x* $_f$+ *y*) = $_i$− *y* $_f$+ ($_i$− *x*)

definition
ipp-cond3 :: $['a, 'a \Rightarrow 'a] \Rightarrow$ *bool* **where**
ipp-cond3 $z\ i \longleftrightarrow {}_i{-}\ z = z$

lemma *add-set-mono*:$A \subseteq B \Longrightarrow$ *add-set f A n* $\subseteq$ *add-set f B n*
⟨*proof*⟩

lemma *addition-inc-add*:*add-set f A n* $\subseteq$ *addition-set f A*
⟨*proof*⟩

lemma *addition-inc-add0*: $A \subseteq$ *addition-set f A*
⟨*proof*⟩

lemma *addition-set-mono*:$A \subseteq B \Longrightarrow$ *addition-set f A* $\subseteq$ *addition-set f B*
⟨*proof*⟩

lemma *a-in-aug-pm-set*:$a \in A \Longrightarrow a \in$ *aug-pm-set z i A*
⟨*proof*⟩

lemma *A-sub-aug-pm-set*:$A \subseteq$ *aug-pm-set z i A*
⟨*proof*⟩

lemma *addition-sub-aug-pm-addition*:
addition-set f A $\subseteq$ *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *assoc-bpp-restrict*:⟦ $A \subseteq B$; *assoc-bpp B f*⟧ $\Longrightarrow$ *assoc-bpp A f*
⟨*proof*⟩

lemma *addition-assoc*:⟦*assoc-bpp A f*; $x \in$ *addition-set f A*;
$y \in$ *addition-set f A*; $z \in$ *addition-set f A*⟧ $\Longrightarrow$
$(x\ {}_f{+}\ y)\ {}_f{+}\ z = x\ {}_f{+}\ (y\ {}_f{+}\ z)$
⟨*proof*⟩

lemma *bpp-closedTr*:*assoc-bpp A f* $\Longrightarrow$
$\forall x\ y.\ x \in$ *add-set f A n* $\wedge\ y \in$ *add-set f A m* $\longrightarrow$
$x\ {}_f{+}\ y \in$ *add-set f A* $(n + m + Suc\ 0)$
⟨*proof*⟩

lemma *bpp-closed1*:⟦*assoc-bpp A f*; $x \in$ *add-set f A n*; $y \in$ *add-set f A m*⟧ $\Longrightarrow$
$x\ {}_f{+}\ y \in$ *add-set f A* $(n + m + Suc\ 0)$
⟨*proof*⟩
lemma *bpp-closed*:⟦*assoc-bpp A f*; $x \in$ *addition-set f A*; $y \in$ *addition-set f A*⟧
$\Longrightarrow\ x\ {}_f{+}\ y \in$ *addition-set f A*
⟨*proof*⟩

lemma *aug-addition-inc-z*: $z \in$ *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *aug-bpp-closed*:$[\![$*assoc-bpp* (*aug-pm-set* z i A) f;
$x \in$ *addition-set* f (*aug-pm-set* z i A);
$y \in$ *addition-set* f (*aug-pm-set* z i A)$]\!] \Longrightarrow$
x ${}_f+$ $y \in$ *addition-set* f (*aug-pm-set* z i A)
⟨*proof*⟩

lemma *aug-commute*:$[\![$*commute-bpp* f (*aug-pm-set* z i A);
$x \in$ *addition-set* f (*aug-pm-set* z i A);
$y \in$ *addition-set* f (*aug-pm-set* z i A)$]\!] \Longrightarrow x$ ${}_f+$ $y = y$ ${}_f+$ x
⟨*proof*⟩

lemma *addition-set-inc-z*:$z \in$ *addition-set* f (*aug-pm-set* z i A)
⟨*proof*⟩

lemma *aug-ipp-closed0*:$[\![$*commute-bpp* f (*aug-pm-set* z i A);
assoc-bpp (*aug-pm-set* z i A) f; *ipp-cond1* A i; *ipp-cond2* z A i f;
ipp-cond3 z i; $x \in$ *add-set* f (*aug-pm-set* z i A) *0*$]\!] \Longrightarrow$
${}_i-$ $x \in$ *add-set* f (*aug-pm-set* z i A) *0*
⟨*proof*⟩

lemma *aug-ipp-closedTr*:$[\![$*commute-bpp* f (*aug-pm-set* z i A);
assoc-bpp (*aug-pm-set* z i A) f; *ipp-cond1* A i; *ipp-cond2* z A i f;
ipp-cond3 z $i$$]\!] \Longrightarrow$
$\forall x.$ $x \in$ *add-set* f (*aug-pm-set* z i A) $n \longrightarrow$
${}_i-$ $x \in$ *add-set* f (*aug-pm-set* z i A) n
⟨*proof*⟩

lemma *aug-ipp-closedTr2*:$[\![$*commute-bpp* f (*aug-pm-set* z i A);
assoc-bpp (*aug-pm-set* z i A) f; *ipp-cond1* A i; *ipp-cond2* z A i f;
ipp-cond3 z i; $x \in$ *add-set* f (*aug-pm-set* z i A) $n$$]\!] \Longrightarrow$
${}_i-$ $x \in$ *add-set* f (*aug-pm-set* z i A) n
⟨*proof*⟩

lemma *aug-ipp-closed*:$[\![$*commute-bpp* f (*aug-pm-set* z i A);
assoc-bpp (*aug-pm-set* z i A) f; *ipp-cond1* A i; *ipp-cond2* z A i f;
ipp-cond3 z i; $x \in$ *addition-set* f (*aug-pm-set* z i A)$]\!] \Longrightarrow$
${}_i-$ $x \in$ *addition-set* f (*aug-pm-set* z i A)
⟨*proof*⟩

lemma *aug-zero-unique*:$[\![$*commute-bpp* f (*aug-pm-set* z i A);
z1 $\in$ *addition-set* f (*aug-pm-set* z i A); *zeroA* z i f A z;
zeroA z i f A *z1*$]\!] \Longrightarrow z =$ *z1*
⟨*proof*⟩

lemma *inv-aug-addition*:$[\![$*commute-bpp* f (*aug-pm-set* z i A);
assoc-bpp (*aug-pm-set* z i A) f; *ipp-cond1* A i; *ipp-cond2* z A i f;
ipp-cond3 z i; *inv-ipp* z i f A; *commute-bpp* f (*aug-pm-set* z i A);
zeroA z i f A $z$$]\!] \Longrightarrow$
$\forall a \in$*addition-set* f (*aug-pm-set* z i A). $({}_i-a)$ ${}_f+$ $a = z$

⟨*proof*⟩

definition
fag-gen-by :: ['*a set*, '*a* ⇒ '*a* ⇒ '*a*, '*a* ⇒ '*a*, '*a*] ⇒ '*a aGroup* **where**
fag-gen-by A f i z = (|*carrier* = *addition-set f* (*aug-pm-set z i A*),
pop = $\lambda x \in$(*addition-set f* (*aug-pm-set z i A*)).
$\lambda y \in$(*addition-set f* (*aug-pm-set z i A*)). $x\ {}_f\!+\ y$,
mop = $\lambda x \in$(*addition-set f* (*aug-pm-set z i A*)). ${}_i\!-\ x$, *zero* = *z*|)

lemma *fag-gen-carrier*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *inv-ipp z i f A*; *commute-bpp f* (*aug-pm-set z i A*);
zeroA z i f A z⟧ ⟹
carrier (*fag-gen-by A f i z*) = *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *addition-set-sub-fag-gen-carrier*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *inv-ipp z i f A*; *commute-bpp f* (*aug-pm-set z i A*);
zeroA z i f A z⟧ ⟹ *addition-set f A* ⊆ *carrier* (*fag-gen-by A f i z*)
⟨*proof*⟩

lemma *fag-aGroup*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *inv-ipp z i f A*; *commute-bpp f* (*aug-pm-set z i A*);
zeroA z i f A z⟧ ⟹ *aGroup* (*fag-gen-by A f i z*)
⟨*proof*⟩

6.2 Abelian group generated by a singleton (constructive)

definition
fag-single :: ['*a*, '*a* ⇒ '*a* ⇒ '*a*, '*a* ⇒ '*a*, '*a*] ⇒ '*a aGroup* **where**
fag-single a f i z = *fag-gen-by* {*a*} *f i z*

lemma *aug-pm-aug-pm-minus*:*ipp-cond1* {*a*} *i* ⟹
aug-pm-set z i {*a*} = *aug-pm-set z i* {${}_i\!-\ a$}
⟨*proof*⟩

lemma *ipp-cond1-minus*:*ipp-cond1* {*a*} *i* ⟹ *ipp-cond1* {${}_i\!-\ a$} *i*
⟨*proof*⟩

lemma *ipp-cond2-minus*:⟦*ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*⟧ ⟹
ipp-cond2 z {${}_i\!-\ a$} *i f*
⟨*proof*⟩

lemma *zeroA-minus*:⟦*ipp-cond1* {*a*} *i*; *zeroA z i f* {*a*} *z1*⟧ ⟹

zeroA z i f $\{_i- \ a\}$ *z1*
⟨*proof*⟩

lemma *inv-ipp-minus*:⟦*ipp-cond1* $\{a\}$ *i*; *inv-ipp z i f* $\{a\}$⟧ ⟹
inv-ipp z i f $\{_i- \ a\}$
⟨*proof*⟩

lemma *fag-single-additionTr1*:⟦*commute-bpp f* (*aug-pm-set z i* $\{a\}$);
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*;
ipp-cond3 z i; *inv-ipp z i f* $\{a\}$; *commute-bpp f* (*aug-pm-set z i* $\{a\}$);
zeroA z i f $\{a\}$ *z*⟧ ⟹
∀ *s*. *s*∈ *add-set f* $\{a\}$ (*Suc n*) ⟶ *s* $_f+$ $_i-$ *a* ∈ *add-set f* $\{a\}$ *n*
⟨*proof*⟩

lemma *fag-single-additionTr2*:⟦*commute-bpp f* (*aug-pm-set z i* $\{a\}$);
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*;
ipp-cond3 z i; *inv-ipp z i f* $\{a\}$; *commute-bpp f* (*aug-pm-set z i* $\{a\}$);
zeroA z i f $\{a\}$ *z*; *s* ∈ *add-set f* $\{a\}$ *0*⟧ ⟹ *s* $_f+$ $_i-$ *a* = *z*
⟨*proof*⟩

lemma *ipp-conditions*:⟦*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*;
ipp-cond2 z $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ *z*⟧ ⟹
assoc-bpp (*aug-pm-set z i* $\{ _i- \ a\}$) *f* ∧ *ipp-cond1* $\{ _i- \ a\}$ *i* ∧
ipp-cond2 z $\{ _i- \ a\}$ *i f* ∧ *inv-ipp z i f* $\{ _i- \ a\}$ ∧
commute-bpp f (*aug-pm-set z i* $\{ _i- \ a\}$) ∧ *zeroA z i f* $\{ _i- \ a\}$ *z*
⟨*proof*⟩

lemma *fag-single-additionTr3*:⟦*commute-bpp f* (*aug-pm-set z i* $\{a\}$);
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*;
ipp-cond3 z i; *inv-ipp z i f* $\{a\}$; *commute-bpp f* (*aug-pm-set z i* $\{a\}$);
zeroA z i f $\{a\}$ *z*; *s*∈ *add-set f* $\{_i- \ a\}$ *n*⟧ ⟹
s $_f+$ $_i-$ *a* ∈ *add-set f* $\{_i- \ a\}$ (*Suc n*)
⟨*proof*⟩

lemma *fag-single-elemTr*:⟦*commute-bpp f* (*aug-pm-set z i* $\{a\}$);
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*;
ipp-cond3 z i; *inv-ipp z i f* $\{a\}$; *commute-bpp f* (*aug-pm-set z i* $\{a\}$);
zeroA z i f $\{a\}$ *z*⟧ ⟹
∀ *x*. *x* ∈ *add-set f* (*aug-pm-set z i* $\{a\}$) *n* ⟶
(∃ *n1*. *x* ∈ *add-set f* $\{a\}$ *n1*) ∨ (∃ *m1*. *x* ∈ *add-set f* $\{_i- \ a\}$ *m1*) ∨ *x* = *z*
⟨*proof*⟩

lemma *fag-single-elem*:⟦*commute-bpp f* (*aug-pm-set z i* $\{a\}$);
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*;
ipp-cond3 z i; *inv-ipp z i f* $\{a\}$; *commute-bpp f* (*aug-pm-set z i* $\{a\}$);
zeroA z i f $\{a\}$ *z*; *x* ∈ *addition-set f* (*aug-pm-set z i* $\{a\}$)⟧ ⟹
(∃ *n1*. *x* ∈ *add-set f* $\{a\}$ *n1*) ∨ (∃ *m1*. *x* ∈ *add-set f* $\{_i- \ a\}$ *m1*) ∨ *x* = *z*

⟨*proof*⟩

lemma *add-set-single1Tr*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹
∀ *x y*. *x* ∈ *add-set f* {*a*} *n* ∧ *y* ∈ *add-set f* {*a*} *n* ⟶ *x* = *y*
⟨*proof*⟩

lemma *add-set-single-nonempty1*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ ∃ *x*. *x*∈*add-set f* {*a*} *n*
⟨*proof*⟩

lemma *add-set-single-nonempty2*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ ∃ *x*. *x*∈*add-set f* $\{_i- a\}$ *n*
⟨*proof*⟩

lemma *add-set-single1*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *x* ∈ *add-set f* {*a*} *n*; *y* ∈ *add-set f* {*a*} *n*⟧ ⟹ *x* = *y*
⟨*proof*⟩

lemma *add-set-single2*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *x* ∈ *add-set f* $\{_i- a\}$ *n*; *y* ∈ *add-set f* $\{_i- a\}$ *n*⟧ ⟹
x = *y*
⟨*proof*⟩

lemma *fag-single-additionTr4*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z* ⟧ ⟹
∀ *s t*. *s* ∈ *add-set f* {*a*} *n* ∧ *t* ∈ *add-set f* $\{_i- a\}$ *n*⟶ $s\ {}_f+\ t = z$
⟨*proof*⟩

lemma *fag-single-additionTr4-1*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*;*s* ∈ *add-set f* {*a*} *n*; *t* ∈ *add-set f* $\{_i- a\}$ *n* ⟧ ⟹
$s\ {}_f+\ t = z$
⟨*proof*⟩

lemma *fag-single-additionTr5*:⟦*assoc-bpp* (*aug-pm-set z i* {*a*}) *f*;

ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ $z\rrbracket \Longrightarrow$
$\forall$ *m*. $m <$ *Suc n* $\longrightarrow$ (*THE x*. $x \in$ *add-set f* $\{a\}$ (*Suc n*)) $_f+$
(*THE x*. $x \in$ *add-set f* $\{_i- a\}$ *m*) = (*THE x*. $x \in$ *add-set f* $\{a\}$ $(n - m)$)
⟨*proof*⟩

lemma *fag-single-additionTr5-1*:⟦*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f*;
ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ *z*; $m <$ *Suc n*⟧ $\Longrightarrow$
(*THE x*. $x \in$ *add-set f* $\{a\}$ (*Suc n*)) $_f+$ (*THE x*. $x \in$ *add-set f* $\{_i- a\}$ *m*) =
(*THE x*. $x \in$ *add-set f* $\{a\}$ $(n - m)$)
⟨*proof*⟩

lemma *fag-single-additionTr5-2*:⟦*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f*;
ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ *z*; $n <$ *Suc m*⟧ $\Longrightarrow$
(*THE x*. $x \in$ *add-set f* $\{_i- a\}$ (*Suc m*)) $_f+$ (*THE x*. $x \in$ *add-set f* $\{a\}$ *n*) =
(*THE x*. $x \in$ *add-set f* $\{_i- a\}$ $(m - n)$)
⟨*proof*⟩

definition
free-gen-condition :: $['a \Rightarrow 'a \Rightarrow 'a,\ 'a \Rightarrow 'a,\ 'a,\ 'a] \Rightarrow$ *bool* **where**
free-gen-condition f i a z $\longleftrightarrow$ ($\forall$ *n*. $z \notin$ *add-set f* $\{a\}$ *n*)

definition
fg-elem-single :: $['a \Rightarrow 'a \Rightarrow 'a,\ 'a \Rightarrow 'a,\ 'a,\ 'a] \Rightarrow$ *int* $\Rightarrow$ $'a$ **where**
fg-elem-single f i a z n = (*if* $0 = n$ *then z else*
(*if* $0 < n$ *then* (*THE x*. $x \in$ (*add-set f* $\{a\}$ (*nat* $(n - 1)$))))
else (*THE x*. $x \in$ (*add-set f* $\{_i- a\}$ (*nat* $(- n - 1)$))))))

abbreviation
FGELEMSNGLE ((5-$\bigodot$ -$_{-,-,-}$) [99,98,98,98,98]99) **where**
$n\bigodot a_{f,i,z}$ == *fg-elem-single f i a z n*

lemma *single-addition-pm-mem*:⟦*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f*;
ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ $z\rrbracket \Longrightarrow$
($n\bigodot a_{f,i,z}$) $\in$ *addition-set f* (*aug-pm-set z i* $\{a\}$)
⟨*proof*⟩

lemma *assoc-aug-assoc*:*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f* $\Longrightarrow$ *assoc-bpp* $\{a\}$ *f*
⟨*proof*⟩

lemma *single-addition-posTr*:⟦*commute-bpp f* (*aug-pm-set z i* $\{(a::'a)\}$);
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*;
ipp-cond3 z i; *inv-ipp z i f* $\{a\}$; *commute-bpp f* (*aug-pm-set z i* $\{a\}$);
zeroA z i f $\{a\}$ *z*; $0 < (n::int)$; $0 < (m::int)$⟧ $\Longrightarrow$
(*THE x*. $x \in$ *add-set f* $\{a\}$ (*nat* $(n - 1)$)) $_f+$

$(THE\ x.\ x \in add\text{-}set\ f\ \{a\}\ (nat\ (m - 1))) =$
$(THE\ x.\ x \in add\text{-}set\ f\ \{a\}\ (nat\ (n + m - 1)))$
⟨*proof*⟩

lemma *single-addition-pos*:⟦*commute-bpp f (aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *0* < (*n*::*int*); *0* < (*m*::*int*)⟧ ⟹
$(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$
⟨*proof*⟩

lemma *single-addition-neg*:⟦*commute-bpp f (aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; (*n*::*int*) < *0*; (*m*::*int*) < *0* ⟧ ⟹
$(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$
⟨*proof*⟩

lemma *single-addition-zero*:⟦*commute-bpp f (aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ $0\bigodot a_{f,i,z} = z$
⟨*proof*⟩

lemma *s-a-p-1*:⟦*assoc-bpp* (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*;
ipp-cond2 z {*a*} *i f*; *ipp-cond3 z i*; *inv-ipp z i f* {*a*};
commute-bpp f (*aug-pm-set z i* {*a*}); *zeroA z i f* {*a*} *z*;
m < *0*; *0* < *n*⟧ ⟹ $(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$

⟨*proof*⟩

lemma *single-addition-pm*:⟦*commute-bpp f (aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ $(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$
⟨*proof*⟩

lemma *single-inv*:⟦*commute-bpp f (aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ ${}_i{-}\ (m\bigodot a_{f,i,z}) = (-m)\bigodot a_{f,i,z}$
⟨*proof*⟩

lemma *free-ag-single*:⟦*commute-bpp f (aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *free-gen-condition f i a z*; *n* ≠ *m*⟧ ⟹
$(n\bigodot a_{f,i,z}) \neq (m\bigodot a_{f,i,z})$
⟨*proof*⟩

definition
fags-cond :: $['a \Rightarrow 'a \Rightarrow 'a, 'a, 'a \Rightarrow 'a, 'a] \Rightarrow bool$ **where**
fags-cond f z i a $\longleftrightarrow$ *commute-bpp f* (*aug-pm-set z i* $\{a\}$) $\wedge$
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f* $\wedge$ *ipp-cond1* $\{a\}$ *i* $\wedge$
ipp-cond2 z $\{a\}$ *i f* $\wedge$ *ipp-cond3 z i* $\wedge$ *inv-ipp z i f* $\{a\}$ $\wedge$
commute-bpp f (*aug-pm-set z i* $\{a\}$) $\wedge$ *zeroA z i f* $\{a\}$ *z* $\wedge$
free-gen-condition f i a z

lemma *fag-single-free*:$[\![$*fags-cond f z i a*; $n \neq m]\!] \Longrightarrow (n \bigodot a_{f,i,z}) \neq (m \bigodot a_{f,i,z})$
⟨*proof*⟩

lemma *fag-single-free1*:$[\![$*fags-cond f z i a*;$(n \bigodot a_{f,i,z}) = (m \bigodot a_{f,i,z})]\!] \Longrightarrow n = m$
⟨*proof*⟩

definition
fags-carr :: $['a \Rightarrow 'a \Rightarrow 'a, 'a, 'a \Rightarrow 'a, 'a] \Rightarrow 'a$ *set* **where**
fags-carr f z i a $= \{x.\ \exists n.\ x = n \bigodot a_{f,i,z}\}$

definition
fags-bpp :: $['a \Rightarrow 'a \Rightarrow 'a, 'a, 'a \Rightarrow 'a, 'a] \Rightarrow 'a \Rightarrow 'a \Rightarrow 'a$ **where**
fags-bpp f z i a $= (\lambda x \in ($*fags-carr f z i a*$).\ \lambda y \in ($*fags-carr f z i a*$).$
$((THE\ n.\ x = n \bigodot a_{f,i,z}) + (THE\ m.\ y = m \bigodot a_{f,i,z})) \bigodot a_{f,i,z})$

definition
fags-ipp :: $['a \Rightarrow 'a \Rightarrow 'a, 'a, 'a \Rightarrow 'a, 'a] \Rightarrow 'a \Rightarrow 'a$ **where**
fags-ipp f z i a $= (\lambda x \in ($*fags-carr f z i a*$).$
$(- (THE\ n.\ x = n \bigodot a_{f,i,z})) \bigodot a_{f,i,z})$

lemma *fags-mem*:*fags-cond f z i a* $\Longrightarrow (n \bigodot a_{f,i,z}) \in$ *fags-carr f z i a*
⟨*proof*⟩

lemma *fags-ippTr*:*fags-cond f z i a* $\Longrightarrow$
fags-ipp f z i a $(n \bigodot a_{f,i,z}) = (- n) \bigodot a_{f,i,z}$
⟨*proof*⟩

lemma *fags-bppTr*:*fags-cond f z i a* $\Longrightarrow$
fags-bpp f z i a $(n \bigodot a_{f,i,z})\ (m \bigodot a_{f,i,z}) = (n + m) \bigodot a_{f,i,z}$
⟨*proof*⟩

definition
fags :: $['a \Rightarrow 'a \Rightarrow 'a, 'a, 'a \Rightarrow 'a, 'a] \Rightarrow 'a$ *aGroup* **where**
fags f z i a = (|*carrier* = *fags-carr f z i a*,
pop = *fags-bpp f z i a*,
mop = *fags-ipp f z i a*, *zero* = *z*|)

lemma *fags-ag*:*fags-cond f z i a* $\Longrightarrow$ *aGroup* (*fags f z i a*)
⟨*proof*⟩

6.3 Abelian Group generated by one element II (nonconstructive)

definition
 ag-single-gen :: $[('a, 'm)$ *aGroup-scheme*, $'a] \Rightarrow$ *bool* **where**
 ag-single-gen A a $\longleftrightarrow$ *aGroup A* $\wedge$ *carrier A* $= \bigcap$ $\{H.$ *asubGroup A H* $\wedge$ $a \in H\}$

primrec *aSum* :: $[('a, 'm)$ *aGroup-scheme*, *nat*,$'a] \Rightarrow 'a$ **where**
 aSum-0: *aSum A 0 a* $= \mathbf{0}_A$
| *aSum-Suc*: *aSum A* (*Suc n*) $a =$ *aSum A n a* $\pm_A$ a

definition
 sprod-n-a ::$[('a, 'm)$ *aGroup-scheme*, *int*, $'a] \Rightarrow 'a$ **where**
 sprod-n-a A n x $=$ (*if* $0 \leq n$ *then* (*aSum A* (*nat n*) x)
 else (*aSum A* (*nat* $(- n)$) $(-_{a A}$ x)))

abbreviation
 SPRODNA ((*3*-$\triangleright$-$_-$) [*95*,*95*,*96*]*95*) **where**
 $n \triangleright a_A$ $==$ *sprod-n-a A n a*

lemma (**in** *aGroup*) *asum-mem*:$a \in$ *carrier A* $\Longrightarrow$ *aSum A n a* $\in$ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mem0*:$a \in$ *carrier A* $\Longrightarrow$ $n \triangleright a_A \in$ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-zero0*:$a \in$ *carrier A* $\Longrightarrow$ $0 \triangleright a_A = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-1*:$a \in$ *carrier A* $\Longrightarrow$ $1 \triangleright a_A = a$
⟨*proof*⟩

lemma (**in** *aGroup*) *asumTr*:$a \in$ *carrier A* $\Longrightarrow$
 aSum A $(n + m)$ $a =$ *aSum A n a* $\pm$ (*aSum A m a*)
⟨*proof*⟩

lemma (**in** *aGroup*) *aSum-zero*:$a \in$ *carrier A* $\Longrightarrow$ *aSum A n* $\mathbf{0} = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add1p*:$[\![$ $a \in$ *carrier A*; $0 \leq n$; $0 \leq m]\!] \Longrightarrow$
 $(n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add1m*:$[\![$ $a \in$ *carrier A*; $n < 0$; $m < 0]\!] \Longrightarrow$
 $(n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add2Tr*: $a \in carrier\ A \Longrightarrow$
$\mathbf{0} = aSum\ A\ n\ a \pm (aSum\ A\ n\ (-_a\ a))$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add2p*: $[\![a \in carrier\ A;\ 0 \leq n]\!] \Longrightarrow$
$\mathbf{0} = n \triangleright a_A \pm ((-n) \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add2m*: $[\![a \in carrier\ A;\ n < 0]\!] \Longrightarrow$
$\mathbf{0} = n \triangleright a_A \pm ((-n) \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add3pm*: $[\![a \in carrier\ A;\ 0 < n;\ m < 0]\!] \Longrightarrow$
$(n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add3mp*: $[\![\ a \in carrier\ A;\ n < 0;\ 0 < m]\!] \Longrightarrow$
$(n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-sum0*: $[\![\ a \in carrier\ A]\!] \Longrightarrow (n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-inv0*: $a \in carrier\ A \Longrightarrow -_a\ (n \triangleright a_A) = (-\ n) \triangleright a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *m-x-asum*: $[\![\ a \in carrier\ A;\ b \in carrier\ A]\!]$
$\Longrightarrow aSum\ A\ m\ (a \pm b) = (aSum\ A\ m\ a) \pm (aSum\ A\ m\ b)$
⟨*proof*⟩

lemma (**in** *aGroup*) *asum-multTr-pp*: $a \in carrier\ A \Longrightarrow$
$aSum\ A\ m\ (aSum\ A\ n\ a) = aSum\ A\ (m * n)\ a$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-pp*: $[\![\ a \in carrier\ A;\ 0 \leq m;\ 0 \leq n]\!]$
$\Longrightarrow m \triangleright (n \triangleright a_A)_A = (m * n) \triangleright a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *asum-multTr-pm*: $[\![a \in carrier\ A;\ 0 \leq m;\ n < 0]\!] \Longrightarrow$
$aSum\ A\ (nat\ m)\ (aSum\ A\ (nat\ (-\ n))\ (-_a\ a)) =$
$aSum\ A\ (nat\ (m * (-\ n)))\ (-_a\ a)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-pm*: $[\![a \in carrier\ A;\ 0 \leq m;\ n < 0]\!] \Longrightarrow$
$m \triangleright (n \triangleright a_A)_A = (m * n) \triangleright a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *asum-multTr-mp*: $[\![a \in carrier\ A;\ m < 0;\ 0 \leq n]\!] \Longrightarrow$

$aSum\ A\ (nat\ (-m))(-_a\ (aSum\ A\ (nat\ n)\ a)) = aSum\ A\ (nat\ ((-\ m) * n))\ (-_a\ a)$
⟨proof⟩

lemma (**in** *aGroup*) *nt-mult-mp*: $[\![a \in carrier\ A;\ m < 0;\ 0 \leq n]\!] \Longrightarrow m \triangleright (n \triangleright a_A)_A = (m * n) \triangleright a_A$
⟨proof⟩

lemma (**in** *aGroup*) *asum-multTr-mm*: $[\![a \in carrier\ A;\ m < 0;\ n < 0]\!] \Longrightarrow aSum\ A\ (nat\ (-m))(-_a\ (aSum\ A\ (nat\ (-\ n))\ (-_a\ a))) = aSum\ A\ (nat\ ((-\ m) * (-\ n)))\ a$
⟨proof⟩

lemma (**in** *aGroup*) *nt-mult-mm*: $[\![\ a \in carrier\ A;\ m < 0;\ n < 0]\!] \Longrightarrow m \triangleright (n \triangleright a_A)_A = (m * n) \triangleright a_A$
⟨proof⟩

lemma (**in** *aGroup*) *nt-mult-assoc0*: $a \in carrier\ A \Longrightarrow m \triangleright n \triangleright a_{A\,A} = (m * n) \triangleright a_A$
⟨proof⟩

lemma (**in** *aGroup*) *single-gen-carrTr*: $a \in carrier\ A \Longrightarrow asubGroup\ A\ \{x.\ \exists n.\ x = (n \triangleright a_A)\}$
⟨proof⟩

lemma (**in** *aGroup*) *ag-single-inc-a*: $ag\text{-}single\text{-}gen\ A\ a \Longrightarrow a \in carrier\ A$
⟨proof⟩

lemma (**in** *aGroup*) *single-gen*: $ag\text{-}single\text{-}gen\ A\ a \Longrightarrow carrier\ A = \{g.\ \exists n.\ g = (n \triangleright a_A)\}$
⟨proof⟩

definition
single-gen-free :: [('*a*, '*m*) *aGroup-scheme*, '*a*] ⇒ *bool* **where**
$single\text{-}gen\text{-}free\ A\ a == \forall n.\ n \neq 0 \longrightarrow \mathbf{0}_A \neq n \triangleright a_A$

definition
sfg :: [('*a*, '*m*) *aGroup-scheme*, '*a*] ⇒ *bool* **where**
$sfg\ A\ a \longleftrightarrow ag\text{-}single\text{-}gen\ A\ a \wedge single\text{-}gen\text{-}free\ A\ a$

lemma (**in** *aGroup*) *single-gen-free-neg*: $[\![sfg\ A\ a;\ n \triangleright a_A = \mathbf{0}]\!] \Longrightarrow n = 0$
⟨proof⟩

lemma (**in** *aGroup*) *sfg-G-inc-a*: $sfg\ A\ a \Longrightarrow a \in carrier\ A$
⟨proof⟩

lemma *sfg-agroup*: $sfg\ A\ a \Longrightarrow aGroup\ A$
⟨proof⟩

lemma (**in** *aGroup*) *mem-G-nt*:$[\![sfg\ A\ a;\ x \in carrier\ A]\!] \Longrightarrow \exists\, n.\ x = n \triangleright a_A$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nt-mem*:$sfg\ A\ a \Longrightarrow n \triangleright a_A \in carrier\ A$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nt-zero*:$sfg\ A\ a \Longrightarrow 0 \triangleright a_A = \mathbf{0}$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nt-sum*:$sfg\ A\ a \Longrightarrow (n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nt-inv*:$sfg\ A\ a \Longrightarrow -_a(n \triangleright a_A) = (-\ n) \triangleright a_A$
$\langle proof \rangle$

lemma (**in** *aGroup*) *nt-mult-assoc*:$sfg\ A\ a \Longrightarrow m \triangleright n \triangleright a_{A\,A} = (m * n) \triangleright a_A$
$\langle proof \rangle$

lemma (**in** *aGroup*) *sfg-free*:$[\![sfg\ A\ a;\ n \neq m\ ]\!] \Longrightarrow n \triangleright a_A \neq (m \triangleright a_A)$
$\langle proof \rangle$

lemma (**in** *aGroup*) *sfg-free-inj*:$[\![sfg\ A\ a;\ n \triangleright a_A = (m \triangleright a_A)\ ]\!] \Longrightarrow n = m$
$\langle proof \rangle$

6.4 Free Generated Modules (constructive)

definition
  sop-one::$[('r,\ 'm)\ Ring\text{-}scheme,\ 'r \Rightarrow 'a \Rightarrow 'a,\ 'a\ set] \Rightarrow bool$ **where**
  $sop\text{-}one\ R\ s\ A \longleftrightarrow (\forall x \in A.\ (1_{r\,R})\ {}_s\!\cdot\ x = x)$

definition
  sop-assoc :: $[('r,\ 'm)\ Ring\text{-}scheme,\ 'r \Rightarrow 'a \Rightarrow 'a,\ 'a\ set] \Rightarrow bool$ **where**
  $sop\text{-}assoc\ R\ s\ A \longleftrightarrow (\forall a \in carrier\ R.\ \forall b \in carrier\ R.\ \forall x \in A.$
  $(a \cdot_{r\,R}\ b)\ {}_s\!\cdot\ x = a\ {}_s\!\cdot\ (b\ {}_s\!\cdot\ x))$

definition
  sop-inv :: $[('r,\ 'm)\ Ring\text{-}scheme,\ 'r \Rightarrow 'a \Rightarrow 'a,\ 'a \Rightarrow 'a,\ 'a\ set]$
  $\Rightarrow bool$ **where**
  $sop\text{-}inv\ R\ s\ i\ A \longleftrightarrow (\forall r \in carrier\ R.\ \forall x \in A.\ r\ {}_s\!\cdot\ ({}_i\!-\ x) = (-_{a\,R}\ r)\ {}_s\!\cdot\ x)$

definition
  sop-distr1 :: $[('r,\ 'm)\ Ring\text{-}scheme,\ 'r \Rightarrow 'a \Rightarrow 'a,\ 'a \Rightarrow 'a \Rightarrow 'a,$
  $'a \Rightarrow 'a,\ 'a\ set,\ 'a] \Rightarrow bool$ **where**
  $sop\text{-}distr1\ R\ s\ f\ i\ A\ z \longleftrightarrow (\forall a \in carrier\ R.\ \forall b \in carrier\ R.$
  $\forall x \in (aug\text{-}pm\text{-}set\ z\ i\ A).\ (a \pm_R\ b)\ {}_s\!\cdot\ x = (a\ {}_s\!\cdot\ x)\ {}_f\!+\ (b\ {}_s\!\cdot\ x))$

definition
  sop-distr2 :: $[('r,\ 'm)\ Ring\text{-}scheme,\ 'r \Rightarrow 'a \Rightarrow 'a,\ 'a \Rightarrow 'a \Rightarrow 'a,$
  $'a \Rightarrow 'a,\ 'a\ set,\ 'a] \Rightarrow bool$ **where**

sop-distr2 R s f i A z ⟷ (∀ *a*∈*carrier R*.
∀ *x*∈*addition-set f* (*aug-pm-set z i A*).
∀ *y*∈*addition-set f* (*aug-pm-set z i A*).
$a\ {}_s\cdot\ (x\ {}_f+\ y) = (a\ {}_s\cdot\ x)\ {}_f+\ (a\ {}_s\cdot\ y))$

definition
sop-z :: [(′*r*, ′*m*) *Ring-scheme*, ′*r* ⇒ ′*a* ⇒ ′*a*, ′*a*] ⇒ *bool* **where**
sop-z R s z ⟷ (∀ *r*∈*carrier R*. $r\ {}_s\cdot\ z = z$)

definition
fgmodule :: [(′*r*, ′*m*) *Ring-scheme*, ′*a set*, ′*a*, ′*a* ⇒ ′*a*, ′*a* ⇒ ′*a* ⇒ ′*a*, ′*r* ⇒ ′*a* ⇒ ′*a*] ⇒ (′*a*, ′*r*) *Module* **where**
fgmodule R A z i f s =
(|*carrier* = *addition-set f* (*aug-pm-set z i* (*s-set R s A*)),
pop = λ*x*∈*addition-set f* (*aug-pm-set z i* (*s-set R s A*)).
λ*y*∈*addition-set f* (*aug-pm-set z i* (*s-set R s A*)). $x\ {}_f+\ y$,
mop = λ*x*∈*addition-set f* (*aug-pm-set z i* (*s-set R s A*)). $_i-\ x$,
zero = *z*,
sprod = λ*r*∈*carrier R*.
λ*x*∈*addition-set f* (*aug-pm-set z i* (*s-set R s A*)). $r\ {}_s\cdot\ x$ |)

lemma *fgmodule-carr*:*carrier* (*fgmodule R A z i f s*) =
addition-set f (*aug-pm-set z i* (*s-set R s A*))
⟨*proof*⟩

lemma *a-in-s-set*:*a* ∈ *A* ⟹ *a* ∈ *s-set R s A*
⟨*proof*⟩

lemma (**in** *Ring*) *ra-in-s-set*:⟦*r* ∈ *carrier R*; *a* ∈ *A*⟧ ⟹ $r\ {}_s\cdot\ a$ ∈ *s-set R s A*
⟨*proof*⟩

lemma *in-aug-pm-set*:
x ∈ *aug-pm-set z i A* = (*x* = *z* ∨ *x* ∈ *A* ∨ *x* ∈ *minus-set i A*)
⟨*proof*⟩

lemma (**in** *Ring*) *in-s-set*:*x* ∈ *s-set R s A* ⟹ (∃ *r* ∈ *carrier R*. ∃ *a* ∈ *A*.
$x = r\ {}_s\cdot\ a$) ∨ *x* ∈ *A*
⟨*proof*⟩

lemma (**in** *Ring*) *sop-closedTr0*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);
sop-one R s (*aug-pm-set z i* (*s-set R s A*)); *sop-z R s z*;
r ∈ *carrier R*; *x* ∈ *aug-pm-set z i* (*s-set R s A*)⟧ ⟹
$r\ {}_s\cdot\ x$ ∈ *aug-pm-set z i* (*s-set R s A*)
⟨*proof*⟩

lemma (**in** *Ring*) *sop-closedTr*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);
sop-one R s (*aug-pm-set z i* (*s-set R s A*)); *sop-z R s z*⟧ $\Longrightarrow$
$\forall r\in carrier\ R.\ \forall x\in add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))\ n.$
$r\ {}_s\cdot\ x \in add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *sop-closed*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);
sop-one R s (*aug-pm-set z i* (*s-set R s A*)); *sop-z R s z*⟧ $\Longrightarrow$
$\forall r\in carrier\ R.\ \forall x\in addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)).$
$r\ {}_s\cdot\ x \in addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))$
⟨*proof*⟩

lemma (**in** *Ring*) *sop-oneTr*:⟦*commute-bpp f* (*aug-pm-set z i* (*s-set R s A*));
assoc-bpp (*aug-pm-set z i* (*s-set R s A*)) *f*;
ipp-cond1 (*s-set R s A*) *i*; *ipp-cond2 z* (*s-set R s A*) *i f*;
ipp-cond3 z i; *inv-ipp z i f* (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-one R s (*aug-pm-set z i* (*s-set R s A*))⟧ $\Longrightarrow$
$\forall x\in add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))\ n.\ (1_r)\ {}_s\cdot\ x = x$
⟨*proof*⟩

lemma (**in** *Ring*) *sop-one*:⟦*commute-bpp f* (*aug-pm-set z i* (*s-set R s A*));
assoc-bpp (*aug-pm-set z i* (*s-set R s A*)) *f*; *ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-one R s (*aug-pm-set z i* (*s-set R s A*))⟧ $\Longrightarrow$
$\forall x\in addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)).\ (1_r)\ {}_s\cdot\ x = x$
⟨*proof*⟩

lemma (**in** *Ring*) *sop-assocTr*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);

sop-one R s (aug-pm-set z i (s-set R s A)); sop-z R s z⟧ ⟹
∀ *a*∈*carrier R*. ∀ *b*∈*carrier R*.
∀ *x*∈*add-set f (aug-pm-set z i (s-set R s A)) n*.
a $_{s}\cdot$ (*b* $_{s}\cdot$ *x*) = (*a* $\cdot_{r}$ *b*) $_{s}\cdot$ *x*
⟨*proof*⟩

lemma (**in** *Ring*) *sop-assoc*:⟦*ipp-cond1 (s-set R s A) i*;
ipp-cond2 z (s-set R s A) i f; *ipp-cond3 z i*;
inv-ipp z i f (s-set R s A); *zeroA z i f (s-set R s A) z*;
sop-distr2 R s f i (s-set R s A) z;
sop-assoc R s (aug-pm-set z i (s-set R s A));
sop-inv R s i (s-set R s A); *sop-z R s z*;
sop-one R s (aug-pm-set z i (s-set R s A))⟧ ⟹
∀ *a*∈*carrier R*. ∀ *b*∈*carrier R*.
∀ *x*∈*addition-set f (aug-pm-set z i (s-set R s A))*.
a $_{s}\cdot$ (*b* $_{s}\cdot$ *x*) = (*a* $\cdot_{r}$ *b*) $_{s}\cdot$ *x*
⟨*proof*⟩

lemma (**in** *Ring*) *s-set-commute*:⟦*commute-bpp f (aug-pm-set z i (s-set R s A))*;
x ∈ *addition-set f (aug-pm-set z i (s-set R s A))*;
y ∈ *addition-set f (aug-pm-set z i (s-set R s A))*⟧ ⟹
x $_{f}+$ *y* = *y* $_{f}+$ *x*
⟨*proof*⟩

lemma (**in** *Ring*) *add-s-set-inc-add-set*:
add-set f (aug-pm-set z i A) n ⊆
add-set f (aug-pm-set z i (s-set R s A)) n
⟨*proof*⟩

lemma (**in** *Ring*) *sop-distr1Tr*:⟦*commute-bpp f (aug-pm-set z i (s-set R s A))*;
assoc-bpp (aug-pm-set z i (s-set R s A)) f; *ipp-cond1 (s-set R s A) i*;
ipp-cond2 z (s-set R s A) i f; *ipp-cond3 z i*;
inv-ipp z i f (s-set R s A); *zeroA z i f (s-set R s A) z*;
sop-distr1 R s f i (s-set R s A) z;
sop-distr2 R s f i (s-set R s A) z;
sop-assoc R s (aug-pm-set z i (s-set R s A));
sop-inv R s i (s-set R s A);
sop-one R s (aug-pm-set z i (s-set R s A)); *sop-z R s z*⟧ ⟹
∀ *a*∈*carrier R*. ∀ *b*∈*carrier R*. ∀ *x*∈ *add-set f (aug-pm-set z i (s-set R s A)) n*.
(*a* ± *b*) $_{s}\cdot$ *x* = *a* $_{s}\cdot$ *x* $_{f}+$ (*b* $_{s}\cdot$ *x*)
⟨*proof*⟩

lemma (**in** *Ring*) *sop-distr1* :⟦*commute-bpp f (aug-pm-set z i (s-set R s A))*;
assoc-bpp (aug-pm-set z i (s-set R s A)) f; *ipp-cond1 (s-set R s A) i*;
ipp-cond2 z (s-set R s A) i f; *ipp-cond3 z i*;
inv-ipp z i f (s-set R s A); *zeroA z i f (s-set R s A) z*;
sop-distr1 R s f i (s-set R s A) z;
sop-distr2 R s f i (s-set R s A) z;
sop-assoc R s (aug-pm-set z i (s-set R s A));

sop-inv R s i (s-set R s A);
sop-one R s (aug-pm-set z i (s-set R s A)); sop-z R s z$\rrbracket \Longrightarrow$
$\forall$ *a*$\in$*carrier R.* $\forall$ *b*$\in$*carrier R.*
$\forall$ *x*$\in$ *addition-set f (aug-pm-set z i (s-set R s A)).*
$(a \pm b)\ {}_s\cdot\ x = a\ {}_s\cdot\ x\ {}_f+ (b\ {}_s\cdot\ x)$
⟨*proof*⟩

definition
fgmodule-condition ::[(′*r*, ′*m*) *Ring-scheme*, ′*a* ⇒ ′*a* ⇒ ′*a*, ′*a* ⇒ ′*a*,
′*r* ⇒ ′*a* ⇒ ′*a*, ′*a set*, ′*a*] ⇒ *bool* **where**
fgmodule-condition R f i s A z ⟷
commute-bpp f (aug-pm-set z i (s-set R s A)) ∧
assoc-bpp (aug-pm-set z i (s-set R s A)) f ∧
ipp-cond1 (s-set R s A) i ∧ *ipp-cond2 z (s-set R s A) i f* ∧
ipp-cond3 z i ∧ *inv-ipp z i f (s-set R s A)* ∧
zeroA z i f (s-set R s A) z ∧ *sop-distr1 R s f i (s-set R s A) z* ∧
sop-distr2 R s f i (s-set R s A) z ∧
sop-assoc R s (aug-pm-set z i (s-set R s A)) ∧
sop-inv R s i (s-set R s A) ∧
sop-one R s (aug-pm-set z i (s-set R s A)) ∧ *sop-z R s z*

lemma (in *Ring*) *sop-closed1*:⟦*fgmodule-condition R f i s A z*; *r* ∈ *carrier R*;
x ∈ *addition-set f (aug-pm-set z i (s-set R s A))*⟧ ⟹
$r\ {}_s\cdot\ x$ ∈ *addition-set f (aug-pm-set z i (s-set R s A))*
⟨*proof*⟩

lemma (in *Ring*) *fgmodule-is-module*:*fgmodule-condition R f i s A z*
⟹ *R module (fgmodule R A z i f s)*
⟨*proof*⟩

lemma (in *Ring*) *a-in-carr-fgmodule*:*a* ∈ *A*
⟹ *a* ∈ *carrier (fgmodule R A z i f s)*
⟨*proof*⟩

6.5 A fgmodule and a free module

lemma (in *Ring*) *fg-zeroTr*:⟦*fgmodule-condition R f i s A z*; *a* ∈ *A*⟧ ⟹
$\mathbf{0}\ {}_s\cdot\ a = z$
⟨*proof*⟩

lemma (in *Ring*) *fg-genTr0*:⟦*fgmodule-condition R f i s A z*;
x ∈ *aug-pm-set z i (s-set R s A)*⟧ ⟹
x ∈ *linear-span R (fgmodule R A z i f s) (carrier R) A*
⟨*proof*⟩

lemma (in *Ring*) *fg-genTr*:*fgmodule-condition R f i s A z* ⟹
∀ *x*. *x* ∈ (*add-set f (aug-pm-set z i (s-set R s A)) n*) ⟶
x ∈ *linear-span R (fgmodule R A z i f s) (carrier R) A*
⟨*proof*⟩

lemma (**in** *Ring*) *generator-of-fgm*:*fgmodule-condition R f i s A z* $\Longrightarrow$
generator R (*fgmodule R A z i f s*) *A*
⟨*proof*⟩

lemma (**in** *Ring*) *fg-freeTr1*:⟦*R module M*; *free-generator R M A*;
R module fgmodule R A z i f s; *free-generator R* (*fgmodule R A z i f s*) *A*;
$g \in$ *mHom R M* (*fgmodule R A z i f s*); $\forall x \in A.\ g\ x = x$⟧ $\Longrightarrow$
$\forall$*fa sa*. *fa* $\in \{j.\ j \le (n{::}nat)\} \rightarrow A \wedge$ *sa* $\in \{j.\ j \le n\} \rightarrow$ *carrier R* $\longrightarrow$
l-comb R (*fgmodule R A z i f s*) *n sa* (*cmp g fa*) =
l-comb R (*fgmodule R A z i f s*) *n sa fa*
⟨*proof*⟩

lemma (**in** *Ring*) *fg-freeTr*:⟦*R module M*; *free-generator R M A*;
R module fgmodule R A z i f s;
free-generator R (*fgmodule R A z i f s*) *A*;
$g \in$ *mHom R M* (*fgmodule R A z i f s*); $\forall x \in A.\ g\ x = x$;
fa $\in \{j.\ j \le (n{::}nat)\} \rightarrow A$; *sa* $\in \{j.\ j \le n\} \rightarrow$ *carrier R*⟧ $\Longrightarrow$
l-comb R (*fgmodule R A z i f s*) *n sa* (*cmp g fa*) =
l-comb R (*fgmodule R A z i f s*) *n sa fa*
⟨*proof*⟩

lemma (**in** *Ring*) *fg-free1*:⟦ $A \neq \{\}$; *fgmodule-condition R f i s A z*;
free-generator R (*fgmodule R A z i f s*) *A*; *R module M*;
free-generator R M A⟧ $\Longrightarrow M \cong_R$ (*fgmodule R A z i f s*)
⟨*proof*⟩

lemma (**in** *Ring*) *fg-free*:⟦*fgmodule-condition R f i s A z*;
free-generator R (*fgmodule R A z i f s*) *A*; *R module M*;
free-generator R M A⟧ $\Longrightarrow M \cong_R$ (*fgmodule R A z i f s*)
⟨*proof*⟩

6.6 Direct sum, again

definition
miota :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*] ⇒ '*a* ⇒ '*a* **where**
miota R M1 M = ($\lambda x \in$*carrier M1*. *x*)

definition
msubmodule ::[('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*] ⇒ *bool* **where**
msubmodule R M M1 $\longleftrightarrow$ *miota R M1 M* $\in$ *mHom R M1 M* $\wedge$
(*carrier M1*) $\subseteq$ (*carrier M*)

definition
ds2 :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*] ⇒ *bool* **where**

ds2 R M M1 M2 $\longleftrightarrow$ *R module M* $\wedge$ *msubmodule R M M1* $\wedge$ *msubmodule R M M2* $\wedge$
$(\forall x \in carrier\ M.\ \exists m1 \in carrier\ M1.\ \exists m2 \in carrier\ M2.\ x = m1 \pm_M m2) \wedge$

$(carrier\ M1) \cap (carrier\ M2) = \{\mathbf{0}_M\}$

abbreviation
DS2 ((4-/ $\bigoplus$ -,- -) [92,93,92,92]92) **where**
$M1 \bigoplus_{R,M} M2$ == *ds2 R M M1 M2*

lemma (**in** *Ring*) *ds2-commute*:⟦*R module M1*; *R module M2*; *R module M*;
$M1 \bigoplus_{R,M} M2$⟧ $\Longrightarrow$ $M2 \bigoplus_{R,M} M1$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-addition*:⟦*R module M*; *R module M1*; *msubmodule R M M1*;
$x \in carrier\ M1$; $y \in carrier\ M1$⟧ $\Longrightarrow x \pm_{M1} y = x \pm_M y$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-mOp*:⟦*R module M*; *R module M1*; *msubmodule R M M1*;
$x \in carrier\ M1$⟧ $\Longrightarrow -_{a M1}\ x = -_{a M}\ x$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-sprod*:⟦*R module M*; *R module M1*; *msubmodule R M M1*;
$a \in carrier\ R$; $x \in carrier\ M1$⟧ $\Longrightarrow a \cdot_{s M1} x = a \cdot_{s M} x$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-submodule*:⟦*R module M*; *R module M1*; *msubmodule R M M1*⟧
$\Longrightarrow$ *submodule R M* (*carrier M1*)
⟨*proof*⟩

lemma (**in** *Ring*) *ds2-unique*:⟦*R module M*; *R module M1*; *R module M2*;
ds2 R M M1 M2; $m1 \in carrier\ M1$; $m1' \in carrier\ M1$;
$m2 \in carrier\ M2$; $m2' \in carrier\ M2$;
$m1 \pm_M m2 = m1' \pm_M m2'$⟧ $\Longrightarrow m1 = m1' \wedge m2 = m2'$
⟨*proof*⟩

lemma (**in** *Ring*) *miota-injec*:⟦*R module M*; *R module M1*; *R module M2*;
ds2 R M M1 M2; *msubmodule R M M1*⟧ $\Longrightarrow$
miota R M1 M $\in$ *mHom R M1 M* $\wedge$ $injec_{M1,M}$ (*miota R M1 M*)
⟨*proof*⟩

definition
mproj1 :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('a, 'r, 'm1) *Module-scheme*, ('a, 'r, 'm1) *Module-scheme*] ⇒ 'a ⇒ 'a **where**
mproj1 R M1 M2 M = ($\lambda x \in carrier\ M.$ *THE x1. x1* $\in$ *carrier M1* $\wedge$
$(x \pm_M (-_{a M}\ x1)) \in carrier\ M2$)

definition
mproj2 :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('a, 'r, 'm1) *Module-scheme*, ('a, 'r, 'm1) *Module-scheme*] ⇒ 'a ⇒ 'a **where**
mproj2 R M1 M2 M = *mproj1 R M2 M1 M*

lemma (**in** *Ring*) *ds2-components*:⟦*R module M1*; *R module M2*; *R module M*;
$M1 \bigoplus_{R,M} M2$; *a* ∈ *carrier M*⟧ ⟹
∃ *a1*∈*carrier M1*. ∃ *a2*∈*carrier M2*. $a = a1 \pm_M a2$
⟨*proof*⟩

lemma (**in** *Ring*) *ds2-components1*:⟦*R module M1*; *R module M2*; *R module M*;
$M1 \bigoplus_{R,M} M2$; *a* ∈ *carrier M*⟧ ⟹
∃ *a1*∈*carrier M1*. $a \pm_M -_{aM}\ a1$ ∈ *carrier M2*
⟨*proof*⟩

lemma (**in** *Ring*) *mprojTr1*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*;
x ∈ *carrier M* ⟧ ⟹ ∃!*x1*. *x1* ∈ *carrier M1* ∧ ($x \pm_M -_{aM}\ x1$) ∈ *carrier M2*
⟨*proof*⟩

lemma (**in** *Ring*) *mprojTr2*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*;
x ∈ *carrier M*; *x1* ∈ *carrier M1*; ($x \pm_M (-_{aM}\ x1)$) ∈ *carrier M2*;
y1 ∈ *carrier M1*;($x \pm_M (-_{aM}\ y1)$) ∈ *carrier M2* ⟧ ⟹ *x1* = *y1*
⟨*proof*⟩

lemma (**in** *Ring*) *mprojTr3*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*;
a ∈ *carrier M*; *a1* ∈ *carrier M1*; ($a \pm_M (-_{aM}\ a1)$) ∈ *carrier M2*⟧ ⟹
(*THE x1*. *x1* ∈ *carrier M1* ∧ $a \pm_M -_{aM}\ x1$ ∈ *carrier M2*) = *a1*
⟨*proof*⟩

lemma (**in** *Ring*) *mproj*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*⟧
⟹ *mproj1 R M1 M2 M* ∈ *mHom R M M1*
⟨*proof*⟩

lemma (**in** *Ring*) *mproj2*:⟦*R module M1*; *R module M2*; *R module M*; $M1 \bigoplus_{R,M} M2$⟧
⟹ *mproj2 R M1 M2 M* ∈ *mHom R M M2*
⟨*proof*⟩

6.6.1 Existence of the tensor product

definition
fm-gen-by-prod :: [('r, 'm) *Ring-scheme*, (('a ∗ 'b), 'r, 'm1) *Module-scheme*,

('a, 'r, 'm1) *Module-scheme*, ('b, 'r, 'm1) *Module-scheme*] ⇒ *bool*
((4FM_/ - - -) [100,100,101]100) **where**
FM_R *P M N* ⟷ *R module P* ∧ *free-generator R P* (*M* $\times_c$ *N*)

lemma (**in** *Ring*) *free-gen-gen*:FM_R *P M N* ⟹ *generator R P* (*M* $\times_c$ *N*)
⟨*proof*⟩

lemma (**in** *Ring*) *free-gen-mem*:⟦FM_R *P M N*; *a* ∈ (*M* $\times_c$ *N*)⟧ ⟹ *a* ∈ *carrier P*
⟨*proof*⟩

lemma (**in** *Ring*) *mHom-lin-nsumTr*:⟦*R module M*; *R module N*; *t* ∈ *mHom R M N*⟧ ⟹
f ∈ {*j*. *j* ≤ (*n*::*nat*)} → *carrier M* ⟶ *t* (*nsum M f n*) = *nsum N* (*cmp t f*) *n*
⟨*proof*⟩

lemma (**in** *Ring*) *mHom-lin-nsum*:⟦*R module M*; *R module N*; *t* ∈ *mHom R M N*;
f ∈ {*j*. *j* ≤ (*n*::*nat*)} → *carrier M*⟧ ⟹
t (*nsum M f n*) = *nsum N* (*cmp t f*) *n*
⟨*proof*⟩

lemma (**in** *Ring*) *module-over-zeroring*:⟦*zeroring R*; *R module M*⟧ ⟹
carrier M = {$\mathbf{0}_M$}
⟨*proof*⟩

lemma (**in** *Ring*) *submodule-over-zeroring*:⟦*zeroring R*; *R module M*;
submodule R M N⟧ ⟹ *N* = {$\mathbf{0}_M$}
⟨*proof*⟩

definition
Least-submodule :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
'a *set*] ⇒ 'a *set*
((3LSM_/ -/ -) [100,100,101]100) **where**
LSM_R *M T* = ⋂{*N*. *submodule R M N* ∧ *T* ⊆ *N*}

lemma (**in** *Ring*) *LSM-mem*:⟦*R module M*; *T* ⊆ *carrier M*; *t* ∈ *T*⟧ ⟹
t ∈ (LSM_R *M T*)
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-sub-M*:⟦*R module M*; *T* ⊆ *carrier M*⟧ ⟹
(LSM_R *M T*) ⊆ *carrier M*
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-sub-submodule*:⟦*R module M*; *T* ⊆ *carrier M*;
submodule R M N; *T* ⊆ *N* ⟧ ⟹ (LSM_R *M T*) ⊆ *N*
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-inc-T*:$[\![R\ module\ M;\ T \subseteq carrier\ M]\!] \Longrightarrow T \subseteq (LSM_R\ M\ T)$
$\langle proof \rangle$

lemma (**in** *Ring*) *LSM-submodule*:$[\![R\ module\ M;\ T \subseteq carrier\ M]\!] \Longrightarrow$
$submodule\ R\ M\ (LSM_R\ M\ T)$
$\langle proof \rangle$

lemma (**in** *Ring*) *linear-comb-memTr*:$[\![R\ module\ M;\ submodule\ R\ M\ N;\ T \subseteq N]\!] \Longrightarrow$
$\forall f\ s.\ f \in \{j.\ j \leq (n{::}nat)\} \to T \wedge s \in \{j.\ j \leq n\} \to carrier\ R \longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ f \in N$
$\langle proof \rangle$

lemma (**in** *Ring*) *linear-comb-mem*:$[\![R\ module\ M;\ submodule\ R\ M\ N;\ T \subseteq N;$
$f \in \{j.\ j \leq (n{::}nat)\} \to T;\ s \in \{j.\ j \leq n\} \to carrier\ R]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ f \in N$
$\langle proof \rangle$

lemma (**in** *Ring*) *LSM-eq-linear-span*:$[\![R\ module\ M;\ T \subseteq carrier\ M]\!] \Longrightarrow$
$(LSM_R\ M\ T) = linear\text{-}span\ R\ M\ (carrier\ R)\ T$
$\langle proof \rangle$

lemma (**in** *Ring*) *LSM-sub-ker*:$[\![R\ module\ M;\ R\ module\ N;\ T \subseteq carrier\ M;$
$f \in mHom\ R\ M\ N;\ T \subseteq ker_{M,N}\ f]\!] \Longrightarrow LSM_R\ M\ T \subseteq ker_{M,N}\ f$
$\langle proof \rangle$

definition
tensor-relations1 :: $[('r,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'r,\ 'm1)\ Module\text{-}scheme,$
$('b,\ 'r,\ 'm1)\ Module\text{-}scheme,\ (('a * 'b),\ 'r,\ 'm1)\ Module\text{-}scheme] \Rightarrow$
$('a * 'b)\ set$
((*4TR1*/ -/ -/ -/ -) [*100*,*100*,*100*,*101*]*100*) **where**
$TR1\ R\ M\ N\ MN = \{x.\ \exists m1{\in}carrier\ M.\ \exists m2{\in}carrier\ M.\ \exists n{\in}carrier\ N.$
$x = (m1 \pm_M m2,\ n) \pm_{MN} (-_{a\,MN}\ ((m1,\ n) \pm_{MN} (m2,\ n)))\}$

definition
tensor-relations2 :: $[('r,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'r,\ 'm1)\ Module\text{-}scheme,$
$('b,\ 'r,\ 'm1)\ Module\text{-}scheme,\ (('a * 'b),\ 'r,\ 'm1)\ Module\text{-}scheme] \Rightarrow$
$('a * 'b)\ set$
((*4TR2*/ -/ -/ -/ -) [*100*,*100*,*100*, *101*]*100*) **where**
$TR2\ R\ M\ N\ MN = \{x.\ \exists m{\in}carrier\ M.\ \exists n1{\in}carrier\ N.\ \exists n2{\in}carrier\ N.$
$x = (m,\ n1 \pm_N n2) \pm_{MN} (-_{a\,MN}\ ((m,\ n1) \pm_{MN} (m,\ n2)))\}$

definition
tensor-relations3 :: $[('r,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'r,\ 'm1)\ Module\text{-}scheme,$
$('b,\ 'r,\ 'm1)\ Module\text{-}scheme,\ (('a * 'b),\ 'r,\ 'm1)\ Module\text{-}scheme] \Rightarrow$
$('a * 'b\ )\ set$
((*4TR3*/ -/ -/ -/ -) [*100*,*100*,*100*,*101*]*100*) **where**

$TR3\ R\ M\ N\ P = \{x.\ \exists\, m{\in}carrier\ M.\ \exists\, n{\in}carrier\ N.\ \exists\ \ a{\in}carrier\ R.$
$x = (a \cdot_{sM}\ m,\ n) \pm_P (-_{aP}\ (a \cdot_{sP}\ (m,\ n)))\}$

definition
tensor-relations4 :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('b, 'r, 'm1) *Module-scheme*, (('a * 'b), 'r, 'm1) *Module-scheme*] ⇒
('a * 'b) *set*
((4TR4/ -/ -/ -/ -) [100,100,100,101]100) **where**
$TR4\ R\ M\ N\ MN = \{x.\ \exists\, m{\in}carrier\ M.\ \exists\, n{\in}carrier\ N.\ \exists\ \ a{\in}carrier\ R.$
$x = (m,\ a \cdot_{sN}\ n) \pm_{MN} (-_{aMN}\ (a \cdot_{s\ MN}\ (m,\ n)))\}$

definition
tensor-relations :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('b, 'r, 'm1) *Module-scheme*, (('a * 'b), 'r, 'm1) *Module-scheme*] ⇒
('a * 'b) *set*
((4TR- -/ -/ -) [100,100,101]100) **where**
$TR_R\ M\ N\ MN = LSM_R\ MN\ ((TR1\ R\ M\ N\ MN) \cup (TR2\ R\ M\ N\ MN) \cup$
$(TR3\ R\ M\ N\ MN) \cup (TR4\ R\ M\ N\ MN))$

definition
tensor-product :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('b, 'r, 'm1) *Module-scheme*, (('a * 'b), 'r, 'm1) *Module-scheme*] ⇒
(('a * 'b) *set*, 'r) *Module* **where**
$tensor\text{-}product\ R\ M\ N\ MN = MN\ /_m\ (TR_R\ M\ N\ MN)$

abbreviation
TENSORPROD ((4-/ -⊗-/ -) [92,92,92,93]92) **where**
$M\ {}_P{\bigotimes}_R\ N == tensor\text{-}product\ R\ M\ N\ P$

lemma (**in** *Ring*) *mem-cartesian*:⟦R *module* M; R *module* N; $m \in carrier\ M$;
$n \in carrier\ N$ ⟧ $\Longrightarrow (m,\ n) \in M \times_c N$
⟨*proof*⟩

lemma (**in** *Ring*) *cartesianTr*:⟦R *module* M; R *module* N; $x \in M \times_c N$⟧ $\Longrightarrow$
$\exists\, m\ n.\ m{\in}carrier\ M \wedge n \in carrier\ N \wedge x = (m,\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *free-module-mem*:⟦R *module* M; R *module* N; $m \in carrier\ M$;
$n \in carrier\ N$; $FM_R\ P\ M\ N$⟧ $\Longrightarrow\ (m,\ n) \in carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *FM-P-module*:⟦R *module* M; R *module* N; $FM_R\ P\ M\ N$⟧
$\Longrightarrow R$ *module* P
⟨*proof*⟩

lemma (**in** *Ring*) *TR1-sub-carr*:⟦R *module* M; R *module* N; $FM_R\ P\ M\ N$⟧ $\Longrightarrow$
$(TR1\ R\ M\ N\ P) \subseteq carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR2-sub-carr*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N]\!] \Longrightarrow$
$(TR2\ R\ M\ N\ P) \subseteq carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR3-sub-carr*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N]\!] \Longrightarrow$
$(TR3\ R\ M\ N\ P) \subseteq carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR4-sub-carr*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N]\!] \Longrightarrow$
$(TR4\ R\ M\ N\ P) \subseteq carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR-sub-carr*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N]\!] \Longrightarrow$
$(TR1\ R\ M\ N\ P) \cup (TR2\ R\ M\ N\ P) \cup (TR3\ R\ M\ N\ P) \cup (TR4\ R\ M\ N\ P) \subseteq$
$carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR-submodule*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N]\!] \Longrightarrow$
$submodule\ R\ P\ (TR_R\ M\ N\ P)$
⟨*proof*⟩

lemma (**in** *Ring*) *TR-cont-TR1234*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N]\!]$
$\Longrightarrow$
$TR1\ R\ M\ N\ P \cup TR2\ R\ M\ N\ P \cup TR3\ R\ M\ N\ P \cup TR4\ R\ M\ N\ P \subseteq TR_R\ M$
$N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR1-mem*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N;\ m1 \in$
$carrier\ M;$
$m2 \in carrier\ M;\ n \in carrier\ N]\!] \Longrightarrow (m1 \pm_M m2,\ n) \pm_P -_{aP} ((m1,\ n) \pm_P (m2,$
$n))$
$\in TR_R\ M\ N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR2-mem*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N;\ m \in$
$carrier\ M;$
$n1 \in carrier\ N;\ n2 \in carrier\ N\ ]\!] \Longrightarrow$
$(m,\ n1 \pm_N n2) \pm_P -_{aP} ((m,\ n1) \pm_P (m,\ n2)) \in TR_R\ M\ N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR3-mem*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N;\ m \in$
$carrier\ M;$
$n \in carrier\ N;\ a \in carrier\ R]\!] \Longrightarrow$
$(a \cdot_{sM} m,\ n) \pm_P -_{aP} (a \cdot_{sP} (m,\ n)) \in TR_R\ M\ N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR4-mem*:$[\![R\ module\ M;\ R\ module\ N;\ FM_R\ P\ M\ N;\ m \in$
$carrier\ M;$
$n \in carrier\ N;\ a \in carrier\ R]\!] \Longrightarrow$

$(m,\ a \cdot_{sN} n) \pm_P -_{aP} (a \cdot_{sP} (m, n)) \in TR_R\ M\ N\ P$
⟨proof⟩

lemma (**in** *Ring*) *tensor-product-module*:⟦*R module M*; *R module N*; *FM* $_R$ *P M N*⟧ ⟹
R module (*tensor-product R M N P*)
⟨proof⟩

lemma (**in** *Ring*) *tau-mpj-bilin1*:⟦ *R module M*; *R module N*; *FM* $_R$ *P M N*;
x1 ∈ *carrier M*; *x2* ∈ *carrier M*; *y* ∈ *carrier N*⟧ ⟹
(*mpj P* (TR_R *M N P*)) (*x1* $\pm_M$ *x2*, *y*) =
(*mpj P* (TR_R *M N P*)) (*x1*, *y*) $\pm_{(M\ {}_P\otimes_R\ N)}$ (*mpj P* (TR_R *M N P*) (*x2*, *y*))
⟨proof⟩

lemma (**in** *Ring*) *tau-mpj-bilin2*:⟦*R module M*; *R module N*; *FM* $_R$ *P M N*;
m ∈ *carrier M*; *n1* ∈ *carrier N*; *n2* ∈ *carrier N*⟧ ⟹
(*mpj P* (TR_R *M N P*)) (*m*, *n1* $\pm_N$ *n2*) =
(*mpj P* (TR_R *M N P*)) (*m*, *n1*) $\pm_{(M\ {}_P\otimes_R\ N)}$ (*mpj P* (TR_R *M N P*) (*m*, *n2*))
⟨proof⟩

lemma (**in** *Ring*) *tau-mpj-bilin3*:⟦*R module M*; *R module N*; *FM* $_R$ *P M N*;
m ∈ *carrier M*; *n* ∈ *carrier N*; *a* ∈ *carrier R*⟧ ⟹
(*mpj P* (TR_R *M N P*)) ($a \cdot_{sM}$ *m*, *n*) = $a \cdot_{s(M\ {}_P\otimes_R\ N)}$
(*mpj P* (TR_R *M N P*) (*m*, *n*))
⟨proof⟩

lemma (**in** *Ring*) *tau-mpj-bilin4*:⟦*R module M*; *R module N*; *FM* $_R$ *P M N*;
m ∈ *carrier M*; *n* ∈ *carrier N*; *a* ∈ *carrier R*⟧ ⟹
(*mpj P* (TR_R *M N P*)) (*m*, $a \cdot_{sN}$ *n*) = $a \cdot_{s(M\ {}_P\otimes_R\ N)}$
(*mpj P* (TR_R *M N P*) (*m*, *n*))
⟨proof⟩

definition
tau :: [(′*r*, ′*m*) *Ring-scheme*, (′*a*, ′*r*, ′*m1*) *Module-scheme*,
(′*b*, ′*r*, ′*m1*) *Module-scheme*, ((′*a* ∗ ′*b*), ′*r*, ′*m1*) *Module-scheme*] ⇒
(′*a* ∗ ′*b*) ⇒ (′*a* ∗ ′*b*) **where**
tau R M N P = ($\lambda x \in (M \times_c N).\ x$)

lemma (**in** *Ring*) *tau-func*:⟦*R module M*; *R module N*; *FM* $_R$ *P M N*⟧ ⟹
tau R M N P ∈ $M \times_c N$ → *carrier P*
⟨proof⟩

lemma (**in** *Ring*) *tau-mem*:⟦*R module M*; *R module N*; *m* ∈ *carrier M*;
n ∈ *carrier N*; *FM* $_R$ *P M N*⟧ ⟹ *tau R M N P* (*m*, *n*) ∈ *carrier P*
⟨proof⟩

lemma (**in** *Ring*) *tau-inj0*:⟦¬ *zeroring R*; *R module M*; *R module N*; *FM* $_R$ *P M*

$N \rrbracket$
$\Longrightarrow$ *inj-on* (*tau R M N P*) ($M \times_c N$)
⟨*proof*⟩

lemma (**in** *Ring*) *tau-inj1*:⟦*zeroring R*; *R module M*; *R module N*; FM_R *P M N*⟧
$\Longrightarrow$
inj-on (*tau R M N P*) ($M \times_c N$)
⟨*proof*⟩

lemma (**in** *Ring*) *tau-inj*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ $\Longrightarrow$
inj-on (*tau R M N P*) ($M \times_c N$)
⟨*proof*⟩

lemma (**in** *Ring*) *tau-mpj-bilinear*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ $\Longrightarrow$

bilinear-map (*compose* ($M \times_c N$) (*mpj P* (TR_R *M N P*)) (*tau R M N P*))
R M N ($M \; {}_P\bigotimes_R \; N$)
⟨*proof*⟩

definition
tnm :: [(′*r*, ′*m*) *Ring-scheme*, ((′*a* ∗ ′*b*), ′*r*, ′*m1*) *Module-scheme*,
(′*a*, ′*r*, ′*m1*) *Module-scheme*, (′*b*, ′*r*, ′*m1*) *Module-scheme*] ⇒
(′*a* ∗ ′*b*) ⇒ (′*a* ∗ ′*b*) *set* **where**
tnm R P M N = *compose* ($M \times_c N$) (*mpj P* (TR_R *M N P*)) (*tau R M N P*)

lemma (**in** *Ring*) *tnm-bilinear*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ $\Longrightarrow$
bilinear-map (*tnm R P M N*) *R M N* ($M \; {}_P\bigotimes_R \; N$)
⟨*proof*⟩

lemma (**in** *Ring*) *tnm-mem*:⟦ *R module M*; *R module N*; FM_R *P M N*; $m \in$ *carrier M*;
$n \in$ *carrier N*⟧ $\Longrightarrow$ *tnm R P M N* (*m*, *n*) ∈ *carrier* ($M \; {}_P\bigotimes_R \; N$)
⟨*proof*⟩

definition
tensor-elem :: [(′*r*, ′*m*) *Ring-scheme*, ((′*a* ∗ ′*b*), ′*r*, ′*m1*) *Module-scheme*,
(′*a*, ′*r*, ′*m1*) *Module-scheme*, (′*b*, ′*r*, ′*m1*) *Module-scheme*] ⇒ ′*a* ⇒ ′*b*
⇒ (′*a* ∗ ′*b*) *set* **where**
tensor-elem R P M N m n = *tnm R P M N* (*m*, *n*)

abbreviation
TNSELEM ((6- -,-⊗-,-/ -) [*100*,*100*,*100*,*100*,*100*,*101*]*101*) **where**
$m \; {}_{R,P}{\otimes}_{M,N} \; n$ == *tensor-elem R P M N m n*

lemma (**in** *Ring*) *tensor-univ-propTr*:⟦*R module M*; *R module N*; FM_R *P M N*;
R module Z; *bilinear-map f R M N Z*⟧ $\Longrightarrow$
∃ *g*. *g* ∈ *mHom R P Z* ∧ (*compose* ($M \times_c N$) *g* (*tau R M N P*)) = *f*
⟨*proof*⟩

lemma (**in** *Ring*) *tensor-univ-propTr1*:⟦*R module M*; *R module N*; FM_R *P M N*;

R module Z; *bilinear-map f R M N Z*⟧ ⟹
∃!*g*. *g*∈(*mHom R* (*M* ${}_P\bigotimes{}_R$ *N*) *Z*) ∧ (*compose* (*M* $\times_c$ *N*) *g* (*tnm R P M N*)) = *f*
⟨*proof*⟩

lemma (**in** *Ring*) *tensor-universal-property*:⟦*R module M*; *R module N*; FM_R *P M N* ⟧
⟹ *universal-property R M N* (*M* ${}_P\bigotimes{}_R$ *N*) (*tnm R P M N*)
⟨*proof*⟩

end