



Table of Contents

Introduction	1
I The Interview	6
1 Getting Ready	7
2 Strategies For A Great Interview	13
3 Conducting An Interview	19
4 Problem Solving	22
II Problems	39
5 Primitive Types	40
5.1 Computing the parity of a word	41
5.2 Swap bits	43
5.3 Reverse bits	44
5.4 Find a closest integer with the same weight	45
5.5 Compute $x \times y$ without arithmetical operators	46
5.6 Compute x/y	47
5.7 Compute x^y	48
5.8 Reverse digits	49
5.9 Check if a decimal integer is a palindrome	50
5.10 Generate uniform random numbers	51
5.11 Rectangle intersection	52
6 Arrays	54
6.1 The Dutch national flag problem	56
6.2 Increment an arbitrary-precision integer	59
6.3 Multiply two arbitrary-precision integers	60
6.4 Advancing through an array	61
6.5 Delete duplicates from a sorted array	62

6.6	Buy and sell a stock once	63
6.7	Buy and sell a stock twice	64
6.8	Enumerate all primes to n	65
6.9	Permute the elements of an array	67
6.10	Compute the next permutation	69
6.11	Sample offline data	70
6.12	Sample online data	72
6.13	Compute a random permutation	73
6.14	Compute a random subset	74
6.15	Generate nonuniform random numbers	75
6.16	The Sudoku checker problem	77
6.17	Compute the spiral ordering of a 2D array	79
6.18	Rotate a 2D array	81
6.19	Compute rows in Pascal's Triangle	83
7	Strings	85
7.1	Interconvert strings and integers	86
7.2	Base conversion	87
7.3	Compute the spreadsheet column encoding	88
7.4	Replace and remove	89
7.5	Test palindromicity	90
7.6	Reverse all the words in a sentence	91
7.7	Compute all mnemonics for a phone number	92
7.8	The look-and-say problem	93
7.9	Convert from Roman to decimal	94
7.10	Compute all valid IP addresses	95
7.11	Write a string sinusoidally	96
7.12	Implement run-length encoding	97
7.13	Find the first occurrence of a substring	98
8	Linked Lists	100
8.1	Merge two sorted lists	102
8.2	Reverse a single sublist	103
8.3	Test for cyclicity	104
8.4	Test for overlapping lists—lists are cycle-free	106
8.5	Test for overlapping lists—lists may have cycles	107
8.6	Delete a node from a singly linked list	109
8.7	Remove the k th last element from a list	109
8.8	Remove duplicates from a sorted list	110
8.9	Implement cyclic right shift for singly linked lists	111
8.10	Implement even-odd merge	112
8.11	Test whether a singly linked list is palindromic	113
8.12	Implement list pivoting	114
8.13	Add list-based integers	115
9	Stacks and Queues	117
9.1	Implement a stack with max API	118

9.2	Evaluate RPN expressions	120
9.3	Test a string over "{,},(,),[,]" for well-formedness	122
9.4	Normalize pathnames	123
9.5	Search a postings list	124
9.6	Compute buildings with a sunset view	126
9.7	Compute binary tree nodes in order of increasing depth	128
9.8	Implement a circular queue	130
9.9	Implement a queue using stacks	131
9.10	Implement a queue with max API	132
10	Binary Trees	135
10.1	Test if a binary tree is height-balanced	137
10.2	Test if a binary tree is symmetric	139
10.3	Compute the lowest common ancestor in a binary tree	140
10.4	Compute the LCA when nodes have parent pointers	141
10.5	Sum the root-to-leaf paths in a binary tree	142
10.6	Find a root to leaf path with specified sum	143
10.7	Implement an inorder traversal without recursion	144
10.8	Implement a preorder traversal without recursion	145
10.9	Compute the k th node in an inorder traversal	146
10.10	Compute the successor	146
10.11	Implement an inorder traversal with $O(1)$ space	148
10.12	Reconstruct a binary tree from traversal data	149
10.13	Reconstruct a binary tree from a preorder traversal with markers	151
10.14	Form a linked list from the leaves of a binary tree	152
10.15	Compute the exterior of a binary tree	152
10.16	Compute the right sibling tree	154
10.17	Implement locking in a binary tree	155
11	Heaps	158
11.1	Merge sorted files	159
11.2	Sort an increasing-decreasing array	161
11.3	Sort an almost-sorted array	162
11.4	Compute the k closest stars	163
11.5	Compute the median of online data	165
11.6	Compute the k largest elements in a max-heap	166
11.7	Implement a stack API using a heap	167
12	Searching	169
12.1	Search a sorted array for first occurrence of k	171
12.2	Search a sorted array for entry equal to its index	173
12.3	Search a cyclically sorted array	174
12.4	Compute the integer square root	175
12.5	Compute the real square root	176
12.6	Search in a 2D sorted array	178
12.7	Find the min and max simultaneously	179
12.8	Find the k th largest element	180

12.9	Find the missing IP address	182
12.10	Find the duplicate and missing elements	184
13	Hash Tables	187
13.1	Test for palindromic permutations	191
13.2	Is an anonymous letter constructible?	192
13.3	Implement an ISBN cache	193
13.4	Compute the LCA, optimizing for close ancestors	195
13.5	Compute the k most frequent queries	196
13.6	Find the nearest repeated entries in an array	196
13.7	Find the smallest subarray covering all values	197
13.8	Find smallest subarray sequentially covering all values	200
13.9	Find the longest subarray with distinct entries	202
13.10	Find the length of a longest contained interval	203
13.11	Compute the average of the top three scores	205
13.12	Compute all string decompositions	206
13.13	Test the Collatz conjecture	207
13.14	Implement a hash function for chess	209
14	Sorting	211
14.1	Compute the intersection of two sorted arrays	213
14.2	Merge two sorted arrays	214
14.3	Remove first-name duplicates	215
14.4	Render a calendar	216
14.5	Merging intervals	218
14.6	Compute the union of intervals	219
14.7	Partitioning and sorting an array with many repeated entries	221
14.8	Team photo day—1	223
14.9	Implement a fast sorting algorithm for lists	225
14.10	Compute a salary threshold	226
15	Binary Search Trees	228
15.1	Test if a binary tree satisfies the BST property	230
15.2	Find the first key greater than a given value in a BST	232
15.3	Find the k largest elements in a BST	233
15.4	Compute the LCA in a BST	234
15.5	Reconstruct a BST from traversal data	235
15.6	Find the closest entries in three sorted arrays	238
15.7	Enumerate numbers of the form $a + b\sqrt{2}$	239
15.8	The most visited pages problem	242
15.9	Build a minimum height BST from a sorted array	243
15.10	Insertion and deletion in a BST	244
15.11	Test if three BST nodes are totally ordered	246
15.12	The range lookup problem	248
15.13	Add credits	250

16 Recursion	253
16.1 The Towers of Hanoi problem	254
16.2 Generate all nonattacking placements of n -Queens	256
16.3 Generate permutations	258
16.4 Generate the power set	259
16.5 Generate all subsets of size k	261
16.6 Generate strings of matched parens	262
16.7 Generate palindromic decompositions	264
16.8 Generate binary trees	265
16.9 Implement a Sudoku solver	266
16.10 Compute a Gray code	268
16.11 Compute the diameter of a tree	270
17 Dynamic Programming	272
17.1 Count the number of score combinations	275
17.2 Compute the Levenshtein distance	277
17.3 Count the number of ways to traverse a 2D array	280
17.4 Compute the binomial coefficients	282
17.5 Search for a sequence in a 2D array	283
17.6 The knapsack problem	285
17.7 The bedbathandbeyond.com problem	287
17.8 Find the minimum weight path in a triangle	290
17.9 Pick up coins for maximum gain	291
17.10 Count the number of moves to climb stairs	293
17.11 The pretty printing problem	294
17.12 Find the longest nondecreasing subsequence	296
18 Greedy Algorithms and Invariants	299
18.1 Compute an optimum assignment of tasks	300
18.2 Schedule to minimize waiting time	301
18.3 The interval covering problem	302
18.4 The 3-sum problem	305
18.5 Find the majority element	306
18.6 The gasup problem	307
18.7 Compute the maximum water trapped by a pair of vertical lines	309
18.8 Compute the largest rectangle under the skyline	311
19 Graphs	314
19.1 Search a maze	318
19.2 Paint a Boolean matrix	319
19.3 Compute enclosed regions	321
19.4 Deadlock detection	324
19.5 Clone a graph	325
19.6 Making wired connections	326
19.7 Transform one string to another	327
19.8 Team photo day—2	330
19.9 Compute a shortest path with fewest edges	331

20	Parallel Computing	334
20.1	Implement caching for a multithreaded dictionary	335
20.2	Analyze two unsynchronized interleaved threads	337
20.3	Implement synchronization for two interleaving threads	338
20.4	Implement a thread pool	339
20.5	Deadlock	341
20.6	The readers-writers problem	342
20.7	The readers-writers problem with write preference	343
20.8	Implement a Timer class	344
20.9	Test the Collatz conjecture in parallel	344
III	Domain Specific Problems	346
21	Design Problems	347
21.1	Design a spell checker	348
21.2	Design a solution to the stemming problem	349
21.3	Plagiarism detector	350
21.4	Pair users by attributes	350
21.5	Design a system for detecting copyright infringement	352
21.6	Design TeX	352
21.7	Design a search engine	353
21.8	Implement PageRank	354
21.9	Design TeraSort and PetaSort	355
21.10	Implement distributed throttling	356
21.11	Design a scalable priority system	356
21.12	Create photomosaics	357
21.13	Implement Mileage Run	358
21.14	Implement Connexus	359
21.15	Design an online advertising system	359
21.16	Design a recommendation system	360
21.17	Design an optimized way of distributing large files	361
21.18	Design the World Wide Web	362
21.19	Estimate the hardware cost of a photo sharing app	362
22	Language Questions	364
22.1	References and pointers	364
22.2	Pass-by-reference vs. pass-by-value	364
22.3	Smart pointers	365
22.4	Iterators	366
22.5	Constructors	366
22.6	Default methods	367
22.7	malloc(), free(), new, delete	368
22.8	Strings	368
22.9	push_back() and emplace_back()	369
22.10	Updating a map	369
22.11	Fast function calls	370

22.12	Template functions	370
22.13	Run-time type identification	371
22.14	Dynamic linkage	373
23	Object-Oriented Design	374
23.1	Template Method vs. Strategy	374
23.2	Observer pattern	375
23.3	Push vs. pull observer pattern	375
23.4	Singletons and Flyweights	376
23.5	Adapters	376
23.6	Creational Patterns	377
23.7	Libraries and design patterns	379
24	Common Tools	380
24.1	Merging in a version control system	380
24.2	Hooks	382
24.3	Is scripting is more efficient?	383
24.4	Polymorphism with a scripting language	384
24.5	Dependency analysis	384
24.6	ANT vs. Maven	385
24.7	SQL vs. NoSQL	386
24.8	Normalization	386
24.9	SQL design	387
24.10	IP, TCP, and HTTP	387
24.11	HTTPS	388
24.12	DNS	389
IV	The Honors Class	390
25	Honors Class	391
25.1	Compute the greatest common divisor 🧐	391
25.2	Find the first missing positive entry 🧐	392
25.3	Buy and sell a stock k times 🧐	394
25.4	Compute the maximum product of all entries but one 🧐	394
25.5	Compute the longest contiguous increasing subarray 🧐	396
25.6	Rotate an array 🧐	398
25.7	Identify positions attacked by rooks 🧐	399
25.8	Justify text 🧐	401
25.9	Implement list zipping 🧐	402
25.10	Copy a postings list 🧐	403
25.11	Compute the longest substring with matching parens 🧐	405
25.12	Compute the maximum of a sliding window 🧐	406
25.13	Implement a postorder traversal without recursion 🧐	408
25.14	Compute fair bonuses 🧐	410
25.15	Search a sorted array of unknown length 🧐	412
25.16	Search in two sorted arrays 🧐	413

25.17	Find the k th largest element—large n , small k 🧐	414
25.18	Find an element that appears only once 🧐	415
25.19	Find the line through the most points 🧐	416
25.20	Find the shortest unique prefix 🧐	419
25.21	Find the most visited pages in a window 🧐	421
25.22	Convert a sorted doubly linked list into a BST 🧐	422
25.23	Convert a BST to a sorted doubly linked list 🧐	423
25.24	Merge two BSTs 🧐	424
25.25	The view from above 🧐	426
25.26	Implement regular expression matching 🧐	428
25.27	Synthesize an expression 🧐	431
25.28	Count inversions 🧐	433
25.29	Draw the skyline 🧐	435
25.30	Measure with defective jugs 🧐	437
25.31	Compute the maximum subarray sum in a circular array 🧐	438
25.32	Determine the critical height 🧐	440
25.33	Find the maximum 2D subarray 🧐	442
25.34	Implement Huffman coding 🧐	444
25.35	Trapping water 🧐	448
25.36	Search for a pair-sum in an abs-sorted array 🧐	449
25.37	The heavy hitter problem 🧐	451
25.38	Find the longest subarray whose sum $\leq k$ 🧐	453
25.39	Road network 🧐	454
25.40	Test if arbitrage is possible 🧐	456

V Notation, and Index 458

Notation 459

Index of Terms 461