class cast exception, 143
clone, of object, 211
code reuse, 235
CoinPuzzle example, 231
collection classes, 511
collections, 511–512
comment, 33–37
comparable, 253
comment, 33–37
compiler, 2
comparability, 254
comparator, 140–142
compare, 253
comparing values, 197
compile-time, 69
CompInsSort example, 142
complexity, 1, 81
of method, 235
space, 82, 92
time, 82, 92
central, 504
connected components, 404
concrete class, 505
concrete class, 505
connected component, 404
ConstantGenerator example, 153
cost, 15
constructor, 10
create, 59, 373
copy, 59, 373
container types, 72
craftsmen, 81
cycle, 404
DAG, 404
Dahl, Roald, 315
data
class, 507
package, 507
encoding, 241, 303
unicode, 492
energy, 195, 201, 258
equality, 162
equal, 162
equals, default, 13
equivalent, 161
equality, 162
design
deterministic, 134
disjoint, 277
dictionary, 369
Dijkstra example, 432
disjoint, 277
documentation, 513, 516
Dodge, Charles Lutwidge, 219
doubly linked list, 201
Doyle, Sir, 441
Drake, Milton, 489
dummy nodes, 215
dynamic programming, 99
element
cost, 15
current, 15
equality, 162
class, 507
package, 507
encoding, 241, 303
unicode, 492
energy, 195, 201, 258
equality, 162
equality, 162
equality, 162
equality, 162
equality, 162
equality, 162
equality, 162
equality, 162
example
AbstractGenerator, 152
AbstractCard, 156
Analysis, 35
atinlay, 15
BankAccount, 11
BubbleSort, 119
Calc, 282
Card, 155
Circuit, 334
CoinPuzzle, 231
CompInsSort, 142
ConstantGenerator, 153
Dijkstra, 432
Example, 2
Fibo, 106
Floyd, 428
FullPostage, 99
Generator, 152
HelloWorld, 162, 164
HexBoard, 313
HexMove, 313
HTML, 251
Huffman, 303, 304, 317
Huffman2, 317
Index, 396
InfiniteQuestions, 288
InsertionSort, 125
LinkedList, 216
LongWords, 70
LongWords2, 71
LongWords3, 78
LSystem, 56
MCST, 430
MergeSort, 127
nim, xii
ParkingLot, 183
ParkingLot2, 271
Pedigree, 280
PFGenerator, 168
PhoneBook, 138, 143
PinochleCard, 159
Player, 314
PokerCard, 158
PrimeGenerator, 154
QuickSort, 131, 222
RadixSort, 135
Ratio, 8, 254
RBSymTab, 362
Reachability, 422
Reader, 248
Rect, 20
Recursion, 94, 102
RecursiveIterators, 298
RecursivePostage, 98
SelectionSort, 122
Sort, 265
sqr, 34
StreamReader, 43
StringVector, 71
SymbolTable, 249
SymMap, 369
SymTab, 346
Token, 248
TopoSort, 426
Unique, 182
UniqueFilter, 170
Warshall, 427
WordFreq, 48
WordList, 19

Example example, 2
example, icon for, xii
extensible, 70
extension of class, 408

Fibo example, 106
Fibonacci numbers, 106
fields, 8
FIFO, see also queue
filter, 170
finger method, 192
Floyd example, 428
foreach, 500
forest, 277
format, 496
Fox, Jason, 277
free list, 183, 186, 410
friction, 110
definition, 108, 110
in heaps, 320
Frost, Robert, 507
FullPostage example, 99
function, see also method
garbage collection, 199
garbage collector, 189, 494
gate delay, 334
Geisel, Theodor Seuss, 5, 369
generator, 152, 155
as iterator, 167, 170
Generator example, 152
generic, 38
generic class, 69, 72
Gillespie, Haven, 179
grammars, 57
graph, 403, 434
acyclic, 404
adjacency list, 416
arc, see also graph, edge
complete, 404
dense, 416
directed, 403
directed acyclic, 404
Index

is a graph, 403
iterator, 209–211
remove, 179
singly linked, 188–201
tail, 179
load factor, 383
logic gates, 333
LongWords example, 70
LongWords2 example, 71
LongWords3 example, 78
LSystem example, 56
map, 369–398
ordered, 392–398
Mars, life on, 191
matrix, 60–63
adjacency, 410–416
symmetric, 66, 410
triangular, 66
maze
finish cell, 242
start cell, 242
mazes
solving, 242–244
McCloskey, Robert, 119
MCST example, 430
mechanics of data structures, 6
median, 259
memory leak, 194
mention, 403
merge, 127
mergesort, 127–131
MergeSort example, 127
message, 26, see also method
method, 5, 490
calling, 5
calls, 8
destructive, 213
helper, 318
nondestructive, 194
postcondition, xii
precondition, xii
this, 17
methods, static, 322
minimum spanning tree, 429
Modula-2, 33
mutator, 13

nanosecond, 93
Napster, 15

new operator, 12
nim example, xii
node
degree, 279
depth, 279
interior, 279
level, 279
parent, in tree, 277
sibling, 279
of tree, 277
nondeterministic, 134
null, 189

Oakland, Ben, 489
object, 5, 8, 489
general, 345
model of, 7, 8
object orientation, 5
in languages, 5
terminology of, 8–11
object-oriented programming, 5
objects, comparable, 253, 258
optimization, 91
ordered
list, 257
map, 392–398
vector, 259
ordered structure, 144
ordered structures, 253, 272
overriding methods, 357
overriding operators, 253

package, 11, 489
default, 509
encapsulation, 507
user, 509
ParkingLot example, 183
ParkingLot2 example, 271
partially ordered, 140
Pascal, 33
path, 279
end points, 404
simple, 404
Pedigree example, 280
performance
exponential, 85
linear, 83
polynomial, 83
superlinear, 83
PFGenerator example, 168
philosophy, 13
PhoneBook example, 138 143
Pig Latin, 14
PinochleCard example, 159
pivot, 131 133
Player example, 314
pointer, see also reference
PokerCard example, 158
postcondition, xii 34
postfix, 290
precedence, operator, 282
precondition, xii 34
PrimeGenerator example, 154
principle, xii 3
  assertions, 35 515
  code reuse, 17
  code review, 269
  consistent interfaces, 51
  consistent state, 191
  drawing pictures, 189
  equal hash codes, 386
  fighting imperfection, 509
  hiding opposing references, 286
  icon for, xii
  interface design, 25
  live enumerations, 162
  most general method, 300
  natural interface, 319
  overriding signatures, 256
  principled programmer, 3
  progress in recursion, 96
  protected data, 25
  providing hash function, 385
  public protection, 508
  questioning asymmetry, 201
  read-only iterator values, 211
  reducing friction, 320
  static functions, 322
  symmetry, 194
  testing boundaries, 197
  understanding complexity, 235
principles, 1 3 515 516
principles, abstract, xii
priority queue, 315
problems, importance of, 2
procedure, 249 see also method
productions, 56
program counter, 222
program, state, 34
programming, craft of, xii
programs, writing large, 1
progress in recursion, 95
proof by induction, 94
protected, use of keyword, 507 509
protection of fields, 8 10
queue, 220 229 242
  dequeue, 229
  enqueue, 229
  head, 229
  priority, 315
  tail, 229
quicksort, 131 134
QuickSort example, 131 222
RadixSort example, 135
random access property, 45
Ratio example, 8 254
RBSymTab example, 362
Reachability example, 422
reachability, computing, 427
Reader example, 248
record, traditional, 8
Rect example, 20
recursion, 81 94 101
Recursion example, 94 102
recursive, 94
recursive data structure, 310
RecursiveIterators example, 298
RecursivePostage example, 98
reference, 19 188 189 493
  null, 189
rewriting, 96
Rhoda, 203
Rhonda, 205
rising edge, 337
rotations, 334
runtime, 44 69
scope, of identifier, 491
search
  binary, 260
  depth-first, 422
  linear, 268
selection sort, 122 123
SelectionSort example, 122
self-reference, 81 94 106
semantics, 69 369
  of data structure, 6
<table>
<thead>
<tr>
<th>Class/Concept</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueueVector</td>
<td>237</td>
</tr>
<tr>
<td>ReadStream</td>
<td>494</td>
</tr>
<tr>
<td>ReverseComparator</td>
<td>266</td>
</tr>
<tr>
<td>Set</td>
<td>23</td>
</tr>
<tr>
<td>SetVector</td>
<td>57</td>
</tr>
<tr>
<td>SinglyLinkedList</td>
<td>191</td>
</tr>
<tr>
<td>SinglyLinkedListIterator</td>
<td>210</td>
</tr>
<tr>
<td>SkewHeap</td>
<td>329</td>
</tr>
<tr>
<td>SplayTree</td>
<td>357</td>
</tr>
<tr>
<td>SplayTreeIterator</td>
<td>358</td>
</tr>
<tr>
<td>Stack</td>
<td>221</td>
</tr>
<tr>
<td>StackList</td>
<td>227</td>
</tr>
<tr>
<td>Structure</td>
<td>22</td>
</tr>
<tr>
<td>Table</td>
<td>393</td>
</tr>
<tr>
<td>ValueIterator</td>
<td>382</td>
</tr>
<tr>
<td>Vector</td>
<td>45 50 74 96 151 165</td>
</tr>
<tr>
<td>VectorIterator</td>
<td>165</td>
</tr>
<tr>
<td>VectorHeap</td>
<td>321</td>
</tr>
<tr>
<td>Vertex</td>
<td>407</td>
</tr>
<tr>
<td>structures</td>
<td></td>
</tr>
<tr>
<td>associative</td>
<td>345</td>
</tr>
<tr>
<td>subclass</td>
<td>409 504</td>
</tr>
<tr>
<td>subroutine, see also method</td>
<td></td>
</tr>
<tr>
<td>subtype</td>
<td>409 503 504</td>
</tr>
<tr>
<td>superclass</td>
<td>409</td>
</tr>
<tr>
<td>supertype</td>
<td>504</td>
</tr>
<tr>
<td>SymbolTable</td>
<td>346</td>
</tr>
<tr>
<td>SymbolTable example</td>
<td>249</td>
</tr>
<tr>
<td>SymMap example</td>
<td>369</td>
</tr>
<tr>
<td>symmetry</td>
<td>108</td>
</tr>
<tr>
<td>in interface</td>
<td>108</td>
</tr>
<tr>
<td>is predictability</td>
<td>108</td>
</tr>
<tr>
<td>SymTab example</td>
<td>346</td>
</tr>
<tr>
<td>syntax</td>
<td>69</td>
</tr>
<tr>
<td>Syracuse sequence</td>
<td>113</td>
</tr>
<tr>
<td>table</td>
<td>392 398</td>
</tr>
<tr>
<td>tail recursion</td>
<td>98</td>
</tr>
<tr>
<td>the entire argument array</td>
<td>70</td>
</tr>
<tr>
<td>The Phone Company</td>
<td>1</td>
</tr>
<tr>
<td>this</td>
<td>17</td>
</tr>
<tr>
<td>time stamp</td>
<td>424</td>
</tr>
<tr>
<td>time–space trade-off</td>
<td>193</td>
</tr>
<tr>
<td>token</td>
<td>247</td>
</tr>
<tr>
<td>Token example</td>
<td>248</td>
</tr>
<tr>
<td>topological sort</td>
<td>424</td>
</tr>
<tr>
<td>TopoSort example</td>
<td>426</td>
</tr>
<tr>
<td>totally ordered</td>
<td>140</td>
</tr>
<tr>
<td>tree</td>
<td>277 309</td>
</tr>
<tr>
<td>arity</td>
<td>279</td>
</tr>
<tr>
<td>AVL</td>
<td>311</td>
</tr>
<tr>
<td>binary</td>
<td>279</td>
</tr>
<tr>
<td>binary search</td>
<td>343 364</td>
</tr>
<tr>
<td>complete</td>
<td>278</td>
</tr>
<tr>
<td>degenerate</td>
<td>279</td>
</tr>
<tr>
<td>degree</td>
<td>279</td>
</tr>
<tr>
<td>expression</td>
<td>282</td>
</tr>
<tr>
<td>full</td>
<td>278 279</td>
</tr>
<tr>
<td>height balanced</td>
<td>311</td>
</tr>
<tr>
<td>is a graph</td>
<td>403</td>
</tr>
<tr>
<td>leaf</td>
<td>279</td>
</tr>
<tr>
<td>minimum spanning</td>
<td>429</td>
</tr>
<tr>
<td>oriented</td>
<td>279</td>
</tr>
<tr>
<td>pruning</td>
<td>313</td>
</tr>
<tr>
<td>red-black</td>
<td>361</td>
</tr>
<tr>
<td>root</td>
<td>277</td>
</tr>
<tr>
<td>rotation</td>
<td>354</td>
</tr>
<tr>
<td>splay</td>
<td>354 360</td>
</tr>
<tr>
<td>sub</td>
<td>277 279</td>
</tr>
<tr>
<td>traversing expression</td>
<td>282</td>
</tr>
<tr>
<td>trivial</td>
<td>277</td>
</tr>
<tr>
<td>type</td>
<td></td>
</tr>
<tr>
<td>conversion</td>
<td>492</td>
</tr>
<tr>
<td>primitive</td>
<td>491</td>
</tr>
<tr>
<td>reference</td>
<td>491</td>
</tr>
<tr>
<td>type bound</td>
<td>79</td>
</tr>
<tr>
<td>type parameter, actual</td>
<td>75</td>
</tr>
<tr>
<td>type parameter, formal</td>
<td>75</td>
</tr>
<tr>
<td>type parameters,</td>
<td>73</td>
</tr>
<tr>
<td>Unique example</td>
<td>182</td>
</tr>
<tr>
<td>UniqueFilter example</td>
<td>170</td>
</tr>
<tr>
<td>user</td>
<td>24</td>
</tr>
<tr>
<td>value</td>
<td></td>
</tr>
<tr>
<td>in association</td>
<td>369</td>
</tr>
<tr>
<td>value, comparison of</td>
<td>197</td>
</tr>
<tr>
<td>vector</td>
<td>43 54</td>
</tr>
<tr>
<td>capacity</td>
<td>46 50 55</td>
</tr>
<tr>
<td>cost of heapifying</td>
<td>339</td>
</tr>
<tr>
<td>end</td>
<td>27</td>
</tr>
<tr>
<td>extending</td>
<td>53</td>
</tr>
<tr>
<td>extent of</td>
<td>50</td>
</tr>
<tr>
<td>iterator</td>
<td>165 167</td>
</tr>
<tr>
<td>size</td>
<td>65</td>
</tr>
<tr>
<td>size of</td>
<td>50</td>
</tr>
<tr>
<td>tail</td>
<td>65</td>
</tr>
<tr>
<td>two-dimensional</td>
<td>60</td>
</tr>
</tbody>
</table>
vertex
  degree, 403
  destination, 403
  in-degree, 403
  out-degree, 403
  sink, 403
  source, 403

Warshall example, 427
Warshall's algorithm, 427
Web
  programming, xi
  resources available on, 2
White, Elwyn Brooks, 253
WordFreq example, 48
WordList example, 19 23 47