Index

`||`, 281
`|`, 39, 93
`!!`, 15, 285
`$!`, 150, 175, 223, 287
`$`, 170
`&&`, 40, 281
`()`, 26, 124, 185, 284
`*`, 34, 280
`++`, 16, 62, 286
`+`, 34, 280
`->`, 27, 287
`-`, 34, 280
`.`, 81, 287
`/=`, 31, 280
`/`, 35, 54, 281
`::`, 22
`:`, 41, 285
`<|>`, 181, 291
`<$>`, 163, 289
`<*>`, 159, 289
`<-`, 47, 126
`<=`, 31, 280
`<>`, 294
`<`, 31, 280
`==`, 31, 280
`=>`, 30
`>=`, 31, 280
`>>=`, 166, 290
`>`, 31, 280
`[]`, 25, 113, 178, 285
`\`, 42, 81, 215
`^`, 15, 284
`_`, 40, 48, 61, 70, 71

`abs`, 34, 39, 280
absolute value, *see* `abs`
accumulation, 5, 79, 224, 240
actions, 12, 124
addition, *see* `+`, `sum`
`All`, 199, 293
`all`, 76, 205, 296
alpha-beta pruning, 151
`Alternative`, 181, 184, 291
alternatives, 181, 290
  `Maybe`, 182, 291
  `Parser`, 182
  laws, 182
  list, 291
`and`, 77, 80, 205, 296
`Any`, 199, 293
`any`, 76, 205, 296
append, *see* `++`
application, 3, 16, 22, 28, 170, 223
  partial, 28, 74, 77
  strict, *see* `$!`
`Applicative`, 159, 289
applicative style, 159, 163, 180
applicatives, 7, 157, 289
  `IO`, 161, 290
  `Maybe`, 159, 289
  `Parser`, 180
  function, 175
  laws, 162
  list, 160, 289
  state, 170
assignment, 4, 213
associativity, 15, 79, 228
  for addition, 80, 187, 228, 233
  for alternatives, 182
  for append, 236
  for application, 28
  for applicatives, 162, 163
  for composition, 81
  for cons, 42
  for function types, 28
  for grammars, 189
  for monads, 174
  for monoids, 196
  for multiplication, 187

binary trees, 97, 155, 171, 175, 200
bind, 165
`Bool`, 22, 24, 281
Booleans, 281
buffering, 150

`case`, 88, 164, 179, 262
catamorphisms, 210
category theory, 174

`Char`, 24, 282
characters, *see* `Char`, 282
`chr`, 53, 282
`class`, 99, 154, 159, 166, 196, 200, 207
classes, 31, 99, 280
  constraints, 30, 99
  default definitions, 99
  derived instances, 100
  instances, 30, 99
  methods, 31
clearing the screen, 134
combinatory logic, 175
comments, 20
commutativity, 118
  for addition, 118, 246
  for multiplication, 118, 228
composition, *see* `.`
comprehensions
  list, 7, 47
  set, 47, 56
  string, 51
`concat`, 48, 206, 296
concatenation, *see* `concat`
conjunction, *see* `&&`, `and`
cons, *see* `:`
`const`, 43, 287
continuation-passing style, 252
control characters, 24, 134
`Control.Applicative`, 179, 290
`Control.Monad`, 172, 183, 291
crush operators, 210
`curry`, 284

`data`, 93
`Data.Char`, 132, 140, 179, 282
`Data.Foldable`, 200, 205, 225, 294
`Data.List`, 35, 87, 140
`Data.Monoid`, 197, 292
defunctionalisation, 254
`deriving`, 100
`digitToInt`, 132, 282
disjunction, *see* `||`, `or`
distributivity, 156, 228, 243
  contravariant, 235
`div`, 18, 35, 84, 281
division, *see* `div`, `/`
`do`, 126, 166, 181
domain-specific languages, 7, 74
`Double`, 25, 283
Dr Seuss, 178
`drop`, 16, 64, 68, 286
`dropWhile`, 76, 286

effects, 7, 124, 162
`elem`, 202, 294
`empty`, 181, 291
`Eq`, 31, 49, 280
equality, *see* `==`
equational reasoning, 8, 228, 257
`error`, 90, 190, 288
Euclid's algorithm, 71
evaluation, 15, 22, 113, 212
  call-by-name, 215, 216
  call-by-value, 214, 216
  innermost, 214
  lazy, 7, 26, 40, 49, 51, 148, 150, 219
  outermost, 214
  top-level, 223
`even`, 38, 65, 283
examples
  abstract machine, 106, 261
  base conversion, 83
  binary string transmitter, 82
  Caesar cipher, 52
  calculator, 191
  compiler, 241, 249
  countdown problem, 111
  expression parser, 187
  fast reverse, 238
  Fibonacci sequence, 64, 226
  game of life, 133
  game of nim, 129
  hangman, 128
  insertion sort, 63
  Newton's method, 227
  prime numbers, 49, 222
  quicksort, 10, 65
  sieve of Eratosthenes, 222
  tautology checker, 101
  tic-tac-toe, 139
  tree relabelling, 171
  virtual machine, 256
  voting algorithms, 86
exceptions, 160, 261
exponentiation, *see* `^`
expressions, 3
  arithmetic, 106, 111, 187, 241
  conditional, *see* `if`
  impure, 125
  lambda, *see* `\`
  logical, 101
  pure, 125
  reducible, 213

`False`, 22, 24
`filter`, 75, 211, 285
`filterM`, 173
`flip`, 287
`Float`, 24, 54, 283
`fmap`, 154, 288
`fold`, 200, 294
`Foldable`, 200, 294
foldables, 7, 200
  `Maybe`, 210
  `Tree`, 202, 210

list, 201, 295
`foldl`, 68, 80, 200, 225, 294
`foldl'`, 225
`foldl1`, 203, 294
`foldMap`, 200, 294
`foldr`, 68, 77, 89, 200, 294
`foldr1`, 143, 203, 294
FP, 8
`Fractional`, 35, 281
`fromIntegral`, 54
`fst`, 41, 284
functions, 3, 16, 27, 287
combinator, 254
combinatorial, 114
composite, *see* `.`
constant, *see* `const`
constructor, 94
continuation, 252
curried, 28, 43, 73, 215, 224
generic, 7, 156, 162, 172, 205
higher-order, 7, 74
identity, *see* `id`
nameless, *see* `\`
overloaded, 6, 30
polymorphic, 6, 29, 51, 74
recursive, 10, 59
strict, 215, 223
total, 27
`Functor`, 154, 288
functors, 7, 153, 288
`IO`, 155, 289
`Maybe`, 154, 289
`Parser`, 179
`Tree`, 155
function, 175
laws, 156, 236
list, 154, 236, 289
state, 169

game trees, 145
generators, *see* `<-`
`getAll`, 293
`getAny`, 293
`getChar`, 125, 287
`getLine`, 127, 288
`getProduct`, 199, 293
`getSum`, 198, 292
GHC, 14, 116, 150
GHCi, 14, 17
commands, 18
grammars, 187
ambiguous, 189
greatest common divisor, 71
guards, 7, 39, 48, 70

`head`, 15, 42, 285
`hSetBuffering`, 150
`hSetEcho`, 128

`id`, 82, 287
identifiers, 18, 185
identities, 118
for addition, 10, 232
for alternatives, 182
for append, 236, 238
for applicatives, 162
for composition, 82
for division, 118
for functors, 156
for monads, 174
for monoids, 196
for multiplication, 60, 67, 118
`if`, 23, 39
`import`, 52
indentation, 6, *see* layout rule
induction, 8, 231
hypothesis, 232
on expressions, 243, 250
on lists, 235
on numbers, 232, 234
on trees, 240
inequality, *see* `/=`
infinity, 216, 231
infix notation, 18
`init`, 70, 286
input/output, *see* `IO`, 287
`instance`, 99
`Int`, 24, 283
`Integer`, 24, 227, 283
`Integral`, 35, 69, 281
`intToDigit`, 283
`IO`, 12, 124, 287
`isAlpha`, 183, 282
`isAlphaNum`, 183, 282
`isDigit`, 144, 183, 282
`isLower`, 183, 282
`isSpace`, 185, 282
`isUpper`, 183, 282
ISWIM, 8
`iterate`, 83, 286

`join`, 173

keywords, 6, 19

lambda calculus, 8, 45
layout rule, 19, 126, 166
tabs, 20
`length`, 16, 48, 61, 78, 80, 202, 294
`let`, 170
lexicographic ordering, 32, 87
Lisp, 8
lists, 6, 9, 15, 25, 47, 285
elements, 25, 41
empty, *see* `[]`
indexing, *see* `!!`
infinite, 26, 51, 83, 219
length, 25

singleton, 25, 113, 160, 178
sorted, 10, 50, 62, 65, 72, 87
logical
operators, *see* `&&`, `||`, `not`
propositions, 101
tautologies, 101
values, *see* `Bool`
lookup tables, 49, 93
Luhn algorithm, 46, 91

`many`, 184, 291
`map`, 74, 153, 207, 287
`mapM`, 172, 209, 297
`mappend`, 196, 292
`max`, 31, 280
`maximum`, 202, 294
`Maybe`, 95, 284
`mconcat`, 196, 292
`mempty`, 196, 292
`min`, 31, 280
minimax algorithm, 148
`minimum`, 202, 294
Miranda, 8
ML, 8
`mod`, 35, 48, 53, 84, 222, 281
`Monad`, 12, 166, 290
`MonadPlus`, 183, 291
monadplus
`Maybe`, 292
list, 292
monads, 7, 12, 164, 290
`IO`, 168, 290
`Maybe`, 167, 290
`Parser`, 181
function, 176
laws, 174
list, 167, 290
state, 168, 170
substitution, 176
`Monoid`, 196, 292
monoids, 196, 292
`Maybe`, 197, 292
addition, 198, 292
conjunction, 199, 293
disjunction, 199, 293
function, 210
laws, 196
list, 197, 292
multiplication, 198, 199, 293
pair, 210
`mplus`, 183, 291
multiplication, *see* `*`, `product`
`mzero`, 183, 291

naming requirements, 18, 92, 93
`negate`, 34, 280
negation, *see* `not`, `negate`
`newtype`, 95, 169, 179
non-determinism, 161
`not`, 40, 281
`null`, 70, 202, 294
`Num`, 10, 30, 34, 68, 280
numbers, 283
binary, 82
decimal, 82
floating-point, *see* `Float`, `Double`
integer, *see* `Int`, `Integer`
natural, 95, 96, 185, 187, 231
perfect, 57
prime, 49, 222
random, 151
Unicode, 53, 85, 282

`odd`, 65, 81, 284
operators, 44, 76, 79, 112
`or`, 77, 80, 205, 296
`Ord`, 11, 31, 50, 62, 280
`ord`, 53, 282
`otherwise`, 39, 281
overloading, 30

parse trees, 187
parsers, 177
pattern matching, 7, 40, 67, 230
patterns, 40
list, 41, 61
non-overlapping, 230
tuple, 41
wildcard, *see* `_`
polymorphism, 29
powerset, 173
predecessor, 59, 65
predicates, 75, 183
`print`, 116, 288
priorities, 15, 17, 187
`Product`, 199, 293
`product`, 16, 61, 66, 77, 80, 202, 294
programming
batch, 123
functional, 4
imperative, 5, 213
interactive, 123
modular, 8, 221
`pure`, 159, 289
`putChar`, 125, 288
`putStr`, 127, 288
`putStrLn`, 127, 288

`Read`, 33, 280
`read`, 33, 280
`readLn`, 288
`recip`, 35, 38, 281
reciprocation, 44, *see* `recip`
recursion, 7, 59
multiple, 64
mutual, 65, 107, 185
redex, 213

reduction, 213
  under lambdas, 216
remainder, *see* `mod`
`repeat`, 84, 286
`replicate`, 234, 286
reserved words, 19
`return`, 125, 166, 290
`reverse`, 16, 61, 78, 80, 235, 238, 287
rounding errors, 25

scalar product, 58
scripts, 17
sections, 44, 77
`seq`, 175
`sequence`, 209, 297
`sequenceA`, 162, 209, 297
sharing, 218
`Show`, 32, 112, 280
`show`, 32, 280
side effects, 7, 12, 124
sign, *see* `signum`
`signum`, 34, 39, 280
`snd`, 41, 284
`some`, 184, 291
sorting
  insertion sort, 63
  merge sort, 72
  quicksort, 10, 65
spacing, 186
`splitAt`, 38, 144, 286
stacks, 241, 250
  control, 107
  underflow, 244
standard prelude, 15, 280
state of the world, 124
state transformer, 168, 178
`String`, 24, 283
strings, *see* `String`, 283
subtraction, *see* `-`
successor, 44, 96, 216
`Sum`, 198, 292
`sum`, 9, 16, 77, 79, 80, 202, 294
`System.IO`, 128, 140
`System.IO.Unsafe`, 137

tabs, 20
`tail`, 15, 42, 285
`take`, 16, 221, 285
`takeWhile`, 76, 285
termination, 4, 8, 216, 219
tokens, 186
`toList`, 203, 294
`toLower`, 283
`toUpper`, 283
`transpose`, 141
`Traversable`, 207, 297
traversables, 7, 206, 297
  `Maybe`, 210, 297
  `Tree`, 208, 210
  list, 208, 297
`traverse`, 207, 297
`True`, 22, 24
truth tables, 102
tuples, 26, 284
  arity, 26, 41
  components, 26, 41
  empty, *see* `()`
  pairs, 26, 47, 50, 284
  triples, 26, 284
type
  constructors, 93
  declarations, 92, 93, 95
  errors, 23
  inference, 6, 10, 22
  parameters, 92, 95
  safety, 23, 96
  variables, 29
`type`, 92
types, 6, 10, 22
  applicative, *see* `Applicative`
  basic, 23
  container, 155
  equality, *see* `Eq`
  foldable, *see* `Foldable`
  fractional, *see* `Fractional`
  function, 27
  functorial, *see* `Functor`
  integral, *see* `Integral`
  list, 25, 97, 285
  maybe, *see* `Maybe`
  monadic, *see* `Monad`
  monoid, *see* `Monoid`
  numeric, *see* `Num`
  ordered, *see* `Ord`
  overloaded, 30
  parameterised, 92, 95
  polymorphic, 29
  readable, *see* `Read`
  recursive, 96
  showable, *see* `Show`
  traversable, *see* `Traversable`
  tree, 97, 147, 227, 240
  tuple, 26

`uncurry`, 284
`unfold`, 90
Unicode, 24
`unlines`, 142
`unsafePerformIO`, 137

`where`, 19

`zip`, 50, 63, 286
`ZipList`, 175
zipper, 108
`zipWith`, 143