

- abstract data type, viii, 60
  - deque, 189–190
  - dictionary, 376–377, 396–397
  - graph, 583–588
  - list, 211–215
  - map, 355–359
  - partition, 524–528
  - priority queue, 305–311
  - queue, 170–171
  - sequence, 221
  - set, 520–528
  - stack, 153–154
  - string, 8, 541–543
  - tree, 249–252
  - vector, 201–202
- abstraction, 59
- $(a, b)$  tree, 474–476
  - depth property, 475
  - size property, 475
- Ackermann function, 528
- acyclic, 612
- adaptability, 57, 58
- adaptable priority queue, 342
- adapter pattern, 94
- Adel’son-Vel’skii, 484
- adjacency list, 589, 592
- adjacency matrix, 589, 594
- adjacent, 584
- ADT, *see* abstract data type
- Aggarwal, 484
- Aho, 198, 242, 484, 538, 579
- Ahuja, 658
- algorithm, 100
- algorithm analysis, 114–127
  - average case, 116
  - worst case, 116
- alphabet, 8, 544
- amortization, 207, 524–528
- ancestor, 246, 611
- antisymmetric, 306
- API, *see* application programming interface
- application programming interface, 80, 154
- arc, 583
- Archimedes, 100, 138
- Ariadne, 582
- arithmetic, 19
- Arnold, 54, 242
- array, 32–33
- array list, 200, 201
- associative stores, 355
- asymmetric, 583
- asymptotic analysis, 122–127
- asymptotic notation, 117–122
  - big-Oh, 118–121, 124–127
  - big-Omega, 121
  - big-Theta, 122
- attribute, 602
- audit trail, 378
- AVL tree, 421–432
  - balance factor, 429
  - height-balance property, 421
- back edge, 599, 615, 616, 652
- Baeza-Yates, 484, 538, 579
- bag, 404
- balance factor, 429
- balanced search tree, 446
- Barůvka, 656, 658
- base class, 62
- base type, 6, 9, 15
- Bayer, 484
- Bentley, 352, 405
- BFS, *see* breadth-first search
- biconnected graph, 654

- big-Oh notation, 118–121, 124–127, 138
- big-Omega notation, 121
- big-Theta notation, 122
- binary recursion, 147
- binary search, 270, 382–386
- binary search tree, 410–417
  - insertion, 413
  - removal, 414
  - rotation, 425
  - trinode restructuring, 425
- binary tree, 247, 263–292, 488
  - complete, 321, 323–328
  - full, 247
  - improper, 247
  - left child, 247
  - level, 264
  - linked structure, 280–290
  - proper, 247
  - right child, 247
  - vector representation, 278–279
- binomial expansion, 661
- bipartite graph, 656
- bit vector, 534
- Booch, 98, 242
- bootstrapping, 445
- Boyer, 579
- Brassard, 138
- breadth-first search, 609–611, 616
- brute force, 544
- brute-force pattern matching, 544
- B-tree, 477–478
- bubble-sort, 241
- bucket array, 360
- bucket-sort, 515–516
- Budd, 98, 242
- buffer, 34
  
- call-by-value, 162
- Campione, 54
- Cardelli, 98, 198
- Carlsson, 352
- casting, 84–87, 160–161
  - explicit, 24
  - implicit, 24
- catch, 78
- ceiling function, 112
  
- cell, 32
- central processing unit, 138
- character-jump heuristic, 546
- Chernoff bound, 535
- child, 245
- children, 245
- Chinese Remainder Theorem, 53
- circular list, 240
- Clarkson, 538
- class, 3–10, 56, 60
- class inheritance diagram, 63
- clustering, 370
- coding, 42
- Cole, 579
- collision resolution, 361, 367–371
- Comer, 484
- comparator, 307, 308, 357, 486
- complete binary tree, 321, 323–328
- Complete Binary Tree Property:, 321
- complete graph, 651
- composition pattern, 307
- compound object, 13
- compression function, 361, 366
- concatenation, 8, 21
- conditional probability, 663
- connected components, 587, 600, 611
- constant function, 105
- constructor, 6, 13, 68
- constructor chaining, 67
- container, 242
- contradiction, 128
- contrapositive, 128
- control flow, 25–31
- Cormen, 484, 658
- Cornell, 54
- CPU, 138
- CRC cards, 43
- Crochemore, 579
- cross edge, 611, 615, 616
- cubic function, 107
- cursor, 213, 235
- cycle, 586
  - directed, 586
  
- DAG, *see* directed acyclic graph
- data packets, 241

- data structure, 100
  - secondary, 446
- debugging, 42
- decision tree, 248, 411, 513
- decoration position, 602
- decorator pattern, 602–603, 606
- decrease-and-conquer, *see* prune-and-search
- degree, 108, 584
- DeMorgan's Law, 129
- Demurjian, 98, 198
- depth, 253–256
- depth-first search, 597–609, 615
- deque, 189–193
  - abstract data type, 189–190
  - linked-list implementation, 190–193
- descendent, 246, 611
- design patterns, 43, 88–94
  - adapter, 94
  - amortization, 207
  - brute force, 544
  - comparator, 308–309
  - composition, 307
  - decorator, 602–603, 606
  - divide-and-conquer, 487–492, 501–502
  - dynamic programming, 571–574
  - greedy method, 569
  - iterator, 232–237
  - position, 211
  - prune-and-search, 529–531
  - template method, 274–278, 522, 606
- destination, 584
- DFS, *see* depth-first search
- Di Battista, 302, 658
- diameter, 299
- dictionary, 354–405
  - (2,4) tree, 443–454
  - abstract data type, 376–377, 396–397
  - AVL tree, 421–432
  - binary search tree, 410–417
  - list-based, 378–380
  - ordered, 354, 376, 396, 416
  - red-black tree, 455–471
  - search table, 382–386
  - skip list, 386–394
  - unordered, 376, 378
  - update operations, 389, 391, 413, 414, 423, 427
- digraph, 612
- Dijkstra, 658
- Dijkstra's algorithm, 628–635
- directed acyclic graph, 619–621
- directed cycle, 612
- discovery edge, 599, 611, 615, 616
- distance, 627
- divide-and-conquer, 487–492, 501–502
- division method, 366
- d*-node, 443
- dot operator, 18
- double black, 462
- double hashing, 370
- double red, 457
- double-ended queue, *see* deque
- down-heap bubbling, 330, 340
- dynamic binding, 64
- dynamic dispatch, 64
- dynamic programming, 149, 570–574, 617
- edge, 248, 583
  - destination, 584
  - end vertices, 584
  - incident, 584
  - multiple, 585
  - origin, 584
  - outgoing, 584
  - parallel, 585
  - self-loop, 585
- edge list, 589
- edge list structure, 590
- edit distance, 577
- element, 32
- encapsulation, 60
- end vertices, 584
- endpoints, 584
- ENIAC, 3, 98
- entry, 306, 307, 355
- equality tester, 357
- Euclid's Algorithm, 53
- Euler path, 648
- Euler tour, 648, 653

- Euler tour traversal, 271, 302
- Even, 658
- event, 663
- evolvability, 58
- exception, 49, 76–79
- expected value, 664
- exponent function, *see* exponential function
- exponential function, 109
- exponentiation, 145
- expression, 17–24
- extension, 66
- external memory, 473
  
- factorial, 89–90, 660
- fail fast, 240
- failure function, 551
- favorite list, 225
- Fibonacci progression, 73, 661
- field, 3, 9, 56
- FIFO, 170
- first-in first-out, 170
- Flajolet, 138
- Flanagan, 54
- floor function, 112
- Floyd, 352
- Floyd-Warshall algorithm, 617, 658
- forest, 587
- forward edge, 615
- frame, 162
- friendly, 5
- full binary tree, 247
- function, 12
- fusion, 452, 476, 477
  
- Gamma, 98
- garbage collection, 623–625
  - mark-sweep, 624
- Gauss, 106
- generic merge algorithm, 521
- geometric sum, 662
- Gibbons, 658
- Golberg, 242
- golden ratio, 661
- Gonnet, 352, 484, 538
- Gosling, 54, 242
  
- Graham, 138, 658
- graph, 582–658
  - abstract data type, 583–588
  - acyclic, 612
  - breadth-first search, 609–611, 614–616
  - connected, 587, 611
  - data structures, 589–596
    - adjacency list, 592–594
    - adjacency matrix, 594–596
    - edge list, 589–591
  - dense, 601, 619
  - depth-first search, 597–609, 614–616
  - digraph, 612
  - directed, 583, 584, 612–621
    - acyclic, 619–621
    - strongly connected, 612
  - methods, 588
  - mixed, 584
  - reachability, 612–613, 616–619
  - shortest paths, 616–619
  - simple, 585
  - sparse, 601
  - traversal, 597–611
  - undirected, 583, 584
  - weighted, 626–658
- greedy method, 569, 627, 628
- greedy-choice, 569
- Guibas, 484
- Gutttag, 98, 198, 242
  
- Harmonic number, 196, 662
- hash code, 361, 362
- hash function, 361, 370
- hash table, 360–375
  - bucket array, 360
  - capacity, 360
  - chaining, 367
  - clustering, 370
  - collision, 361
  - collision resolution, 367–371
  - double hashing, 370
  - linear probing, 369
  - open addressing, 371
  - quadratic probing, 370
  - rehashing, 375

- header, 191
- heap, 304, 320–341
  - bottom-up construction, 338–341
- heap-order property, 320
- heap-sort, 336–341, 486
- height, 253–256, 416
- height-balance property, 421, 423, 425, 427
- Hell, 658
- heuristic, 228
- hierarchy, 61
- Hoare, 538
- Hopcroft, 198, 242, 484, 538, 658
- Horner’s method, 136
- Horstmann, 54
- HTML tags, 167
- Huang, 538
- Huffman coding, 567–569
  
- implicit cast, 24
- import, 40
- improper binary tree, 247
- in-degree, 584
- in-place, 348, 510, 625
- incidence container, 592
- incident, 584
- incoming edges, 584
- independent, 663, 664
- index, 32, 355
- induction, 129–130
- information hiding, 60
- inheritance, 62–74
- inorder traversal, 410, 414, 424, 425
- insertion-sort, 318–319, 486
- instance variable, 3
- instance variable, 56
- instance variables, 9
- integrated development environment, 43
- interface, 60, 80–84, 86, 154
- Internet, 241
- inversion, 319, 535
- inversions, 518
- iterator, 200, 232–237
  
- JáJá, 302
- Jarník, 658
  
- Java, 2–54, 57–87
  - arrays, 32–33
  - casting, 84–87
  - control flow, 25–31
  - exceptions, 76–79
  - expressions, 17–24
  - input, 34–36
  - interfaces, 80–84
  - method stack, 162–163
  - methods, 11–16
  - output, 34–36
  - packages, 40–41
  - threads, 179–181
- Java stack, 162
- Java Virtual Machine, 140, 161, 178–180, 198
- javadoc, 44
- Jones, 658
- Josephus problem, 176
- JVM, *see* Java Virtual Machine
  
- Karger, 658
- Karp, 302
- key, 305, 354–356, 377, 443
- Klein, 658
- Knuth, 138, 198, 242, 302, 352, 405, 484, 538, 579, 658
- Kosaraju, 658
- Kruskal, 658
- Kruskal’s algorithm, 640–644
  
- L’Hôpital’s Rule, 665
- Landis, 484
- Langston, 538
- last node, 321
- last-in first-out, 152
- LCS, *see* longest common subsequence
- leaves, 246
- Lecroq, 579
- left child, 247
- left subtree, 247
- Leiserson, 484, 658
- level, 264, 609
- level numbering, 262, 278
- level order traversal, 300
- Levisse, 405

- lexicographical, 516
- life-critical applications, 57
- LIFO, 152
- Lindholm, 198
- linear exponential, 662
- linear function, 105
- linear probing, 369
- linearity of expectation, 531, 664
- linked list, 182–188
  - doubly linked, 190–193, 215–218, 222
  - singly linked, 182–186
- linked structure, 280
- Liskov, 98, 198, 242
- list, 210–218, 378
  - abstract data type, 211–218
- literal, 17
- literals, 17
- live objects, 624
- load factor, 368
- local variable, 15
- local variables, 15
- locality of reference, 228
- location-aware entry, 343
- log file, 378
- log-star, 528
- logarithm function, 110, 659
  - natural, 659
- longest common subsequence, 570–574
- looking-glass heuristic, 546
- loop invariant, 131
- lowest common ancestor, 299
- Magnanti, 658
- map, 356
  - abstract data type, 355–359
  - hash table, 360–375
- mark-sweep algorithm, 624
- master method, 665
- maximal independent set, 653
- McCreight, 484, 579
- McDiarmid, 352
- median, 529
- Megiddo, 538
- Mehlhorn, 484, 658
- members, 3
- memory, 138
- memory heap, 178
- memory management, 623
- merge-sort, 487–500
  - tree, 488
- mergeable heap, 482
- method, 11–16, 56
  - body, 11
  - signature, 11
- minimum spanning tree, 638–647
  - Kruskal’s algorithm, 640–644, 647
  - Prim-Jarnik algorithm, 644–645, 647
- Minotaur, 582
- mixin, 83
- modularity, 61
- modulo, 174, 660
- Moore, 579
- Morris, 579
- Morrison, 579
- Motwani, 405, 538
- move-to-front heuristic, 200, 228
- MST, *see* minimum spanning tree
- multi-way search tree, 443
- multi-way tree, 443–446
- multiple inheritance, 83
- multiple recursion, 150
- multiprogramming, 179
- Munro, 352
- mutually independent, 663
- n-log-n function, 112
- natural join, 241
- natural logarithm, 659
- nested class, 168, 225, 314, 429, 470
- node, 245, 249, 583
  - ancestor, 246
  - balanced, 423
  - child, 245
  - descendent, 246
  - external, 245
  - internal, 245
  - parent, 245
  - redundant, 560
  - root, 245
  - sibling, 245
  - size, 438

- unbalanced, 423
- nontree edge, 615, 616
- null string, 541
- number classes, 7
- numeric progression, 68
  
- object, 3–10, 56
- object-oriented design, 56–98, 198
- objects, 3
- open addressing, 369, 371
- operand stack, 165
- order statistic, 529
- ordered dictionary, 354, 376, 396
- origin, 584
- Orlin, 658
- out-degree, 584
- outgoing edge, 584
- overflow, 449, 477
- overloading, 65
- override, 65
  
- package, 5, 40–41
- palindrome, 98, 577
- parameter passing, 13
- parent, 245
- parenthetic string representation, 258
- partition, 524–528
- Patashnik, 138
- path, 248, 586
  - directed, 586
  - length, 627
  - simple, 586
- path compression, 527
- path length, 300
- pattern matching, 544–554
  - Boyer-Moore algorithm, 546–551
  - brute force, 544–545
  - Knuth-Morris-Pratt algorithm, 551–554
- polymorphic, 65
- polymorphism, 64–65
- polynomial, 108, 136
- polynomial hash code, 363
- portability, 58
- position, 211, 249, 387
- postorder traversal, 259
  
- power function, 145
- Pratt, 579
- prefix, 541
- prefix code, 567
- prefix sum, 127
- preorder traversal, 256
- Prim, 658
- Prim-Jarnik algorithm, 644–645
- primitive operations, 114–115
- primitive type, 6
- priority queue, 304–352, 486, 537
  - ADT, 310
    - heap implementation, 328–332
    - list implementation, 313–314
- priority search tree, 351
- probability, 663–665
- probability space, 663
- procedure, 12
- program counter, 162
- proper binary tree, 247
- prune-and-search, 529–531
- pseudo-code, 103–104
- pseudo-random number generators, 386
- Pugh, 405
- push, 207
  
- quadratic function, 106
- quadratic probing, 370
- queue, 170–181
  - abstract data type, 170–171
  - array implementation, 172–175
  - linked-list implementation, 188
- quick-sort, 501–512
  - tree, 502
  
- radix-sort, 516–517
- Raghavan, 405, 538
- RAM, 138
- Ramachandran, 302
- random access machine, 138
- random variable, 664
- randomization, 386, 387
- randomized quick-select, 530
- randomized quick-sort, 508
- rank, 382
- reachability, 612

- recurrence equation, 143, 500
- recursion, 89–93, 141–152, 164
  - binary, 147–149
  - higher-order, 147–152
  - linear, 142–146
  - multiple, 150–152
  - tail, 146
  - traces, 143
- recursion trace, 90
- red-black tree, 455–471
  - depth property, 455
  - external property, 455
  - internal property, 455
  - recoloring, 459
  - root property, 455
- Reed, 352
- reference, 6, 9, 15, 64
- reference type, 9
- refinement, 66
- reflexive, 306
- rehashing, 375
- relaxation, 629
- replacement, 66
- restructure, 425
- reusability, 57, 58
- Ribeiro-Neto, 579
- right child, 247
- right subtree, 247
- Rivest, 484, 658
- Robson, 242
- robustness, 57
- root, 245
- root objects, 624
- rotation, 425
  - double, 425
  - single, 425
- round robin, 176, 181
- running time, 100–102, 115–127
- Samet, 484
- sample space, 663
- scan forward, 388
- Schaffer, 352
- search engine, 520, 566
- search table, 382–386
- Sedgewick, 138, 352, 484
- seed, 386
- selection, 529–531
- selection-sort, 317–318, 486
- self-loop, 585
- sentinel, 191, 356, 377
- separate chaining, 367
- sequence, 221–224
  - abstract data type, 221
- set, 520–528
- shortest path, 627–635
  - Dijkstra's algorithm, 628–635
- sibling, 245
- sieve algorithm, 403
- signature, 11, 14, 18, 65
- singly linked list, 182–186
- skip list, 386–394
  - analysis, 392–394
  - insertion, 389
  - levels, 387
  - removal, 391–392
  - searching, 388–389
  - towers, 387
  - update operations, 389–392
- Sleator, 484
- slicing floorplan, 300
- slicing tree, 300
- sorting, 312, 486–517
  - bucket-sort, 515–516
  - heap-sort, 336–341
  - in-place, 337, 510
  - insertion-sort, 318–319
  - lower bound, 513–514
  - merge-sort, 487–500
  - priority-queue, 312
  - quick-sort, 501–512
  - radix-sort, 516–517
  - selection-sort, 317–318
  - stable, 516
- space usage, 100, 117
- spanning subgraph, 587
- spanning tree, 587, 597, 599, 600, 609, 611, 638
- sparse array, 241
- specialization, 66
- splay tree, 432–442



- split, 449, 477
- stable, 516
- stack, 152–167
  - abstract data type, 153–154
  - array implementation, 156–160
  - linked-list implementation, 186
- statement block, 15
- Stephen, 579
- Stirling's Approximation, 661
- stop words, 558, 579
- stragglings, 533
- string, 8
  - abstract data type, 8, 541–543
  - immutable, 542
  - mutable, 543
  - null, 541
  - prefix, 541
  - suffix, 541
- strong typing, 80, 84
- strongly connected, 612
- subclass, 62
- subgraph, 587
- subproblem optimization, 571
- subproblem overlap, 571
- subsequence, 570
- substring, 541
- subtree, 246
- suffix, 541
- summation, 108, 662
  - geometric, 110
- summation puzzles, 150
- superclass, 62
- symmetric, 583
  
- tags, 44
- Tamassia, 302, 658
- Tarjan, 302, 484, 658
- telescoping sum, 662
- template method pattern, 274–278, 522, 606
- testing, 42
- text compression, 567–569
- Theseus, 582
- thread, 179–181
- throw, 76
- token, 166
  
- topological ordering, 620–621
- total order, 306
- tower-of-twos, 528
- Towers of Hanoi, 196
- trailer, 191
- transfer, 452
- transitive, 306
- transitive closure, 612, 615
- traveling salesman problem, 628
- tree, 245–302, 587
  - abstract data type, 249–252
  - binary, *see* binary tree
  - binary tree representation, 291–292
  - child node, 245
  - decision, 248
  - depth, 253–256
  - edge, 248
  - external node, 245
  - height, 253–256
  - internal node, 245
  - level, 264
  - linked structure, 290–291
  - multi-way, 443–446
  - node, 245
  - ordered, 247
  - parent node, 245
  - path, 248
  - root node, 245
- tree edge, 615, 616
- tree reflection, 297
- tree traversal, 256–262, 267–278
  - Euler tour, 272–278
  - generic, 274–278
  - inorder, 269–271
  - level order, 300
  - postorder, 259–262, 267–268
  - preorder, 256–259, 267
- trie, 556–566
  - compressed, 560
  - standard, 556
- trinode restructuring, 424, 458
- try-catch block, 78
- Tsakalidis, 484
- (2,4) tree, 443–454
  - depth property, 447

- size property, 447
- type, 3, 9
  
- Ullman, 198, 242, 484, 538
- underflow, 452, 477
- union-by-size, 527
- union-find, 524–528
- unordered dictionary, 376
- up-heap bubbling, 330
- update methods, 37
  
- van Leeuwen, 658
- variable modifiers, 10
- vector, 201–209, 382
  - abstract data type, 201–202
  - implementation, 203–209
- vertex, 583
  - degree, 584
  - in-degree, 584
  - out-degree, 584
- Vishkin, 302
- Vitter, 484
  
- Walrath, 54
- Wegner, 98, 198
- Williams, 98, 352
- Wood, 242
- wrapper class, 7
  
- Yellin, 198
  
- zig, 433, 440
- zig-zag, 433, 440
- zig-zig, 432, 440