Index

9-1-1, 2
abstract data type, see interface
adjacency list, 252
adjacency matrix, 249
algorithmic complexity attack, 132
amortized cost, 21
amortized running time, 20
ancestor, 133
array
    circular, 38
ArrayDeque, 40
ArrayQueue, 36
arrays, 29
ArrayStack, 30
asymptotic notation, 12
AVL tree, 206
B*-tree, 304
B+-tree, 304
B-tree, 286
backing array, 29
Bag, 28
BDeque, 71
Bibliography on Hashing, 128
big-Oh notation, 12
binary heap, 211
binary logarithm, 10
binary search, 272, 289
binary search tree, 140
    height balanced, 206
    partial rebuilding, 173
    random, 154
    randomized, 169
    red-black, 185
    size-balanced, 148
    versus skiplist, 105
binary search tree property, 140
binary tree, 133
    complete, 215
    heap-ordered, 212
    search, 140
binary-tree traversal, 136
BinaryHeap, 211
BinarySearchTree, 140
BinaryTree, 135
BinaryTrie, 266
binomial coefficients, 12
binomial heap, 222
black node, 190
black-height property, 190
block, 283, 284
block store, 285
BlockStore, 285
borrow, 298
bounded deque, 71
Index

BPlusTree, 307
breadth-first traversal, 139
breadth-first-search, 256

celebrity, see universal sink
ChainedHashTable, 107
chaining, 107
child, 133
  left, 133
  right, 133
circular array, 38
coin toss, 17, 98
collision resolution, 128
colour, 190
Comparator, 226
compare(a, b), 226
compare(x, y), 9
comparison tree, 236
comparison-based sorting, 226
complete binary tree, 215
complexity
  space, 20
  time, 20
conflict graph, 247
connected components, 263
connected graph, 263
contact list, 1
conted B-tree, 304
correctness, 20
CountdownTree, 183
counting-sort, 239
credit invariant, 302
credit scheme, 179, 302
CubishArrayStack, 61
cuckoo hashing, 129
cycle, 247
cycle detection, 260
DaryHeap, 223
decreaseKey(u, y), 222
degree, 254
dependencies, 22
depth, 133
depth-first-search, 258
deque, 6
  bounded, 71
descendant, 133
dictionary, 8
directed edge, 247
directed graph, 247
disk access model, 304
divide-and-conquer, 226
DLLList, 67
doubly-linked list, 67
DualArrayDeque, 43
dummy node, 67
Dyck word, 28
DynamiteTree, 183
e (Euler's constant), 10
degree, 247
directed edge, 247
directed graph, 247
disk access model, 304
divide-and-conquer, 226
dllList, 67
doubly-linked list, 67
DualArrayDeque, 43
dummy node, 67
Dyck word, 28
DynamiteTree, 183
e (Euler’s constant), 10
degree, 247
directed edge, 247
directed graph, 247
disk access model, 304
divide-and-conquer, 226
dllList, 67
doubly-linked list, 67
DualArrayDeque, 43
dummy node, 67
Dyck word, 28
DynamiteTree, 183
expected cost, 21
expected running time, 17, 20
expected value, 17
exponential, 10
Ext4, 304
external memory, 283
external memory hashing, 305
external memory model, 284
external storage, 283

318
Index

Eytzinger’s method, 211
factorial, 11
family tree, 147
FastArrayStack, 35
Fibonacci heap, 222
FIFO queue, 5
file system, 1
finger, 103, 171
finger search
in a skiplist, 103
in a treap, 171
fusion tree, 281

general balanced tree, 181

FIFO queue, 5
file system, 1
finger, 103, 171
finger search
in a skiplist, 103
in a treap, 171
fusion tree, 281

H_k (harmonic number), 154
hard disk, 283
harmonic number, 154
hash code, 107, 122
for arrays, 125
for compound objects, 123
for primitive data, 123
for strings, 125
hash function
perfect, 128
hash table, 107
cuckoo, 129
two-level, 129
hash value, 107
hash(x), 107
hashing
multiplicative, 110, 129
multiply-add, 129
tabulation, 169
universal, 129
hashing with chaining, 107, 128
heap, 211
binary, 211
binomial, 222
Fibonacci, 222
leftist, 222
pairing, 222
skew, 222
heap order, 212
heap property, 159
heap-ordered binary tree, 212
heap-sort, 233
height
in a tree, 133
of a skiplist, 87
of a tree, 133
height-balanced, 206
HFS+, 304
I/O model, 304
in-order number, 148
in-order traversal, 148
in-place algorithm, 243
incidence matrix, 262
indicator random variable, 17
interface, 4
Java Collections Framework, 26
Java Runtime Environment, 60
leaf, 133
left child, 133
left rotation, 161

319
left-leaning property, 194
left-leaning red-black tree, 194
leftist heap, 222
LIFO queue, 5, see also stack
linear probing, 114
LinearHashTable, 114
linearity of expectation, 17
linked list, 63
  doubly-, 67
  singly-, 63
  space-efficient, 71
  unrolled, see also SEList
List, 6
logarithm, 10
  binary, 10
  natural, 10
lower-bound, 235
map, 8
matched string, 28
MeldableHeap, 217
memcpy(d, s, n), 36
memory manager, 60
merge, 187, 299
merge-sort, 84, 226
min-wise independence, 169
MinDeque, 85
MinQueue, 85
MinStack, 85
modular arithmetic, 37
multiplicative hashing, 110, 129
multiply-add hashing, 129
n, 22
natural logarithm, 10
no-red-edge property, 190
NTFS, 304
number
  in-order, 148
  post-order, 148
  pre-order, 148
O notation, 12
open addressing, 114, 128
Open Source, xiii
ordered tree, 133
pair, 8
pairing heap, 222
palindrome, 83
parent, 133
partial rebuilding, 173
path, 247
pedigree family tree, 147, 222
perfect hash function, 128
perfect hashing, 128
permutation, 11
  random, 154
pivot element, 230
planarity testing, 262
post-order number, 148
post-order traversal, 148
potential, 48
potential method, 48, 80, 205
pre-order number, 148
pre-order traversal, 148
prime field, 126
priority queue, 5, see also heap
probability, 15
queue
  FIFO, 5
  LIFO, 5
Index

priority, 5
quicksort, 230
radix-sort, 241
RAM, 18
random binary search tree, 154
random permutation, 154
randomization, 15
randomized algorithm, 15
randomized binary search tree, 169
randomized data structure, 15
RandomQueue, 60
reachable vertex, 247
recursive algorithm, 136
red node, 190
red-black tree, 185, 194
RedBlackTree, 194
remix, xiii
right child, 133
right rotation, 161
rooted tree, 133
RootishArrayStack, 49
rotation, 161
run, 118
running time, 20
  amortized, 20
  expected, 17, 20
  worst-case, 20
scapegoat, 173
ScapegoatTree, 174
search path
  in a BinaryTrie, 266
  in a binary search tree, 140
  in a skiplist, 88
secondary structure, 275
SEList, 71
sentinel node, 88
Sequence, 184
share, xiii
simple path/cycle, 247
singly-linked list, 63
size-balanced, 148
skew heap, 222
skiplist, 87
  versus binary search tree, 105
SkiplistList, 93
SkiplistSSet, 90
SLList, 63
social network, 1
solid-state drive, 283
sorting algorithm
  comparison-based, 226
sorting lower-bound, 235
source, 247
space complexity, 20
spanning forest, 263
speciation event, 147
species tree, 147
split, 187, 290
square roots, 56
SSet, 9
stable sorting algorithm, 241
stack, 5
std::copy(a0,a1,b), 36
Stirling’s Approximation, 11
stratified tree, 280
string
  matched, 28
strongly-connected graph, 263
successor search, 9
System.arraycopy(s,i,d,j,n), 36
tabulation hashing, \(121, 169\)  \(\text{XFastTrie}, 272\)
target, \(247\)  \(\text{XOR-list, 82}\)
tiered-vector, \(59\)  \(\text{YFastTrie, 275}\)
time complexity, \(20\)
traversal
  breadth-first, \(139\)
in-order, \(148\)
of a binary tree, \(136\)
post-order, \(148\)
pre-order, \(148\)
Treap, \(159\)
TreapList, \(172\)
tree, \(133\)
  \(d\)-ary, \(222\)
  binary, \(133\)
  ordered, \(133\)
  rooted, \(133\)
tree traversal, \(136\)
Treque, \(60\)
two-level hash table, \(129\)
underflow, \(295\)
universal hashing, \(129\)
universal sink, \(263\)
unrolled linked list, see also \(\text{SEList}\)
USet, \(8\)
van Emde Boas tree, \(280\)
vertex, \(247\)
wasted space, \(54\)
web search, \(1\)
\(\text{WeightBalancedTree}, 183\)
word, \(19\)
word-RAM, \(18\)
worst-case running time, \(20\)