PRACTICE QUESTIONS

Note:

1. Many of these questions can be easily solved in higher complexity but your goal is to find an efficient solution.
2. You can easily find its solution and explanations on internet, but first try to solve them on your own.
3. The Time complexity and related Topic of each task is mentioned.
4. **Use struct Node** given at the **end** for **Linked List tasks**.

Task 1: Search Insert Position O (Log n) | Binary Search

Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.

[1,3,5,6]
5
Output: 2

[1,3,5,6]
2
Output: 1

[1,3,5,6]
7

[1,3,5,6]
0

Task 2: Squares of a Sorted Array O (N) | Two Pointer Approach

Given an integer array nums sorted in **ascending** order, return an array of **the squares of each number** sorted in ascending.

[-4,-1,0,3,10]

Output: [0,1,9,16,100]

How ? After squaring, the array becomes [16,1,0,9,100].

After sorting, it becomes [0,1,9,16,100].

[-7,-3,2,3,11]

Output: [4,9,9,49,121]

Task 3: Rotate Array 0 (N)

Given an array, rotate the array to the right by k steps.

nums = [1,2,3,4,5,6,7]

k = 3

Output: [5,6,7,1,2,3,4]

Explanation:

rotate 1 step to the right: [7,1,2,3,4,5,6]

rotate 2 steps to the right: [6,7,1,2,3,4,5]

rotate 3 steps to the right: [5,6,7,1,2,3,4]

[-1,-100,3,99]

2

Task 4: Reverse String O (N) | Two pointers

Reverse a given string.

Task 5: Middle of a Linked List O (N) | Two pointers

Write a function that takes head of a Linked List and returns the middle node of that Linked List.

Task 6: Remove Nth Node from End of List O (N) | Two pointers

Write a function that takes head of a Linked List and returns head after deleting the Nth Node from the end of Linked List.

List	N	Output
[1,2,3,4,5]	2	[1.2.3.5]
[1]	1	[]
[1.2]	1	[1]
[1,2]	2	[2]

Task 7: Reverse Linked List O (N) | Recursion

Write a function that takes head of a linked list and returns head of reversed linked list.

Task 8: Merge Two Sorted Lists O (N) | Recursion | Merge Procedure in Merge Sort

Write a function that takes heads of two linked list and returns head of merged sorted list

Input: list1 = [1,2,4], list2 = [1,3,4]

Output: [1,1,2,3,4,4]

Task 9: Median of Two Sorted Arrays O (N) | Merge Sort

Write a function that takes two sorted arrays as input and return median of these two arrays.

Input: nums1 = [1,3], nums2 = [2]

Output: 2.00000

Explanation: merged array = [1,2,3] and median is 2.

Input: nums1 = [1,2], nums2 = [3,4]

Output: 2.50000

Explanation: merged array = [1,2,3,4] and median is (2 + 3) / 2 = 2.5.

Task 10: Addition of Two Numbers O(N) | Stack

Add two numbers using stack

Task 11: Missing Number O (N) | Sum of Series

Write a function that takes an array [0 – N] and returns the number missing in that array.

Input: [3,0,1]

Output: 2

Why? since n = 3 so number that is missing from range [0 – 3] is 2

NODE

```
struct Node {
      int val;
      Node *next;
      Node(int x = 0, Node *Nnext = nullptr) {
          val = x;
          next = Nnext;
      }
 };
```