Topic1

Write a note on **Linked list** and add codes for the following problems:

1. Write a Code to insert a node in a sorted linked list.
2. Reverse a singly linked list.
3. Suppose there are two singly linked lists both of which intersect at some point and become a single linked list. The head or start pointers of both the lists are known, but the intersecting node is not known. Also, the number of nodes in each of the lists before they intersect is unknown and may be different in each list. List1 may have n nodes before it reaches the intersection point, and List2 might have m nodes before it reaches the intersection point where m and n may be m = n,m < n or m > n. Give an algorithm for finding the merging point.
4. How will you find the middle of the linked list?
5. How will you display a Linked List from the end?
6. Check whether the given Linked List length is even or odd?
7. If the head of a Linked List is pointing to kth element, then how will you get the elements before kth element?

Topic2

Write a note on **Linked list** and add codes for the following problems:

1. Given two sorted Linked Lists, how to merge them into the third list in sorted order?
2. Given a binary tree convert it to doubly linked list.
3. How do we sort the Linked Lists?
4. Split a Circular Linked List into two equal parts. If the number of nodes in the list are odd then make first list one node extra than second list.
5. We are given a pointer to a node (not the tail node) in a singly linked list. Delete that node from the linked list.
6. Given a linked list with even and odd numbers, create an algorithm for making changes to the list in such a way that all even numbers appear at the beginning.
7. Given a linked list, how do you modify it such that all the even numbers appear before all the odd numbers in the modified linked list?

Topic3

Write a note on **Stacks** and add codes for the following problems:

1. Discuss how stacks can be used for checking balancing of symbols
2. Given a stack, how to reverse the elements of the stack using only stack operations (push & pop)?
3. Show how to implement one queue efficiently using two stacks. Analyze the running time of the queue operations.
4. How do we implement two stacks using only one array? Our stack routines should not indicate an exception unless every slot in the array is used?
5. 3 stacks in one array: How to implement 3 stacks in one array?
6. How to implement a stack which will support following operations in O(1) time
7. complexity?

- Push which adds an element to the top of stack.
- Pop which removes an element from top of stack.
- Find Middle which will return middle element of the stack.
- Delete Middle which will delete the middle element.

Topic4

Write a note on **Queue** and add codes for the following problems:

1. Give an algorithm for reversing a queue Q. To access the queue, we are only allowed to use the methods of queue ADT.
2. How can you implement a queue using two stacks?
3. Show how you can efficiently implement one stack using two queues. Analyze the running time of the stack operations.
4. Given a queue Q containing n elements, transfer these items on to a stack S (initially empty) so that front element of Q appears at the top of the stack and the order of all other items is preserved. Using enqueue and dequeue operations for the queue, and push and pop operations for the stack, outline an efficient O(n) algorithm to accomplish theabove task, using only a constant amount of additional storage.
5. What is the most appropriate data structure to print elements of queue in reverse order?
6. Given an integer k and a queue of integers, how do you reverse the order of the first k elements of the queue, leaving the other elements in the same relative order? Fore example, if k=4 and queue has the elements [10, 20, 30, 40, 50, 60, 70, 80, 90]; the out put would be [40, 30, 20, 10, 50, 60, 70, 80, 90].

Topic5

Write a note on **Trees** and add codes for the following problems:

1. Give an algorithm for finding maximum element in binary tree.
2. Give an algorithm for finding the maximum element in binary tree without recursion.
3. Give an algorithm for searching an element in binary tree.
4. Give an algorithm for searching an element in binary tree without recursion
5. Give an algorithm for inserting an element into binary tree.
6. Give an algorithm for finding the size of binary tree.
7. Give an algorithm for deleting the tree.
8. Give an algorithm for finding the height (or depth) of the binary tree. Give an algorithm for finding the deepest node of the binary tree.

Topic6

Write a note on **Trees** and add codes for the following problems:

1. Give an algorithm for deleting an element (assuming data is given) from binary tree.
2. Give an algorithm for finding the number of leaves in the binary tree without using recursion.
3. Give an algorithm for finding the number of half nodes (nodes with only one child) in the binary tree without using recursion
4. Given two binary trees, return true if they are structurally identical

5. Give an algorithm for finding the diameter of the binary tree. The diameter of a tree (sometimes called the width) is the number of nodes on the longest path between two leaves in the tree.
6. Give an algorithm for finding the level that has the maximum sum in the binary tree
7. Given a binary tree, print out all its root-to-leaf paths.
8. Give an algorithm for checking the existence of path with given sum. That means, given a sum, check whether there exists a path from root to any of the nodes.

Topic7

Write a note on **Trees** and add codes for the following problems:

1. Give an algorithm for finding the sum of all elements in binary tree.
2. Give an algorithm for converting a tree to its mirror. Mirror of a tree is another tree with left and right children of all non-leaf nodes interchanged. The trees below are mirrors to each other.
3. Given two trees, give an algorithm for checking whether they are mirrors of each other.
4. Give an algorithm for finding LCA (Least Common Ancestor) of two nodes in a Binary Tree.
5. Give an algorithm for constructing binary tree from given Inorder and Preorder traversals.
6. Give an algorithm for printing all the ancestors of a node in a Binary tree.
7. Give an algorithm to traverse a binary tree in Zigzag order. For example, the output for the tree below should be: 1 3 2 4 5 6 7.
8. Given a binary tree with three pointers (left, right and nextSibling), give an algorithm for filling the nextSibling pointers assuming they are NULL initially.

Topic8

Write a note on **Searching** and add codes for the following problems:

Searching: Problems & Solutions

1. Given an array of n numbers, give an algorithm for checking whether there are any duplicate elements in the array or no?
2. Given an array of n numbers. Give an algorithm for finding the element which appears the maximum number of times in the array?
3. Let A be an array of n distinct integers. Suppose A has the following property: there exists an index $1 \le k \le n$ such that A[l],..., A[k] is an increasing sequence and A[k +1],..., A[n] is a decreasing sequence. Design and analyze an efficient algorithm for finding k.
4. Given a sorted array of n elements, possibly with duplicates. Find the number of occurrences of a number.
5. Finding second smallest number efficiently.
6. Given an array of 2n elements of which n elements are the same and the remaining n elements are all different. Find the majority element.
7. Given an array with 2n + 1 integer elements, n elements appear twice in arbitrary places in the array and a single integer appears only once somewhere inside. Find the lonely integer with O(n) operations and O(1) extra memory.

8. Given an array A[0...n– 1] of n numbers containing the repetition of some number. Give an algorithm for checking whether there are repeated elements or not. Assume that we are not allowed to use additional space (i.e., we can use a few temporary variables, O(1) storage).

Topic9

Write a note on **Hashing** and add codes for the following problems:

1. Sort an array of 0's, 1's and 2's [or R's, G's and B's]: Given an array A[] consisting of 0's, 1's and 2's, give an algorithm for sorting A[].The algorithm should put all 0's first, then all 1's and finally all 2's at the end. Example Input = {0,1,1,0,1,2,1,2,0,0,0,1}, Output = {0,0,0,0,0,1,1,1,1,1,2,2}
2. Given an input array of size unknown, with all numbers in the beginning and special symbols in the end. Find the index in the array from where the special symbols star
3. Given an array of 2n integers in the following format a1 a2 a3 ...an b1 b2 b3 ...bn. Shuffle the array to a1 b1 a2 b2 a3 b3 ... an bn without any extra memory.
4. Given an array A[], find the maximum j – i such that A[j] > A[i]. For example, Input: {34, 8, 10, 3, 2, 80, 30, 33, 1} and Output: 6 (j = 7, i = 1).
5. Given an array of elements, how do you check whether the list is pairwise sorted or not? A list is considered pairwise sorted if each successive pair of numbers is in sorted (non-decreasing) order.
6. Given an array of n elements, how do you print the frequencies of elements without using extra space. Assume all elements are positive, editable and less than n.
7. There are two sorted arrays A and B. The first one is of size m + n containing only m elements. Another one is of size n and contains n elements. Merge these two arrays into the first array of size m + n such that the output is sorted.