11/11/2024
ARIKALEESWARAN G
22AM002
CSE(AI&ML)

1.
Given a sorted array arr[] (with unique elements) and an integer k, find the index (0-based) of the largest element in arr[] that is less than or equal to k. This element is called the "floor" of k. If such an element does not exist, return -1.

Program:

```
class Solution {
    static int findFloor(int[] arr, int k) {

        int l=0;
        int n = arr.length;
        int r = n-1;
        int res =-1;
        while(l<=r){
            int mid =  l + (r-l)/2;
            if(arr[mid] == k){
                return mid;
            }
            if(arr[mid] < k){
                res = mid;
                l = mid+1;
            }else{
                r = mid -1;
            }
        }
        return res;
    }
}
```

TC: O(*log n*)
SC: O(1)

2.
Given two arrays arr1 and arr2 of equal size, the task is to find whether the given arrays are equal. Two arrays are said to be equal if both contain the same set of elements, arrangements (or permutations) of elements may be different though.

Program:

```
class Solution {

    public static boolean check(int[] arr1, int[] arr2) {
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        for(int i=0;i<arr1.length;i++){
            if(arr1[i] != arr2[i]) return false;
        }
        return true;
    }
}
```

TC: O(*n log n*)
SC : O(1)

3.
Given the head of a singly linked list, return true if it is a palindrome or false otherwise.

Program:

```
class Solution {
    public boolean isPalindrome(ListNode head) {
        List<Integer> lst = new ArrayList<>();
        while(head != null){
            lst.add(head.val);
            head = head.next;
        }
        int l=0;
        int r = lst.size() -1;
        while(l<r && lst.get(l) == lst.get(r)){
            l++;
            r--;
        }
        return l>=r;
    }
}
```

TC: O(n)
SC: O(n)

4.
Given an array arr of size n and an integer x. Find if there's a triplet in the array which sums up to the given integer x.

Program:

```
class Solution {
    // Should return true if there is a triplet with sum equal
    // to x in arr[], otherwise false
    public static boolean find3Numbers(int arr[], int n, int x) {
        // int n=arr.length;
        Arrays.sort(arr);
        for(int i=0;i<n-2;i++){
            int j =i+1;
            int k = n-1;
            while(j<k){
                int sum = arr[i] + arr[j] + arr[k];
                if(sum == x) return true;
                else if(sum < x){
                    j++;
                }else{
                    k--;
                }
            }
        }
        return false;
    }
}
```

TC: O(n^2)
SP: O(1)