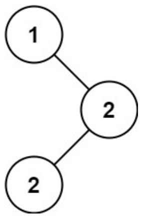
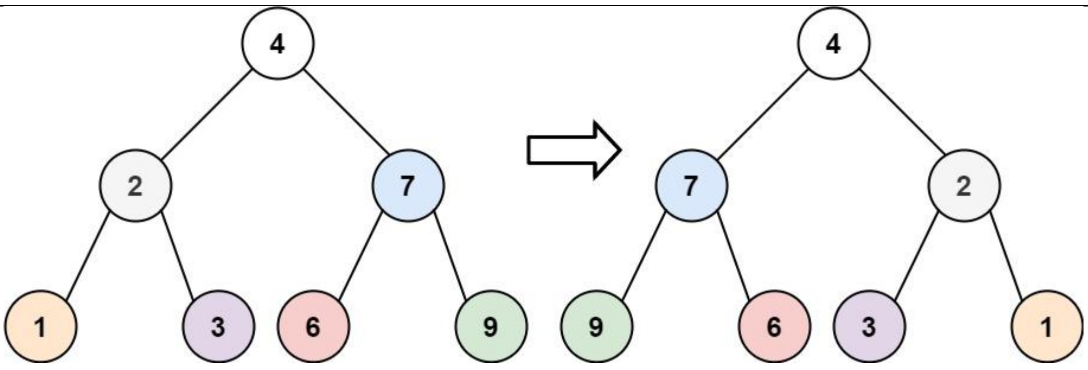
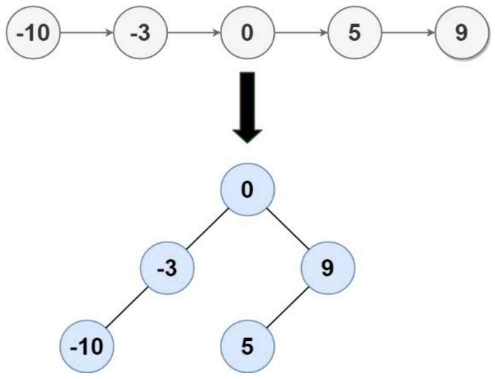
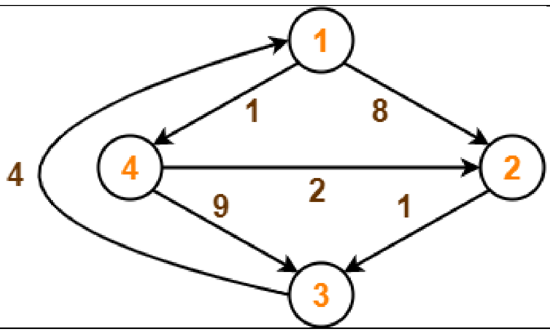
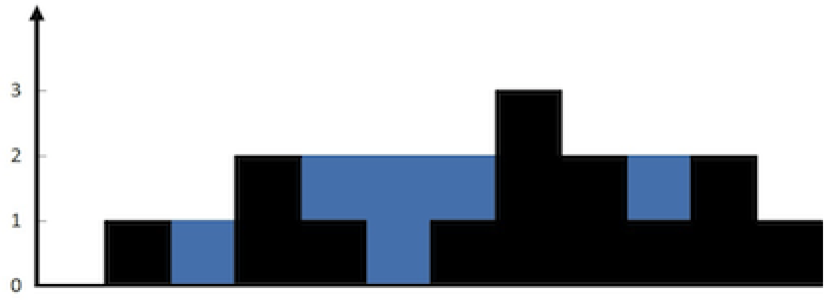
Data Structure and Algorithm Problem Statement List

Sr.	Linked List
1	WAP to sort a number in ascending order in singly linked list.
2	WAP to add an element into already sorted singly linked list.
3	WAP to demerge a doubly linked list into 2 list, one list will hold the odd numbers while another list will hold only even number
4	You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list. You may assume the two numbers do not contain any leading zero, except the number 0 itself. Input: l1 = [2,4,3], l2 = [5,6,4] Output: [7,0,8] Explanation: 342 + 465 = 807
	Stack & Queue
5	WAP to evaluate a given Prefix expression using stack.
6	WAP to convert infix expression into postfix expression.
7	WAP to Implement Stack using Queues.
8	WAP to Implement Queue using Stacks
9	Given a string s which represents an expression, evaluate this expression and return its value. Value should be taken from the user.
	Trees & Binary Search Trees
10	Print a Leaf Nodes in Binary search tree.
11	In a Given BST (Binary) search tree find the 2nd min value.
12	Given the root of a binary search tree (BST) with duplicates, return all the mode(s) (i.e., the most frequently occurred element) in it. If the tree has more than one mode, return them in any order.  Input: root = [1,null,2,2] Output: [2]
13	WAP program to Generate a Binary search tree from multiple node Value provided by user. Perform insert and delete operation on BST.
14	WAP program to perform inorder, preorder and postorder traversal in Binary search tree.
15	Given the root of a binary tree, invert the tree.

	 4 2 7 1 3 6 9 4 7 2 9 6 3 1 Input: root = [4,2,7,1,3,6,9] Output: [4,7,2,9,6,3,1]
16	Given the head of a singly linked list where elements are sorted in ascending order, convert it to a height balanced BST.  Input: head = [-10,-3,0,5,9] Output: [0,-3,9,-10,null,5] Explanation: One possible answer is [0,-3,9,-10,null,5], which represents the shown height balanced BST.
17	From a given inorder and postorder traversal built a binary tree.
	Graph Algorithms
18	WAP to implement a BFS traversal technique in a Graph.
19	WAP to implement a DFS traversal technique in a Graph.
20	WAP to count the degree of each vertex in a directed Graph (where graph is created using edges means if user input 1 2 then there is a directed edge from node 1 to node 2 user enter -1 for stop entering the edge)
21	WAP to implement a Dijkstra algorithm
22	WAP to find a minimum spanning tree of a graph using Kruskal's algorithm.
23	WAP to find a minimum spanning tree of a graph using Prim's algorithm.
24	You are given a directed graph of n nodes numbered from 0 to n - 1, where each node has at most one outgoing edge. The graph is represented with a given 0-indexed array edges of size n, indicating that there is a directed edge from node i to node edges[i]. If there is no outgoing edge from node i, then edges[i] == -1. Return the length of the longest cycle in the graph. If no cycle exists, return -1. A cycle is a path that starts and ends at the same node.
25	Solve all pair shortest path problem for the following given graph using Floyd's algorithm.

	
	Greedy Algorithms
26	There are n different online courses numbered from 1 to n. You are given an array course where courses[i] = [durationi, lastDayi] indicate that the ith course should be taken continuously for durationi days and must be finished before or on lastDayi. You will start on the 1st day and you cannot take two or more courses simultaneously. Return the maximum number of courses that you can take. Input: courses = [[100,200],[200,1300],[1000,1250],[2000,3200]] Output: 3 Explanation: There are totally 4 courses, but you can take 3 courses at most: First, take the 1st course, it costs 100 days so you will finish it on the 100th day, and ready to take the next course on the 101st day. Second, take the 3rd course, it costs 1000 days so you will finish it on the 1100th day, and ready to take the next course on the 1101st day. Third, take the 2nd course, it costs 200 days so you will finish it on the 1300th day. The 4th course cannot be taken now, since you will finish it on the 3300th day, which exceeds the closed date.
27	Using greedy algorithm find an optimal solution for knapsack instance n=7, M = 15 (P1, P2, P3, P4, P5, P6, P7) = (10,5,15,7,6,18,3) and (w1, w2, w3, w4, w5, w6, w7) = (2,3,5,7,1,4,1)
	Dynamic Programming
28	Write a program to Implement 0/1 knapsack problem using dynamic method
29	You have n super washing machines on a line. Initially, each washing machine has some dresses or is empty. For each move, you could choose any m (1 <= m <= n) washing machines, and pass one dress of each washing machine to one of its adjacent washing machines at the same time. Given an integer array machine representing the number of dresses in each washing machine from left to right on the line, return the minimum number of moves to make all the washing machines have the same number of dresses. If it is not possible to do it, return -1. Input: machines = [1,0,5] Output: 3 Explanation: 1st move: 1 0 <-- 5 => 1 1 4 2nd move: 1 <-- 1 <-- 4 => 2 1 3 3rd move: 2 1 <-- 3 => 2 2 2 Input: machines = [0,3,0] Output: 2 Explanation: 1st move: 0 <-- 3 0 => 1 2 0 2nd move: 1 2 --> 0 => 1 1 1

	You are given several boxes with different colors represented by different positive numbers. You may experience several rounds to remove boxes until there is no box left. Each time you can choose some continuous boxes with the same color (i.e., composed of k boxes, k >= 1), remove them and get k * k points. Return the maximum points you can get. Input: boxes = [1,3,2,2,2,3,4,3,1] Output: 23 Explanation: [1, 3, 2, 2, 2, 3, 4, 3, 1] ----> [1, 3, 3, 4, 3, 1] (3*3=9 points) ----> [1, 3, 3, 3, 1] (1*1=1 points) ----> [1, 1] (3*3=9 points) ----> [] (2*2=4 points)
30	You are climbing a staircase. It takes n steps to reach the top. Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?
31	Given an integer n, return the least number of perfect square numbers that sum to n. Input: n = 12 Output: 3 Explanation: 12 = 4 + 4 + 4. Input: n = 13 Output: 2 Explanation: 13 = 4 + 9.
32	You are given an integer array nums and an integer target. You want to build an expression out of nums by adding one of the symbols '+' and '-' before each integer in nums and then concatenate all the integers. For example, if nums = [2, 1], you can add a '+' before 2 and a '-' before 1 and concatenate them to build the expression "+2-1". Return the number of different expressions that you can build, which evaluates to target. Input: nums = [1,1,1,1,1], target = 3 Output: 5 Explanation: There are 5 ways to assign symbols to make the sum of nums be target 3. -1 + 1 + 1 + 1 + 1 = 3 +1 - 1 + 1 + 1 + 1 = 3 +1 + 1 - 1 + 1 + 1 = 3 +1 + 1 + 1 - 1 + 1 = 3 +1 + 1 + 1 + 1 - 1 = 3
33	Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it can trap after raining.  Input: height = [0,1,0,2,1,0,1,3,2,1,2,1]

	Output: 6 Explanation: The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.
34	(Backtracking) Given an array nums of distinct integers, return all the possible permutations. You can return the answer in any order. Input: nums = [1,2,3] Output: [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]
	Advanced Data Structures
35	WAP to implement a sliding window problem. You are given an array of integers nums, there is a sliding window of size k which is moving from the very left of the array to the very right. You can only see the k numbers in the window. Each time the sliding window moves right by one position. Input: nums = [1,3, -1, -3,5,3,6,7], k = 3 Output: [3,3,5,5,6,7] Explanation: Window position Max --------------- ----- [1 3 -1] -3 5 3 6 7 3 1 [3 -1 -3] 5 3 6 7 3 1 3 [-1 -3 5] 3 6 7 5 1 3 -1 [-3 5 3] 6 7 5 1 3 -1 -3 [5 3 6] 7 6 1 3 -1 -3 5 [3 6 7] 7
36	As the ruler of a kingdom, you have an army of wizards at your command. You are given a 0-indexed integer array strength, where strength[i] denotes the strength of the ith wizard. For a contiguous group of wizards (i.e. the wizards' strengths form a subarray of strength), the total strength is defined as the product of the following two values: The strength of the weakest wizard in the group. The total of all the individual strengths of the wizards in the group. Return the sum of the total strengths of all contiguous groups of wizards. Since the answer may be very large, return it modulo 109 + 7. A subarray is a contiguous non-empty sequence of elements within an array. Input: strength = [1,3,1,2] Output: 44 Explanation: The following are all the contiguous groups of wizards: - [1] from [1,3,1,2] has a total strength of min([1]) * sum([1]) = 1 * 1 = 1 - [3] from [1,3,1,2] has a total strength of min([3]) * sum([3]) = 3 * 3 = 9 - [1] from [1,3,1,2] has a total strength of min([1]) * sum([1]) = 1 * 1 = 1 - [2] from [1,3,1,2] has a total strength of min([2]) * sum([2]) = 2 * 2 = 4 - [1,3] from [1,3,1,2] has a total strength of min([1,3]) * sum([1,3]) = 1 * 4 = 4 - [3,1] from [1,3,1,2] has a total strength of min([3,1]) * sum([3,1]) = 1 * 4 = 4 - [1,2] from [1,3,1,2] has a total strength of min([1,2]) * sum([1,2]) = 1 * 3 = 3 - [1,3,1] from [1,3,1,2] has a total strength of min([1,3,1]) * sum([1,3,1]) = 1 * 5 = 5 - [3,1,2] from [1,3,1,2] has a total strength of min([3,1,2]) * sum([3,1,2]) = 1 * 6 = 6 - [1,3,1,2] from [1,3,1,2] has a total strength of min([1,3,1,2]) * sum([1,3,1,2]) = 1 * 7 = 7 The sum of all the total strengths is 1 + 9 + 1 + 4 + 4 + 4 + 3 + 5 + 6 + 7 = 44.

37	Design a data structure that follows the constraints of a Least Recently Used (LRU) cache. Implement the LRUCache class: • LRUCache(int capacity) Initialize the LRU cache with positive size capacity. • int get(int key) Return the value of the key if the key exists, otherwise return -1. • void put(int key, int value) Update the value of the key if the key exists. Otherwise, add the key-value pair to the cache. If the number of keys exceeds the capacity from this operation, evict the least recently used key. The functions get and put must each run in O(1) average time complexity