

Table of Contents

I	The Interview	6
1	Getting Ready	7
2	Strategies For A Great Interview	12
3	Conducting An Interview	19
4	Problem Solving Patterns	23
II	Problems	44
5	Primitive Types	45
5.1	Compute parity	45
5.2	Swap bits	46
5.3	Reverse bits	46
5.4	Find a closest integer with the same weight 🐼	46
5.5	Compute $x \times y$ without multiply or add	46
5.6	Compute x/y 🐼	47
5.7	Compute x^y	47
5.8	Convert base	47
5.9	Compute the spreadsheet column encoding	47
5.10	Reverse digits	47
5.11	Check if a decimal integer is a palindrome	48
5.12	Generate uniform random numbers	48
5.13	Check if rectangles intersect	48
5.14	The open doors problem	48
5.15	Compute the greatest common divisor 🐼	48
6	Arrays	49
6.1	The Dutch national flag problem	49
6.2	Increment a BigInteger	50

6.3	Multiply two BigIntegers	50
6.4	Check if a board game is winnable	50
6.5	Delete a key from an array	51
6.6	Delete duplicates from a sorted array	51
6.7	Find the first missing positive entry	51
6.8	Compute the max difference	51
6.9	Solve generalizations of max difference 🐼	52
6.10	Compute the maximum product of all but one entries 🐼	52
6.11	Compute the longest contiguous increasing subarray 🐼	52
6.12	Enumerate all primes to n 🐼	53
6.13	Permute the elements of an array 🐼	53
6.14	Compute the next permutation	53
6.15	Rotate an array 🐼	53
6.16	Sample offline data	54
6.17	Compute a random permutation	54
6.18	Compute a random subset of $\{0, 1, \dots, n - 1\}$	54
6.19	Sample online data	54
6.20	Generate nonuniform random numbers	54
6.21	The Sudoku checker problem	55
6.22	Print a 2D array in spiral order	55
6.23	Rotate a 2D array	56
6.24	Compute rows in Pascal's Triangle	56
6.25	Identify positions attacked by rooks 🐼	57
6.26	Identify the celebrity 🐼	57
7	Strings	58
7.1	Interconvert strings and integers	58
7.2	Replace and remove	58
7.3	Test palindromicity	59
7.4	Reverse all the words in a sentence	59
7.5	Compute all mnemonics for a phone number	59
7.6	The look-and-say problem	60
7.7	Convert from Roman to decimal	60
7.8	Compute all valid IP addresses	60
7.9	Write a string sinusoidally	61
7.10	Implement run-length encoding	61
7.11	Implement Elias gamma encoding	61
7.12	Implement the UNIX tail command	62
7.13	Left-justify text 🐼	62
7.14	Find the first occurrence of a substring 🐼	62
8	Linked Lists	63
8.1	Merge two sorted lists	64
8.2	Reverse a singly linked list	64

8.3	Reverse a single sublist	65
8.4	Reverse sublists k at a time	65
8.5	Test for cyclicity	65
8.6	Test for overlapping lists—lists are cycle-free	65
8.7	Test for overlapping lists—lists may have cycles	66
8.8	Delete a node from a singly linked list	66
8.9	Remove the k -th last element from a list	66
8.10	Remove duplicates from a sorted list	67
8.11	Implement cyclic right shift for singly linked lists	67
8.12	Implement even-odd merge	67
8.13	Implement list zipping 🧐	68
8.14	Copy a postings list 🧐	68
8.15	Test whether a singly linked list is palindromic	68
8.16	Compute the median of a sorted circular linked list	69
8.17	Implement list pivoting	69
8.18	Sort a list	69
8.19	Add list-based integers	70
9	Stacks and Queues	71
9.1	Implement a stack with max API	71
9.2	Evaluate RPN expressions	72
9.3	Test if parens, brackets, and braces are matched	72
9.4	Compute the longest substring with matching parens 🧐	72
9.5	Normalize pathnames	72
9.6	Print the keys in a BST	73
9.7	Search a postings list	73
9.8	Compute buildings with a sunset view	73
9.9	Sort a stack	74
9.10	Print a binary tree in order of increasing depth	74
9.11	Implement a circular queue	75
9.12	Implement a queue API using two stacks	75
9.13	Implement a queue with max API 🧐	75
9.14	Compute the maximum of a sliding window 🧐	75
9.15	Compute the minimum number of multiplications to evaluate x^n	76
10	Binary Trees	77
10.1	Test if a binary tree is balanced	79
10.2	Find k -unbalanced nodes	80
10.3	Test if a binary tree is symmetric	80
10.4	Compute the LCA in a binary tree	80
10.5	Compute the LCA when nodes have parent pointers	81
10.6	Sum the leaves in a binary tree encoding integers	81
10.7	Find a root to leaf path with specified sum	82
10.8	Compute the k -th node in an inorder traversal	82

10.9	Implement an inorder traversal with $O(1)$ space	82
10.10	Implement preorder and postorder traversals without recursion 🧐	82
10.11	Compute the successor	83
10.12	Reconstruct a binary tree from traversal data	83
10.13	Reconstruct a binary tree from a preorder traversal with marker	83
10.14	Form a linked list from the leaves of a binary tree	84
10.15	Compute the exterior of a binary tree 🧐	84
10.16	Compute right siblings	84
10.17	Implement locking in a binary tree	84
11	Heaps	86
11.1	Merge sorted files	86
11.2	Sort a k -increasing-decreasing array	87
11.3	Sort an almost-sorted array	87
11.4	Compute the k closest stars	87
11.5	Compute the median of online data	87
11.6	Compute the k largest elements in a max-heap	88
11.7	Compute fair bonuses 🧐	88
11.8	Find k elements closest to the median 🧐	88
11.9	Test if x is bigger than the k -th largest element 🧐	88
11.10	Implement stack and queue APIs using heaps	89
12	Searching	90
12.1	Search a sorted array for first occurrence of k	92
12.2	Search a sorted array for the first element greater than k	92
12.3	Search a sorted array for $A[i] = i$	93
12.4	Search a cyclically sorted array	93
12.5	Search a sorted array of unknown length 🧐	93
12.6	Compute the integer square root	93
12.7	Compute the real square root	93
12.8	Search in two sorted arrays 🧐	94
12.9	Search in a 2D sorted array 🧐	94
12.10	Find the min and max simultaneously	94
12.11	Find the k -th largest element	94
12.12	Compute the optimum mailbox placement	95
12.13	Find the k -th largest element—large n , small k 🧐	95
12.14	Find the missing IP address	95
12.15	Find the duplicate and missing elements	95
12.16	Find an element that appears only once 🧐	96
13	Hash Tables	97
13.1	Partition into anagrams	98
13.2	Test for palindromic permutations	98
13.3	Test if an anonymous letter is constructible	99
13.4	Implement an ISBN cache	99

13.5	Compute the LCA, optimizing for close ancestors	99
13.6	Compute the K most frequent queries	99
13.7	Find the line through the most points 🐼	99
13.8	Find the nearest repeated entries in an array	100
13.9	Find the smallest subarray covering all values 🐼	100
13.10	Find smallest subarray that sequentially covering all values 🐼	100
13.11	Find the longest subarray with distinct entries 🐼	101
13.12	Find the length of a longest contained range 🐼	101
13.13	Compute all string decompositions	101
13.14	Find a highest affinity pair	101
13.15	Pair users by attributes	102
13.16	Test the Collatz conjecture	102
13.17	Implement a hash function for chess	102
13.18	Find the shortest unique prefix 🐼	103
14	Sorting	104
14.1	Compute the intersection of two sorted arrays	105
14.2	Implement mergesort in-place	105
14.3	Count the frequencies of characters in a sentence	105
14.4	Find unique elements	106
14.5	Render a calendar	106
14.6	Add a closed interval	106
14.7	Compute the union of intervals	107
14.8	The interval covering problem	107
14.9	Compute an optimum assignment of tasks 🐼	107
14.10	Implement counting sort 🐼	108
14.11	Team photo day—1	108
14.12	Implement a fast sorting algorithm for lists	109
14.13	Compute the smallest nonconstructible change 🐼	109
14.14	Compute a salary threshold	109
14.15	Implement a variable-length sort	109
14.16	Implement a least-distance sort	109
14.17	Schedule time trials	110
14.18	Find the winner and runner-up	110
15	Binary Search Trees	111
15.1	Test if a binary tree satisfies the BST property	111
15.2	Find the first occurrence of k in a BST	112
15.3	Find the first key larger than k in a BST	112
15.4	Find the k largest elements in a BST	113
15.5	Compute the LCA in a BST	113
15.6	Reconstruct a BST from traversal data	113
15.7	Compute the closest entries in three sorted arrays	113
15.8	The most visited pages problem	114

15.9	Find the most visited pages in a window 🧐	114
15.10	Build a BST from a sorted array	114
15.11	Convert a sorted doubly linked list into a BST 🧐	114
15.12	Convert a BST to a sorted doubly linked list 🧐	115
15.13	Merge two BSTs 🧐	115
15.14	Update a BST 🧐	116
15.15	Test if three BST nodes are totally ordered	116
15.16	Test if a binary tree is an almost BST 🧐	116
15.17	Compute the average of the top three scores	117
15.18	The nearest restaurant problem	117
15.19	Compute the view from above 🧐	118
15.20	Test if a binary tree is a min-first BST	118
15.21	Add credits	119
15.22	Count the number of entries in an interval	119
16	Recursion	120
16.1	The Towers of Hanoi problem	120
16.2	Implement regular expression matching 🧐	121
16.3	Enumerate all nonattacking placements of n -Queens	122
16.4	Enumerate permutations	122
16.5	Enumerate the power set	123
16.6	Enumerate all subsets of size k	123
16.7	Enumerate strings of balanced parens	124
16.8	Enumerate palindromic decompositions	124
16.9	Enumerate binary trees	124
16.10	Implement a Sudoku solver	124
16.11	Compute a Gray code	124
16.12	Synthesize an expression	125
16.13	Count inversions 🧐	126
16.14	Compute the diameter of a tree	126
16.15	Draw the skyline 🧐	126
16.16	Find the two closest points 🧐	127
17	Dynamic Programming	128
17.1	Count the number of score combinations	130
17.2	Compute the Levenshtein distance	130
17.3	Compute the binomial coefficients	130
17.4	Count the number of ways to traverse a 2D array	131
17.5	Plan a fishing trip	131
17.6	Search for a sequence in a 2D array	132
17.7	The knapsack problem	132
17.8	Measure with defective jugs 🧐	133
17.9	Test if a tie is possible	133
17.10	Divide the spoils fairly	134

17.11	Compute the maximum subarray sum in a circular array 🧐 . . .	134
17.12	The bedbathandbeyond.com problem	134
17.13	Determine the critical height 🧐	134
17.14	Find the maximum weight path in a triangle	135
17.15	Pick up coins for maximum gain	135
17.16	Decompose into palindromic strings	135
17.17	Test if s is an interleaving of s_1 and s_2	136
17.18	Count the number of steps in a board game	136
17.19	Compute the probability of a Republican majority	136
17.20	The pretty printing problem	136
17.21	Find the longest nondecreasing subsequence 🧐	137
17.22	Voltage selection in a logic circuit 🧐	137
17.23	Find the maximum 2D subarray 🧐	138
18	Greedy Algorithms and Invariants	139
18.1	Implement Huffman coding 🧐	140
18.2	Implement a schedule which minimizes waiting time	140
18.3	Trapping water 🧐	141
18.4	Load balancing 🧐	141
18.5	Pack for USPS priority mail 🧐	142
18.6	The 3-sum problem 🧐	142
18.7	The gasup problem	143
18.8	Enumerate numbers of the form $a + b\sqrt{2}$ 🧐	143
18.9	Find the majority element	144
18.10	Search for a pair-sum in an abs-sorted array 🧐	144
18.11	Compute the maximum water trapped by a pair of vertical lines .	144
18.12	The heavy hitter problem 🧐	144
18.13	Find the longest subarray whose sum $\leq k$ 🧐	145
18.14	Compute the largest rectangle under the skyline 🧐	146
19	Graphs	147
19.1	Search a maze	150
19.2	Paint a Boolean matrix	150
19.3	Compute enclosed regions	150
19.4	Clone a graph	152
19.5	Transform one string to another 🧐	152
19.6	Making wired connections	152
19.7	Test degrees of connectedness 🧐	152
19.8	Team photo day—2	154
19.9	Compute a minimum delay schedule, unlimited resources 🧐 . .	154
19.10	Compute a shortest path with fewest edges	154
19.11	Road network 🧐	154
19.12	Test if arbitrage is possible 🧐	155

20 Parallel Computing	156
20.1 Implement caching for a multithreaded dictionary	157
20.2 Analyze two unsynchronized interleaved threads	157
20.3 Implement synchronization for two interleaving threads	158
20.4 Implement a thread pool	158
20.5 Implement asynchronous callbacks	158
20.6 Implement a Timer class	159
20.7 The readers-writers problem	159
20.8 The readers-writers problem with write preference	159
20.9 The readers-writers problem with fairness	160
20.10 Implement a producer-consumer queue	160
20.11 Test the Collatz conjecture in parallel	160
20.12 Implement broadcast in a tree-structured network 🐼	160
20.13 Design TeraSort and PetaSort	161
20.14 Implement distributed throttling	161
21 Design Problems	162
21.1 Create photomosaics	162
21.2 Design a spell checker	162
21.3 Design a solution to the stemming problem	163
21.4 Plagiarism detector	163
21.5 Design a system for detecting copyright infringement	163
21.6 Design TeX	163
21.7 Design a search engine	163
21.8 Implement PageRank	164
21.9 Design a scalable priority system	164
21.10 Implement Mileage Run	164
21.11 Implement Connexus	165
21.12 Design an online advertising system	165
21.13 Design a recommendation system	165
21.14 Design an optimized way of distributing large files	165
21.15 Design the World Wide Web	165
21.16 Estimate the hardware cost of a photo sharing app	165
III Hints	166
IV Solutions	175
V Notation and Index	480
Index of Terms	483