

---

# Index

- ⊙ asymptotic notation, 157
- ⌊−⌋ floor function, 46, 52, 62
- ⊥ undefined value, 24, 28, 30, 104, 177, 215, 217, 226, 233
- π, 50, 215
- ⊑ approximation ordering, 217
- (%) integral ratio, 50
- "" empty list of characters, 18
- " double quotes, 4
- ' single quote, 3
- (!!) list-indexing operation, 9
- (!) array-indexing operation, 261
- (\$!) strict application operator, 153, 252
- (\$) application operator, 153
- (&&) boolean conjunction, 10
- () null tuple, 30, 33, 240
- (\*\*) exponentiation, 59
- (,) pair constructor, 74, 146
- (−) subtraction operator, 27
- (−>) function type, 1
- (.) function composition, 3, 15, 31
- (//) array update operation, 262
- (/=) inequality test, 9
- (:) cons, list constructor, 64
- (<=) comparison test, 9, 76
- (<=<) right-to-left Kleisli composition, 247
- (==) equality test, 9
- (>=>) left-to-right Kleisli composition, 247
- (>>) sequence operation, 240
- (>>=) monadic bind, 241, 279
- (~) exponentiation, 59
- (^^) exponentiation, 59
- (| |) boolean disjunction, 32
- .hs Haskell script, 35, 227
- .lhs literate Haskell script, 8, 35, 227
- :load instruction, 13
- :set instruction, 13, 148
- :type instruction, 22
- @ as patterns, *see* patterns
- [] empty list, 17, 63, 64, 104, 121, 151
- \ escape character, 18
- \n newline character, 3, 18
- \t tab character, 18
- ` back-quote, 9, 25
- (++) list concatenation, 10, 15, 69, 113
- n+k patterns, 111
- abs, 50, 54
- absolute value, *see* abs
- abstract data types, 194, 239, 259
- abstract syntax trees, 201
- accumArray, 260
- accumulating functions, 260
- accumulating parameters, 159, 171, 288, 289, 323
- actions, 239
- Agda, 48
- Algorithm Design, 145, 154
- all, 94
- alphabetical order, 5, 19
- anagrams, 16
- and, 74
- anti-symmetric relation, 217
- any, 102
- approx, 218
- Array, 259
- array, 259
- arrays
  - immutable, 259
  - mutable, 254
- associative operations, 15, 26, 70, 112, 113, 119, 124, 129, 184, 231, 246, 247, 281, 326
- assocs, 262
- asymptotic complexity, 155
- Augustsson, L., 47
- auxiliary results, 114, 117, 140
- base cases, *see* induction
- Bentley, J., 21, 127, 144
- Bifunctor, 82
- bifunctors, 82, 87, 302, 326
- binary operators, 25
- binary search, 54
- binary trees, 165, 249

- binding power, 2, 14, 25
- Bird, R., 87, 180
- blank characters, 3
- BNF (Backus-Naur form), 286, 305
- Bool, 10, 30
- boolean
  - conjunction, *see* (&&)
  - disjunction, *see* (||)
- bottom, *see*  $\perp$  undefined value
- braces, 15, 36, 242
- brackets, 15
- breadth-first search, 257
- break, 102
- C, 239
- C#, x
- case expression, *see* expressions
- case analysis, 10, 53
- Category Theory, 71, 87
- chain completeness, 116, 212, 220
- chain of approximations, 215, 218, 225
- Char, 3, 30, 90, 254
- Chitil, O., 209
- comment convention, 8
- common subexpression elimination, 147
- commonWords, 3, 4, 34, 75
- comparison operations, 32
- compiled functions, 154
- compilers, 36, 154
- complete partial orderings, 218
- Complex, 49
- concat, 6, 67, 70, 157
- concatMap, 307
- concrete data types, 194
- conditional expressions, 189
- const, 86
- context, 11
- Control.Monad, 247, 264
- Control.Monad.ST, 251
- Control.Monad.State.Lazy, 251
- Control.Monad.State.Strict, 251
- conversion functions, 53
- coprime numbers, 66
- Coq, 337
- cosine function, ix
- cp cartesian product, 92, 93, 97, 131, 155, 244, 320
- cross, 81, 301
- curry, 86, 135
- Curry, H.B., 87
- cycle, 232
- cyclic lists, 212
- data constructors, 25, 56
- data type declarations, 30, 56, 189, 194, 202, 229
- Data.Array, 259
- Data.Char, 13, 42, 227, 283
- Data.Complex, 49
- Data.List, 75, 106, 125, 154, 311
- Data.Maybe, 319
- Data.STRef, 251
- de Moor, O., 87
- deep embeddings, 194
- default definitions, 31
- dependently-typed languages, 48, 90
- deriving clauses, 39, 57
- directed graphs, 260
- distributive operations, 130
- div, 9, 24, 59
- divide and conquer algorithm, 46, 76
- divMod, 51
- do-notation, 34, 36, 239, 242, 245
- done, 240
- Double, 49
- drop, 79
- dropWhile, 106
- e, 141
- echoing, 241
- efficiency, ix, 145
- Either, 82, 132
- either, 132
- else, 12
- embedded domain-specific language, 209
- empty list, *see* []
- Enum, 65, 90, 211
- enumerations, 65, 210, 217
- enumFrom, 211
- Eq, 31, 56
- equality operations, 31
- equational reasoning, ix, 1, 73, 81, 89, 96, 99, 110, 135, 298
- error, 36, 39, 179
- error messages, 23, 24, 39, 43, 103, 151, 192, 250, 279
- evaluation
  - eager, 28, 154, 157
  - innermost, 28
  - lazy, 28, 75, 80, 89, 145, 154, 175, 198, 243
  - outermost, 28
  - to normal form, 22, 27, 146, 156
- exception handling, 239
- exp, 40, 110, 141
- explicit layout, 242
- exponentiation, 40, 59, 110
- export declarations, *see* modules
- expressions
  - case, 127, 245
  - conditional, 12, 23, 181
  - lambda, 26, 148, 242
  - let, 24, 146, 147, 248
  - well-formed, 22
- factorial function, 28, 220
- factoring parsers, 282
- failure, 39
- Feijen, W., xi
- Fibonacci function, 164, 232, 251, 266
- FilePath, 34

- filter, 38, 67, 70, 72, 97, 98, 119, 134, 299
- flat ordering, 217
- flip, 59, 123, 148
- Float, 2, 30, 49
- Floating, 52
- floating-point literal, 51
- floating-point numbers, 112, 293
- floor, 46, 52, 60
- fmap, 71
- foldl, 122, 150, 152, 260
- foldl', 150, 152
- foldl1, 231
- foldr, 117, 152, 164
- foldr1, 121, 231
- forall, 253
- fork, 81, 134, 301
- Fractional, 50
- fromInteger, 50, 67, 151
- fromIntegral, 51, 67, 151
- fromJust, 319
- fst, 28, 51
- function, 1
  - application, 2, 4, 14
  - arguments, 1, 3
  - composition, *see* (.)
  - computable, 219
  - continuous, 219
  - conversion, 50
  - higher-order, 110
  - identity, *see* id
  - monotonic, 219
  - non-strict, 29, 58
  - overloaded, 31
  - partial, 9
  - polymorphic, 31, 72
  - primitive, 150
  - recursive, 17, 219
  - results, 1, 3
  - strict, 29, 59, 72, 120, 137, 150, 154
  - type, *see* ->
  - values, 145
- Functor, 71, 264
- functors, 87, 264
- getChar, 240
- getLine, 241
- GHC, 35, 36, 154, 169, 275
- GHCi, 12, 22, 36, 154
- Gibbons, J., 209, 275
- global definition, *see* top-level definition
- Goerzen, J., 21
- Gofer, x, 275
- golden ratio, 164
- Graham, R., 62
- grammars, 286
- greedy algorithms, 192
- guard, 306
- guarded equations, 9, 12, 332
- guards, 10
- Hamming, W.R., 232
- Hangman, 266
- Hardy, G.H., 78
- Harper, B., 47
- hash tables, 256
- Haskell, x, 1
  - 1998 online report, 21
  - 2010 online report, 21, 111
  - commands, 1, 33, 34, 239
  - layout, 36
  - libraries, 7, 181
  - numbers, 2, 112
  - Platform, 12, 154
  - reserved words, 12
  - standard prelude, *see* standard prelude
  - syntax, 22
  - values, 22
  - well-formed literals, 122
- head, 38, 68, 72
- head normal form, 146, 157, 172
- helper functions, 26
- Hinze, R., 275
- History of Haskell*, 20
- homomorphisms, *see* laws
- Hughes, J., 209
- Hutton, G., 21, 297
- id, 18, 70, 71, 94, 96, 253
- idempotence, 205, *see* laws
- identities, *see* laws
- identity elements, 15, 96, 184, 247, 281
- identity function, *see* id
- if, 12
- imperative languages, x
- import declarations, *see* modules
- in-place algorithms, 170, 254
- indexitis, 95
- induction, 110, 111
  - base case, 77, 111
  - general, 179
  - inductive case, 77, 111
  - over lists, 113
  - over numbers, 110
  - pre-packaged, 120
- inductive cases, *see* induction
- infinite loops, 24, 28, 30, 77, 124, 211
- infix data constructors, 194
- infix operations, 9
- infixr fixity declaration, 153
- inits, 125
- inlining, 54
- insertion sort, *see* sorting
- instance declarations, 32
- Int, 2, 30, 49, 155, 254
- Integer, 2, 49, 155
- integer literals, 50
- Integral, 51, 155
- interact, 227
- interaction, 221

- interactive programs, 227
- interpreters, 36
- involutions, *see* laws
- IO, 33, 239
- isAlpha, 42
- isSpace, 283
- it, 229
- iterate, 64, 213, 301
- Ix, 254, 255
- Jeuring, J., 209
- join, 264
- Jones, M., 144, 275
- Just, 244
- Knuth, D.E., 21, 62, 167
- KRC, x
- Lapalme, G., 180
- last, 68
- Launchbury, J., 275
- laws
  - arithmetic, 17
  - bifunctor, 302
  - commutative, 186
  - distributive, 186
  - equational, ix
  - functor, 71, 81, 303, 330
  - fusion, 120, 131, 176
  - homomorphisms, 185, 186
  - idempotence, 204
  - identities, ix, 110
  - involutions, 96, 113
  - leapfrog, 247, 265
  - left-distributive, 293
  - left-zero, 293
  - monad laws, 246, 279
  - naturality, 72, 87, 97
  - point-free, 110
  - right-zero, 293
  - trigonometric, ix
  - tupling, 152, 165
- layout description language, 182
- lazy evaluation, *see* evaluation
- least fixed points, 220
- least upper bounds, 218
- left-recursion problem, 287
- Leibniz, G., 243
- length, 69, 79, 160
- let, *see* expressions
- lexicographic order, 76, 189, 217
- liftM, 264
- lines, 188, 200
- Linux, 12
- list
  - adjacency, 261
  - comprehensions, 66, 92, 156, 244, 245, 248
  - concatenation, *see* (++)
  - cyclic, 210
  - doubly-linked, 228
  - finite, 64
  - identity function, 118
  - indexing, *see* (!!)
  - infinite, 53, 64, 75, 108, 210
  - notation, 3
  - partial, 64, 115, 211
- listArray, 260
- lists
  - adjacency, 261
- literate programming, 8
- local definitions, 11, 149, 256
- log, 141
- logarithmic factors, 159
- logarithmic time, 56
- logBase, 2, 141
- lookup, 244, 319
- loop invariants, 255, 263
- lower bounds, 157
- lowercase, 5
- Loyd, S., 267
- Mac, 12
- main, 34, 227
- map, 5, 23, 31, 38, 67, 70, 80, 91, 119, 299
- mapM, 265
- mapM\_, 265
- Marlow, S., 20
- Maslanka, C., 48
- mathematical operators, 217
- matrices, 90, 94, 234
- matrix
  - addition, 105
  - multiplication, 105
  - transpose, 94, 106
- maximum, 122
- Maybe, 39, 244
- McIlory, D., 21
- mean, 151
- Meijer, E., 209, 297
- merge, 76, 211, 231
- mergesort, *see* sorting
- minimum, 104, 107, 122, 157, 211
- Miranda, x
- mkStdGen, 223
- ML, 29, 48
- mod, 9
- modules, 13, 25, 35, 309
  - export declarations, 35, 199
  - hierarchical names, 21
  - import declarations, 13, 35
- Monad, 243, 264
- monadic programming, 239
- MonadPlus, 292
- monads, 243
  - commutative, 264
- monoids, 247
- mplus, 292
- mutable structures, 248
- mzero, 292

- Nat, 56, 110, 132
- natural transformations, 87
- negate, 50
- newline character, *see* \n
- newSTRef, 251
- Newton's method, 60
- newtype declarations, 278
- non-decreasing order, 74
- none, 134
- normal form, 146
- not, 30
- notElem, 98
- Nothing, 244
- nub, 106, 108
- null, 39, 68
- null tuple, *see* ()
- Num, 23, 31, 49, 56
- Number Theory, 219
- numbers
  - complex, *see* Complex
  - floating point, *see* Float, Double
  - floating-point, 60
  - integer, *see* Int, Integer
  - limited precision integers, *see* Int
  - natural, 56, 110
  - unlimited precision integers, *see* Integer
- O'Neill, M., 237
- O'Sullivan, B., 21
- offside rule, 36, 242
- one, 134
- Oppen, D., 209
- or, 102
- Ord, 32
- order of association, 2–4, 17, 25, 62, 196
- Orwell, x
- otherwise, 11
- pairs, 2, 74, 76, 82, 177
- palindromes, 41
- paper-rock-scissors, 221
- paragraphs, 191
- parentheses, 15
- parsers, 276
- parsing, 239
- partial application, 87
- partial numbers, 58
- partition, 311
- Patashnik, O., 62
- Paterson, R., 337
- patterns
  - n+k, 111
  - as patterns, 77, 83
  - disjoint, 68
  - don't care, 67, 73, 103, 268
  - exhaustive, 68
  - irrefutable, 231
  - matching, 57, 67, 68, 74, 115, 194, 332
  - wildcard, 67, 268
- perfect numbers, 65
- Perlis, A., 145
- persistent data structures, 254
- Peyton Jones, S., 21, 209
- Pierce, B., 87
- plumbing combinators, 86
- point-free calculations, 86, 330
- point-free reasoning, 298
- pointers, 146
- postconditions, 255, 263
- precedence, 14, 25, 37
- preconditions, 255, 263
- prefix names, 25
- prefix operators, 50
- Prelude, 25
- primes, 148, 213, 219, 220, 233
- printing values, 33
- profiling tools, 154
- program variables, 251
- programs, 7
- prompt symbol, 13
- prompts, 229, 240
- proof format, xi, 112
- properFraction, 56
- putChar, 240
- putStrLn, 16, 33, 239, 240
- Pythagorean triads, 66
- Python, x, 239, 252
- qualified names, 306
- quicksort, *see* sorting
- Rabbi, F., 180
- Ramanujan, S., 78
- random numbers, 223, 250
- randomR, 223
- rank 2 polymorphic types, 253
- Rational, 49, 50
- Read, 41, 122, 277
- read, 52, 122
- readFile, 34
- reading files, 34
- ReadS, 277
- reads, 277
- readSTRef, 251
- Real, 50
- recursive definitions, 29, 77, 219
- reduction, *see* evaluation
- reduction steps, 155, 156
- reference variables, 251
- reflexive relation, 217
- repeat, 108, 212
- return, 241
- reverse, 42, 72, 113, 117, 123, 159
- running sums, 125
- runST, 252
- SASL, x
- scanl, 125, 127
- scanr, 130

- scientific notation, 60, 112
- scope, 11
- scripts, 7, 25, 148
- sections, 26, 52, 53
- segments, 127, 316
- `select`, 175
- selection sort, *see* sorting
- semicolon, 36
- separator characters, 200
- `seq`, 150, 153, 226
- `sequence_`, 264
- set theory, 211
- shallow embeddings, 187, 192
- shared values, 146
- `Show`, 32, 40, 56, 229, 276, 291
- `Shows`, 289, 311
- `showsPrec`, 290, 291, 308
- side-effects, 243
- sieve of Sundaram, 233
- signum, 50
- Sijtsma, B., 237
- `sin`, 2, 14
- sine function, ix, xi, 2, 14
- size measures, 156, 202
- `snd`, 51
- `sort`, 14, 75, 154
- sorting, 5, 6, 94, 167
  - insertion sort, 172
  - mergesort, 76, 168
  - numbers, 262
  - quicksort, 169, 254, 263
  - selection sort, 172
- space character, 3
- space efficiency, 29, 84, 147, 149, 154, 171
- space leaks, 147, 151, 170, 171
- `span`, 75, 102
- Spivey, M., 337
- `splitAt`, 80, 168
- `sqrt`, 60
- stand-alone programs, 34, 227
- standard prelude, 13, 30, 39, 42, 51, 56, 73, 80, 86, 87, 127, 150, 161, 168, 213, 232
- `STArray`, 254
- state monad, 247
- state threads, 251
- state-thread monad, 251
- Stewart, D., 21
- stream-based interaction, 226
- `STRef`, 251
- strictness flags, 58, 194
- `String`, 6
- strings, 4
- subclasses, 50
- `subseqs`, 133, 146, 158
- subsequences, 146
- `subtract`, 53, 59
- Sudoku, 89, 258, 321
- `sum`, 150
- Sundaram, S.P., 233
- superclasses, 32
- Swierstra, D., 209
- syntactic categories, 286
- `System.Random`, 223
- `System.IO`, 242
- `tail`, 38, 68
- `tails`, 128
- `take`, 6, 30, 79, 218
- `takeWhile`, 52, 106
- taxicab numbers, 79, 88
- terminator characters, 200
- texts, 3, 5, 181
- `then`, 12
- time efficiency, 92, 123, 126, 147, 154, 156, 182
- `toInteger`, 51, 151
- `toLower`, 5, 13, 38
- top-level definitions, 147, 256
- `toRational`, 50
- toruses, 234
- `toUpper`, 44, 227
- transitive relation, 217
- `transpose`, 106
- trees, 71, 161, 165, 190, 194
- trigonometry, ix, 2
- tupling, 164, 170
- tupling law of `foldr`, 165
- tying the recursive knot, 214, 238
- type
  - annotations, 41
  - classes, 25, 30, 31, 155
  - declarations, 7, 20
  - inference, 20, 22
  - signatures, 6, 20, 30, 155
  - synonyms, 5, 90, 183, 278
  - variables, 7, 31
  - well-formed, 22
- types
  - compound, 30
  - isomorphic, 278
  - polymorphic, 31, 72
  - primitive, 30
- UHC (Utrecht Haskell Compiler), 47
- `uncurry`, 86, 135
- `undefined`, 24, 64
- underscore convention, 264
- `unlines`, 227
- `unsafePerformIO`, 242
- `until`, 52, 65
- `unwords`, 44
- `unzip`, 81
- upper bounds, 218
- visible characters, 3
- Visual Basic, x
- Wadler, P., 87, 209
- `where` clauses, 11, 24, 36, 147, 242
- white space, 283

- wholemeal programming, 95
- Wilde, O., 145
- Windows, 12
- WinGHCi, 12
- words, 4, 5, 38, 107, 191
- World, 239
- writeFile, 34
- writeSTRef, 251
- writing files, 34
- zip, 73
- zipWith, 73, 105
- zipWith3, 230