 # Prerequisites:

1. Proficiency in C-programming language and well-versed in one of C++/Java/Python
(C++ and Java are more efficient, but Python is more versatile. C++ or Java is preferred over Python)
2. Well-versed with one of the standard library for data-structures provided by a programming language.
Eg: Standard Template Library or STL by C++ (STL Cheat-sheet)
3. Data-Structures course and being able to code every data-structure from scratch in C.
(Algorithmic design course is not desired for this particular problem-set)
4. All the algorithms discussed in the Data-Structures course as they are mostly not repeated here.
Eg: Evaluation of prefix expression, evaluation of arithematic expressions etc.
5. Basic time and space complexity analysis. (Basics)

#	Question	Pattern	Solution	Difficulty
	Arrays			
1	Sort an array of 0s, 1s and 2s without any sorting algorithm	Bruteforce		
2	Find factorial of a large number	Bruteforce		
3	Find the Union and Intersection of the two sorted arrays.	Bruteforce/bit-manipulation		
4	Find the maximum and minimum element in an array	Bruteforce/heap/quick-sort		
5	Find the "Kth" max and min element of an array	Bruteforce/heap/quick-sort		
6	Merge sorted arrays	merge/merge-sort		
7	Count Inversion	merge/merge-sort		
8	Count of smaller numbers after self	Variation of Count Inversion		
9	Maximum Subarray Sum	Kadane's Algorithm		
10	Maximum Subarray Product	Variation of Kadane's Algorithm		
11	Reverse the array	Bruteforce/2-pointers		
12	Valid Palindrome	2-pointers		
13	Valid Palindrome II	2-pointers		
14	Minimum swaps required bring elements less equal K together	2-pointers/sliding window		
15	Implement strStr()	Fixed-width sliding window		
16	Substrings of Size Three with Distinct Characters	Fixed-width sliding window		
17	Maximum Average Subarray I	Fixed-width sliding window		
18	Number of Sub-arrays of Size K and Average Greater than or Equal to Threshold	Fixed-width sliding window		
19	Grumpy Bookstore Owner	Fixed-width sliding window		
20	Permutation in String	Fixed-width sliding window		
21	Find All Anagrams in a String	Fixed-width sliding window		
22	Contains Duplicate II	Fixed-width sliding window		
23	Contains Duplicate III	Fixed-width sliding window		
24	Sliding Window Maximum	Fixed-width sliding window		
25	Max Consecutive Ones	Variable-width sliding window		
26	Max Consecutive Ones III	Variable-width sliding window		
27	Consecutive Characters	Variable-width sliding window		
28	Longest Substring Without Repeating Characters	Variable-width sliding window		
29	Maximum Erasure Value	Variable-width sliding window		
30	Minimum Size Subarray Sum	Variable-width sliding window		
31	Subarray Product Less Than K	Variable-width sliding window		
32	Minimum Window Substring	Variable-width sliding window		
33	Transpose Matrix	Bruteforce/Pattern-finding		
34	Convert 1D Array Into 2D Array	Bruteforce/Pattern-finding		
35	Reshape the Matrix	Bruteforce/Pattern-finding		
36	Matrix Diagonal Sum	Bruteforce/Pattern-finding		
37	Toeplitz Matrix	Bruteforce/Pattern-finding		
38	Matrix Cells in Distance Order	Bruteforce/Pattern-finding		
39	The K Weakest Rows in a Matrix	Bruteforce/Pattern-finding		
40	Count Negative Numbers in a Sorted Matrix	Bruteforce/Pattern-finding		
41	Cells with Odd Values in a Matrix	Bruteforce/Pattern-finding		
42	Spiral Matrix	Bruteforce/Pattern-finding		
43	Spiral Matrix II	Bruteforce/Pattern-finding		

44	Spiral Matrix III	Bruteforce/Pattern-finding		
45	Lucky Numbers in a Matrix	Bruteforce/Pattern-finding		
46	Special Positions in a Binary Matrix	Bruteforce/Pattern-finding		
47	Set Matrix Zeroes	Bruteforce/Pattern-finding		
48	Single Number	Bruteforce/Bit-manipulation		
49	Missing Number	Bruteforce/Bit-manipulation		
50	Set Mismatch	Cyclic Permutation		
51	Find All Numbers Disappeared in an Array	Cyclic Permutation		
52	Find All Duplicates in an Array	Cyclic Permutation/Negative-Marking		
53	First Missing Positive	Cyclic Permutation/Negative-Marking		
54	Find the Duplicate Number	Cyclic Permutation/Negative-Marking/ hare-tortoise Algorithm		
55	Array Nesting	Cyclic Permutation		
56	Rotate Array	Cyclic Permutation		
57	Rotate Image	Cyclic Permutation		
58	Determine Whether Matrix Can Be Obtained By Rotation	Cyclic Permutation/ Variation of Rotate Matrix		
59	Running Sum	Prefix Sum		
60	Range Queries 1D - Immutable	Prefix Sum		
61	Xor Queries of a subarray	Prefix Sum		
62	Range Queries 2D - Immutable	Prefix Sum		
63	Poduct of Array except self	Prefix Sum		
64	Contigious Array	Prefix Sum		
65	Subarray Sum Equals K	Prefix Sum		
66	Number of submatrices that sum to target	Prefix Sum		
67	For mutable queries and for min/max queries, prefix sum method fails to acheive efficiency. Segment Trees come handy then.	Segment Trees		
68	Arrays go well with many different data-structures(Eg: Hash-table, heaps etc) and algorithm-techniques (Eg: Greedy, DP) which will be covered in DAA course in the next semester.	NA		
	Linked List			
1	Design a Linked List	Design/Simulation		
2	Reverse a Linked List	3-pointers/Recursion		
3	Add Two Numbers	3-pointers/Recursion		
4	Swap Nodes In Pairs	2-pointers		
5	Merge Two Sorted Lists	2-pointers/merge-algorithm		
6	Merge K Sorted Lists	Heap/Sorting		
7	Rotate List	2-pointers/Recursion		
8	Remove Duplicates From Sorted List	2-pointers/Recursion		
9	Remove Duplicates From Sorted List II	2-pointers/Recursion		
10	Linked List Cycle	Slow-Fast pointer/Hashmap/ Hare-Tortoise Algorithm		
11	Linked List Cycle II	Slow-Fast pointer/Hashmap/ Hare-Tortoise Algorithm		
12	Nth node from end of Linked List	Slow-Fast pointer/Brute-Force		
13	Middle Of The Linked List	Slow-Fast pointer/Brute-Force		
14	Palindrome Linked List	Slow-Fast pointer/Stack		
15	Remove Nth Node From End Of The List	2-pointers/Recursion		
16	Sort A Linked List	Sorting/Recursion		
17	Delete Node in a Linked List	Brute-Force		
18	Remove Linked List Elements	Brute-Force/Recursion		
19	Partition List	Recursion/2-pointers		
20	Intersection Of Two Linked Lists	Stack/Recursion/2-pointers		
21	Reverse List Nodes In K Group	Recursion		
22	Flatten A Multilevel Doubly Linked List	Recursion		

23	Implement LRU Cache	List + Hash Table		
	Stacks			
1	Implement 2 stack in an array	Standard LIFO		
2	find the middle element of a stack	Standard LIFO		
3	Check the expression has valid or Balanced parenthesis or not.	Standard LIFO		
4	Reverse a String using Stack	Standard LIFO		
5	Implement Queue using Stack	Standard LIFO		
6	Design a Stack that supports getMin() in O(1) time and O(1) extra space.	Standard LIFO		
7	Remove Duplicate letters	Standard LIFO		
8	Next Greater element(NGE)	Monotonic stack		
9	Next Smaller Element(NSE)	Monotonic stack		
10	Previous Greater Element(PGE)	Monotonic stack		
11	Previous Smaller Element(PSE)	Monotonic stack		
12	Maximum Area Histogram(MAH) (Variations Trapping Rainwater and Container with most water doesn't follow this pattern. Try these!)	Variation of NSE and PSE		
13	Maximal Rectangle (A variation Maximal Square doesn't follow this pattern. Try it!)	Variation of MAH		
14	Remove K-digits	Monotonic stack		
15	Online Stock Span	Monotonic stack		
16	Daily temperatures	Monotonic stack		
	Queues			
1	Implement Queue Using Array	Design/Simulation		
2	Implement Queue using Stacks	Design/Simulation		
3	Design Circular Queue	Design/Simulation		
5	Circular Tour	2-pointers/Standard FIFO		
6	Sliding Window Maximum	Sliding-Window/Queue		
7	First Non Repeating Character In A Stream	Sliding-Window/Queue		
	Trees			
1	Binary Tree Inorder Traversal	Tree Traversal		
2	Binary Tree Preorder Traversal	Tree Traversal		
3	Binary Tree Postorder Traversal	Tree Traversal		
4	N-ary Tree Preorder Traversal	Tree Traversal		
5	N-ary Tree Postorder Traversal	Tree Traversal		
6	Binary Tree Level Order Traversal	BFS		
7	Binary Tree Zigzag Level Order Traversal	BFS		
8	N-ary Tree Level Order Traversal	BFS		
9	Binary Search Tree Iterator	Design/Simulation		
10	Diameter of a Binary tree	DFS/Recursion		
11	Kth Smallest Element in a BST	Recursion/Tree Traversal		
12	Validate A Binary Search Tree	Recursion		
13	Same Tree	DFS/BFS		
14	Symmetric Tree	DFS/BFS		
15	Construct Binary Tree from Preorder and Inorder Traversal	Recursion		
16	Balanced Binary Tree	DFS		
17	Convert Sorted Array to Binary Search Tree	DFS/Recursion		
18	Convert Sorted List to Binary Search Tree	DFS/Recursion		
19	Path Sum	DFS		
20	Path Sum II	DFS		

21	Flatten Binary Tree to Linked List	DFS/Recursion		
22	Populating Next Right Pointers in Each Node	BFS		
23	Invert Binary Tree	DFS/Recursion		
24	Binary Tree Right Side View	DFS/BFS		
25	Binary Tree Bottom View	DFS/BFS		
26	Binary Tree Top View	DFS/BFS		
27	Sum Root to Leaf Numbers	DFS		
28	Binary Tree Maximum Path Sum	DFS		
29	Lowest Common Ancestor of a Binary Search Tree	DFS/Recursion		
30	Lowest Common Ancestor of a Binary Tree	DFS/Recursion		
31	Serialize and Deserialize Binary Tree	Recursion		
32	Delete Node in a BST	Design/Simulation		
33	Inorder Successor In BST	Tree Traversal/DFS		
34	Find Duplicate Subtrees	DFS/Hashmap		
35	Two Sum IV - Input is a BST	DFS/BFS		
36	Maximum Width of Binary Tree	BFS		
37	Longest Univalue Path	DFS		
38	Vertical Order Traversal of a Binary Tree	DFS		
39	Number Of Good Leaf Pairs	DFS		