Technical Interview Answers

# Question 1:

To solve the problem, the anagram t should be converted into a linked list. Fortunately, in python a string can be treated as linked list and functions are available to remove a element of the list list.

The anagram t can be anywhere in the string s hence we must start at each letter of the string s and look for anagram t. This requires a for loop with number of characters in the string s (len\_s – len\_t) and creating a sub string starting with s[i]. Then we would search for anagram t in the substring s.

The anagram t is a linked list and whenever we would see a match between the characters of the s and the characters in the linked list t, we would pop out the character. We would do this for len\_t times to complete the search. At the end of search, we would see if there are any characters in the linked list t. If there are characters, then the result will be False or else it will be true.

The execution will be in order of (len\_s-len\_t) \* len\_t times.

# Question 2:

To find the largest palindrome, we must search the string for all sub strings and check if that is a palindrome.

We loop for each characters in the string and get all possible substrings. Then we reverse and try to match it with string. That is O(n^2) exercise.

We store the numbers and string in a dictionary so that we can find the largest number in the dictionary.

# Question 3:

Minimum spanning tree contains edges with lowest weight and connects all the nodes of the tree. To build a minimum spanning tree, we have to create a dictionary of the edges with weights as keys. Then sort the dictionary in ascending order for weights.

Create dictionary for trees and keep adding trees. For each edge, create a tree if there is no other tree connecting the nodes. If there are two trees connecting the nodes, then connect the trees and create one tree. If there is one tree with one of the nodes, then add the edge to that tree. If both the nodes are found in a single tree, then ignore the edge because it will create a loop. After we have gone through all the edges then we would have only tree left which will be minimum spanning tree.

The complexity of exercise is O(n) and n is number of branches

# Question 4:

In this question, we are given two nodes (Node A and B) and have to find the common ancestor which is farthest away from the root of the Binary Search tree. To achieve that, we start from the root (Node X) and find out if the A & B are larger or smaller than the X. If A and B are smaller than X, then we jump on the left child and make it X. If A and B are larger then we jump on right child and make it X. If X lies between A and B then we stop and X is the common ancestor. We keep on doing this process and keep on going down till we find the common ancestor.

The complexity of the exercise is O(log n)

# Question 5:

In this question, we have a singly link list and we are finding the nth element from the end of the list. In a singly linked link list, we know the head of the list and every node has pointer to the next node. We would start from the first node and store the elements in a stack till we reach the end. After we complete putting element in the stack, we would remove elements in FIFO manner and remove n elements. The nth element will be the element which we want.

The complexity of the exercise is O(n)