### Index

#### A

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A* algorithm</td>
<td>247, 262</td>
</tr>
<tr>
<td>AA-trees algorithms</td>
<td>262</td>
</tr>
<tr>
<td>Acceleration Tool Web Sites</td>
<td>257–258</td>
</tr>
<tr>
<td>Acyclic graphs</td>
<td>269</td>
</tr>
<tr>
<td>Algorithms</td>
<td>9, 22</td>
</tr>
<tr>
<td></td>
<td>black boxes, 23, 34</td>
</tr>
<tr>
<td></td>
<td>book, 2</td>
</tr>
<tr>
<td></td>
<td>graphs, 22</td>
</tr>
<tr>
<td></td>
<td>adjacency lists, 24</td>
</tr>
<tr>
<td></td>
<td>adjacency matrix, 27</td>
</tr>
<tr>
<td></td>
<td>friends attribute, 23</td>
</tr>
<tr>
<td></td>
<td>nutshell, 23</td>
</tr>
<tr>
<td></td>
<td>mathematical sense, 10</td>
</tr>
<tr>
<td></td>
<td>problem instances, 10</td>
</tr>
<tr>
<td></td>
<td>random-access machine, 10</td>
</tr>
<tr>
<td></td>
<td>trees, 22</td>
</tr>
<tr>
<td></td>
<td>binary, 30</td>
</tr>
<tr>
<td></td>
<td>multiway, 31</td>
</tr>
<tr>
<td></td>
<td>rooted, 30</td>
</tr>
<tr>
<td></td>
<td>turing machine, 9</td>
</tr>
<tr>
<td>Alpha-beta pruning</td>
<td>247</td>
</tr>
<tr>
<td>Annihilators</td>
<td>65</td>
</tr>
<tr>
<td>Approximation algorithm</td>
<td>245–246, 286</td>
</tr>
<tr>
<td>Arithmetic series</td>
<td>46</td>
</tr>
<tr>
<td>Assignment problem</td>
<td>220</td>
</tr>
<tr>
<td>Associativity</td>
<td>45</td>
</tr>
<tr>
<td>Asymptotic notation</td>
<td>10, 273</td>
</tr>
<tr>
<td></td>
<td>algorithms</td>
</tr>
<tr>
<td></td>
<td>constant factors, 19</td>
</tr>
<tr>
<td></td>
<td>mylist.sort(), 20</td>
</tr>
<tr>
<td></td>
<td>Python Call Graph, 20</td>
</tr>
<tr>
<td></td>
<td>running times, 21</td>
</tr>
<tr>
<td></td>
<td>statistical solutions, 21</td>
</tr>
<tr>
<td></td>
<td>timeit module, 20</td>
</tr>
<tr>
<td></td>
<td>trace module, 20, 22</td>
</tr>
<tr>
<td></td>
<td>average case, 18</td>
</tr>
<tr>
<td></td>
<td>best case, 18</td>
</tr>
<tr>
<td>black box</td>
<td>11</td>
</tr>
<tr>
<td>built-in functions</td>
<td>16</td>
</tr>
<tr>
<td>expressions</td>
<td>15</td>
</tr>
<tr>
<td>Greek letters</td>
<td>12</td>
</tr>
<tr>
<td>if statements</td>
<td>15</td>
</tr>
<tr>
<td>iteration counts</td>
<td>17</td>
</tr>
<tr>
<td>math refresher</td>
<td></td>
</tr>
<tr>
<td></td>
<td>factorial, 38</td>
</tr>
<tr>
<td></td>
<td>fractional powers, 38</td>
</tr>
<tr>
<td></td>
<td>logarithms, 38</td>
</tr>
<tr>
<td></td>
<td>polynomial, 38</td>
</tr>
<tr>
<td>Math Refresher</td>
<td>14</td>
</tr>
<tr>
<td>multiplying objects</td>
<td>17</td>
</tr>
<tr>
<td>Python lists</td>
<td>11</td>
</tr>
<tr>
<td>recurrences and</td>
<td>59</td>
</tr>
<tr>
<td>running times</td>
<td>15</td>
</tr>
<tr>
<td>sequential and nested cases</td>
<td>16</td>
</tr>
<tr>
<td>sort_w_check</td>
<td>18</td>
</tr>
<tr>
<td>sum function</td>
<td>16</td>
</tr>
<tr>
<td>worst case</td>
<td>18</td>
</tr>
</tbody>
</table>

#### B

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balanced tree structure</td>
<td>262</td>
</tr>
<tr>
<td>Balance factors</td>
<td>85, 86, 277</td>
</tr>
<tr>
<td>Baseball elimination problem</td>
<td>221</td>
</tr>
<tr>
<td>Bellman–Ford algorithm</td>
<td>262, 283</td>
</tr>
<tr>
<td>Binary counting</td>
<td>48</td>
</tr>
<tr>
<td>Binary encoding</td>
<td>275</td>
</tr>
<tr>
<td>Binary search trees</td>
<td>262</td>
</tr>
<tr>
<td>Binary trees</td>
<td></td>
</tr>
<tr>
<td></td>
<td>knockout tournaments and, 47</td>
</tr>
<tr>
<td></td>
<td>properties of, summarized, 56</td>
</tr>
<tr>
<td>Binomial coefficient</td>
<td>52</td>
</tr>
<tr>
<td>Bin packing problem</td>
<td>239, 261</td>
</tr>
<tr>
<td>Bipartite matching problems</td>
<td>284</td>
</tr>
<tr>
<td></td>
<td>applications of, 221</td>
</tr>
<tr>
<td></td>
<td>augmenting path, 211–212</td>
</tr>
<tr>
<td></td>
<td>canceling, 211</td>
</tr>
</tbody>
</table>
INDEX

Bipartite matching problems (cont.)
   definition, 210
   Gale-Shapley algorithm, 211
   graphs, 76, 210
   iterative improvement, 210
   stable marriage problem, 210
   tr function, 211
   zigzagging/cycle path, 211
Bisection search tree, 49, 262
Black boxes
   floating-point numbers, 34, 36
   hidden performance traps, 34
   redundancy, 34
Bogosort algorithm, 51
Boolean satisfiability (SAT), 235
Boost.Python tool, 256
Branch-and-bound (B&B) technique, 247–249, 262
Breadth-first search (BFS) algorithm, 107, 215, 217, 262
   A* algorithm, 206
   Dijkstra’s algorithm, 196
   double-ended queue, 109
   first-in first-out queue, 108
Bucket sort algorithm, 80, 262
Bunch pattern, 32
Busacker–Gowen algorithm, 220, 263

C

Celebrity problem
   brute-force solution, 80
   naive_celeb function, 80
   solution, 81
   sparse graph, 81
Cheapest augmenting path, 220
Cheapest maximum flow, 219
Choosing representatives problem, 222
Christofides algorithm, 263
Cinpy package, 257
Circuit-SAT problem, 235, 261
Clique cover problem, 241
Clique graph, 259
Closest pair problems, 45
Collections.Counter, 273
Combinations, 51
Combinatorial problems, 43, 45
Complexity, 238
Compression, 259
Conjunctive normal form (CNF), 235
Consistent matrix rounding, 223
Convex hulls, 259
Cook-Levin theorem, 235
CorePy module, 257
Cost scaling algorithm (CSA), 224
Counting sort algorithm, 263
Ctypes module, 257
Cython, 256

D

Deadlocks, 80
Depth-first search (DFS), 263–264, 278–279
   directed graphs, 103
   iterative, 102
   preorder/postorder processing, 105
   queue, 102
   with timestamps, 104
   topological sorting, 104
   topological sorting with, 265
Digraphs, 267
Dijkstra’s algorithm, 262–263
Dinic’s algorithm, 224
Directed acyclic graphs (DAGs), 82, 263, 271,
   276–277, 283, 286
Discrete mathematics, 65
Disjoint paths
   augmenting path, s-t network, 213–214
   canceling, 213
   edge connectivity, 212
   Menger’s theorem, 215
   rules, 213
   source and sink, 212–213
   statements, 214
   vertex-disjoint, 213
distributivity, 44
Divide-and-conquer (D&C) algorithms, 279–280
   balance, 115
   balanced decomposition, 116
   balancing methods, 131
   AA-tree, 133
   binary tree, 134
   node rotations, 132
   node splitting, 132
   node types, 132
   pseudonode, 133
   binary search, 118–119
   binary trees
      Binary heaps, 135
      insertion, 121
      linear scan, 120
      mapping, 120
      tree property, 120
   convex hulls, 129–130
   dicts, 122
   hashing, 122
   heapq module, 135
   implementation of, 118
   loglinear, 127
   merge sort, 118
   priority queue, 135
   processors, 131
   puzzle-solving, 127
   RCS, 119
   recurrences, 56, 59
recursive calls, 116
selection algorithm
  heap, 123
  linear time, 123
  pivot, 124
skyline problem, 117
slicing, 130
sorted arrays, 122
sorting
  heapsort, 136
  merge sort, 125
  quicksort, 125
  stirling’s approximation, 127
  Timsort, 126
unbalanced decomposition, 116
Doctors on vacation problem, 222
Double-ended queues, 263
Dynamic arrays, 263
Dynamic programming (DP)
  binary sequence partitioning
    expected optimal search cost, 182
    matrix chain multiplication, 180
    optimal search tree problem, 181, 183
    parsing arbitrary context free languages, 181
    recursive relationships, 182
    split points, 181
  binomial coefficient calculation, 167
  brute-force solution, 164
  caching, 163
  DAG shortest path algorithm, 282
    iterative version, 171–172
    linear-time algorithm, 170
    reaching, 172
    recursive version, 171
    sequential decision process, 170
  DRY principle, 164
  Fibonacci numbers, 165
  knapsack problem
    bounded and unbounded versions, 178
    iterative solution, 180
    iterative version, 178
    memoization, 178–179
    pseudopolynomial, 178
  LCS problem
    code, 282
    comparing sequences, 175
    iterative solution, 177
    memoized implementation, 176
  longest increasing subsequence, 164, 172
    direct recursive implementation, 173
    iterative solution, 173
    recursive decomposition and induction, 174
  @memo decorator, 169
  memoization, 163
  memoizing decorator, 165
  overlapping subproblems, 166
  Pascal's triangle, 167
  path counting, 168
  recursion trees, 166
  recursive decomposition, 169
  sequence comparison, 175–176
  tangled dependencies, 166

E
  Edge cover problem, 242
  Edge-disjoint paths, 212
  Edmonds–Karp algorithm, 217, 220, 263
  Element uniqueness, 261
  Euler tours (Euler circuits), 260
  Exponential series, 49–50

F
  F2PY tool, 256
  Flow problems
    applications of, 221
    circulations, 223
    and cut problems, 260
  Floyd–Warshall algorithm, 263
  Ford–Fulkerson method, 216–218, 221, 263
  For loop, 275
  Functions, 65
G
  Gale–Shapley algorithm, 263
  Gauss, Carl Friedrich, 43, 46, 53
  Geometric series, 50
  Gnomesort algorithm, 264
  Gnomesort function, 63
  GPULib package, 256
  Graphs
    coloring, 260
    friends attribute, 23
    libraries
      Graphine, 34
      Graph-tool, 34
      NetworkX, 34
      python-graph, 34
    nutshell, 23
    representations
      adjacency lists, 24, 27, 33
      adjacency matrix, 33
      subproblem graph, 33
  Graph's nodes, 274
  Graph terminology
    acyclic graphs, 269
    antiparallel edges, 268
    cycle path, 269
    DAG, 271
    definition, 267
Graph terminology (cont.)
neighbors edges, 268
path, 269
rooted tree, 270
spanning tree, 271
supergraph, 269
trees, 270
types of, 268
underlying undirected graph, 269–270

Graph Theory, 209
Greedy algorithms, 280–281
compatible intervals, 156
exchange argument, 157
filling the puzzle, 140
knapsack problem, 143
making change problem, 140
matching problem, 141
optimal schedule, 159
optimization problem, 140
partial schedule, 158
partial solution, 141
proof by cases, 159
resource scheduling, 155
solution pieces, 139
solution without gaps/inversions, 157
stable marriage problem, 142
subset sum problem, 143
time intervals, 158
two-part approach, 157

Halting problem, 260
Hamilton cycle problem, 99, 236, 242, 245, 260, 285
Handshake problems, 45
Handshake recurrence, 277
Hard problems, 227
NPC (see NP-complete problems (NPC))
NTM, 231
in polynomial time, 231
P vs. NP, 230
reduction
binoculars, 228
Castor and Pollux, 228
complexity classes, 228
nodes represent problems, 229
zip line, 229
Hashing, 264
Hash tables, 264
Heapsort algorithm, 264
Huffman’s algorithm, 264, 281
compression application, 144
greedy choice property, 147
heap, 145
implementation, 146
optimal merging, 148
optimal substructure, 147
text compress and decompress, 146
uniform probability, 144
weighted balancing, 144

I
Icosian game, 99
Independent set problem, 241
Induction, 59, 67, 69
balance factors, 85
checkerboard puzzle, 70
correctness proofs, 86
handshake formula, 69
inductive hypothesis, 71
inductive step, 69, 71
loop invariants, 86
relaxation, 87
reverse induction, 86
Insertion sort algorithm, 74, 264
Integers programming, 260
Interpolation search algorithm, 264
Iteration method, 55
Iterative deepening depth-first search (IDDFS), 106–107

J
Johnson’s algorithm, 264
Just-in-time (JIT) compiler, 256

K
Kaliningrad, 97
k-CNF-SAT, 261
k-coloring, 240, 285
Knapsack algorithm, 285
Knapsack problem, 236, 248–249, 260
bin packing, 239
integer programming, 240
linear programming, 240
partition, 239
subset sum problem, 239
knockout tournaments, 45
König’s theorem, 212
Kosaraju’s algorithm, 264
Kruskal’s algorithm, 151, 264, 281

L
Label function, 220
linear running time, 51, 53, 64
Linked lists arrays, 264
llvm-py package, 257
logarithmic algorithms, 49
Longest increasing subsequence (LIS) problem, 260
Longest-path problem, 243
Lower bounds, 223

- **M**
  - Master theorem, 60, 61, 63
  - Matching problems, 260
  - Math module, 52
  - Max-flow problem, 260
  - Maximum flow
    - augmenting path, network, 216
    - BFS, 215, 217
    - bfs_aug function, 217
    - canceling, 216
    - capacity, 215
    - disjoint path rules, 216
    - Edmonds-Karp algorithm, 217
    - Ford-Fulkerson method, 216–218
    - labeling, 217
    - size/magnitude, 215
  - Maximum protection
    - induction
      - assumptions, 76
      - matching problem, 76
      - permutation, 76
    - recursion
      - counting sort algorithm, 79
      - first-in, first-out queue, 78
      - naive_max_perm function, 78
      - recursive algorithm, 77
      - reference counting, 78
  - Maximum tension problems, 219
  - Menger’s theorem, 215, 218
  - Merge sort algorithm, 265
  - Mergesort function, 63
  - Metaheuristics, 251
  - Metrics, 244
  - Minimum-cost flow problem, 284
  - Minimum cut problem, 284
    - applications of, 221
    - duality, 219
    - GPU processors, 219
    - statements, 218
  - Minimum spanning tree problem, 149, 261, 281
    - Euclidean graph, 149
    - Kruskal’s algorithm
      - find and union implementation, 152
      - naïve Kruskal implementation, 151
      - path compression, 152
      - representative, 151
    - Prim’s algorithm
      - breadth-first search, 154
      - heapq library, 153
      - priority-based traversals, 154
      - priority queue, 153
  - Multicommodity flow problem, 213, 224
  - Multiplicative constants, 44

- **N**
  - Nondeterministically polynomial (NP), 230, 234
  - Nondeterministic Turing machine (NTM), 231
  - NP-complete problems (NPC), 232–233, 285–286
    - Boolean expression, 237
    - clause node encoding, 238
    - clique cover, 241
    - CNF, 235
    - colorings, 240
    - Cook-Levin theorem, 235
    - Hamilton cycle problem, 236–237
    - knapsack (see Knapsack problem)
    - NP-hard, 233
    - NTM, 234
    - paths and circuits, 242
    - polynomial time reduction, 235
    - SAT problem, 235
    - transitive reductions and, 233
    - TSP, 233
    - variable, 236–237
  - NP-hard problem, 260
  - Numba, 256
  - NumPy library, 29
  - NumPy package, 255

- **O**
  - Optimal decision trees, 259
  - Ore’s algorithm, 265

- **P**
  - Partition problem, 239, 261
  - Perfect matching, 210
  - Permutations, 51
  - Polynomial (P), 230
  - Predecessor pointer, 282
  - Prerequisites, 4
  - Primality checking, 50, 65
  - Prim's algorithm, 265
  - Problems
    - clique graph, 259
    - closest pair, 259
    - compression, 259
    - convex hull, 259
    - graph coloring, 260
    - independent set, 259
    - optimal decision trees, 259
    - sequence comparison, 260
  - Problem solving advices, 89
  - Pruning, 262
  - Pseudopolynomials, 50, 251
Psyco, 256
PyInline package, 257
PyPy, 256
Pyrex, 256
PyStream package, 256
Python
  Call Graph, 20
  multiprocessing module, 255
  optimization tools, 255

Q
Quadratic running time, 35, 64

R
Radix sort algorithm, 80, 265
Randomized select algorithm, 265
Recurrences, 64
  basic, list of, 55
  checking, 58
  divide-and-conquer, 56
  relations, 43, 53
  unraveling, 54, 57
Recursion, 53, 67
  checkerboard covering
    problem, 72
  depth-first search, 73
  insertion sort, 74
  selection sort, 74–75
  tail recursion optimization, 73
  trees, 56, 61
  working principle, 71
Recursive algorithms, 43, 53, 55, 278
Reduction, 67
  contraposition, 88
  variations, 75
  working principle, 69, 276
Reference counting, 265
Residual networks, 218, 284
Revision control systems (RCSs), 118
Root node, 61
Round-robin tournaments, 45
RPython, 256
Sequence modification, 261
Set covering problem, 242, 261
Shedskin, 256
Shortest path problem
  A* algorithm, 203
    BFS, 206
      implicit graph, 205
      potential function, 204
    Rubik's Cube, 205
  Bellman-Ford algorithm, 283
    changed check, 191
    edges, 192
    logging packages, 191
    negative cycle, 190, 193–194
    relax function, 190
    weighted graph, 191
  DAG, 283
  Dijkstra's algorithm
    BFS, 196
    bidirectional version, 201–203
    dummy nodes, 196
    hidden DAG, 194, 196
    nagging doubt, 202
    traversal function, 201
    vs. Prim's algorithm, 195
  Floyd-Warshall algorithm
    distance, 200
    memory recursive, 199
    heuristic algorithm, 201
    Johnson's algorithm, 197
    relax function
      shortcut, 188
      triangle inequality, 189
Shortest-path problems, 1, 261
Sigma (Σ), 44
Sloppiness, 227
  approximation algorithm, 245–246
  Hamilton cycle problem, 245
  minimum spanning tree, 246
  TSP problem, 245–247
Sorting algorithms, 51, 63, 64, 261
Spanning tree, 271
Square root, 60, 62
Strong induction, 59
Strongly connected components (SCCs), 259
  dfs_topsort function, 110
  directed graph, 109
  Kosaraju's algorithm, 111
  latest finish time, 110
Subsets, 51
Subset sum problem, 239
Supergraph, 269
Supply and demand problem, 222
SWIG tool, 256
INDEX

T

Tail recursion optimization, 73
Timsort algorithm, 64, 265
Tools
  Sage, 52
  Wolfram Alpha computational knowledge engine, 52
Topological sorting, 262, 265, 282
  counting-based, 83
  DAG, 82–83, 276–277
  Naïve algorithm, 82
  Python’s MRO, 84
  tsort command, 82
Travelling Salesman Problem (TSP), 1, 233, 243, 245–247, 260
Traversals problem, 262
  best-first search, 111
  branch and bound approach, 111
  breadth-first search, 93, 107
  circular buffers, 109
  connected components, 93–94, 109
    adjacency sets, 95
    walk function, 96
  depth-first search, 93, 102
  dodecahedron, 98
  dungeon map, 94–95
  edge types, 105
  flood fill, 93
  fringe/frontier nodes, 94
  goal-directed, 111
  IDDFS, 106–107
  infinite mazes, 106
  keep turning left strategy, 99
    backtracking, 100
    recursive tree-traversal, 100
    Trémaux’s algorithm, 101
Königsberg bridge, 97–98
node coloring, 105
predecessors, 96
pruning, 111
shortest paths, 106
tree, 94, 271
Traversal tree, 271
Trees, 270
  binary tree, 30
  multiway tree, 31
  rooted trees, 30
Trémaux’s algorithm, 101, 265, 278
tr function, 213
Twice around the tree algorithm, 265
Two-dimensional
  adjacency array, 273

U

Unladen Swallow, 256

V

Variable changes, 60
Variable substitution, 274
Vertex cover, 241, 261
Vertex-disjoint paths, 213

W, X, Y, Z

Weak induction, 59
Weave package, 257
Wheat and chessboard problems, 49
While loop, 275
Wolfram Alpha computational knowledge engine, 52