Day-28

Python DSA

Number of Occurrence

https://www.geeksforgeeks.org/problems/number-of-occurrence2259/1

```
class Solution:
    def countFreq(self, arr, target):
        # Helper function to find first occurrence
        def first_occurrence(arr, target):
            low, high = 0, len(arr) - 1
            first = -1
            while low <= high:
                mid = (low + high) // 2
                if arr[mid] == target:
                    first = mid
                    high = mid - 1  # continue searching in left half
                elif arr[mid] < target:
                    low = mid + 1
                else:
                    high = mid - 1
            return first

        # Helper function to find last occurrence
        def last_occurrence(arr, target):
            low, high = 0, len(arr) - 1
            last = -1
```

```
        while low <= high:
            mid = (low + high) // 2
            if arr[mid] == target:
                last = mid
                low = mid + 1  # continue searching in right half
            elif arr[mid] < target:
                low = mid + 1
            else:
                high = mid - 1
        return last

    first = first_occurrence(arr, target)
    if first == -1:
        return 0  # target not found

    last = last_occurrence(arr, target)
    return last - first + 1
```

Time Complexity- O(Log N)

Space Complexity – O(1)

Leetcode 33 Search in Rotated Sorted Array

https://leetcode.com/problems/search-in-rotated-sorted-array/

```
class Solution:
    def search(self, nums: List[int], target: int) -> int:
```

```
        n= len(nums)
        low=0
        high = n-1
        while low <= high:
            mid= (low + high)//2
            if nums[mid]==target:
                return mid
            if nums[low] <= nums[mid]:
                if nums[low] <= target <= nums[mid]:
                    high = mid -1
                else:
                    low = mid +1
            else:
                if nums[mid] <= target <= nums[high]:
                    low= mid+1
                else:
                    high = mid-1
        return -1
```

TC- O(log N)

SC- O(1)

Leetcode 81 Search in Rotated Array II

https://leetcode.com/problems/search-in-rotated-sorted-array-ii/

```
class Solution:
    def search(self, nums: List[int], target: int) -> bool:
        n = len(nums)
        low = 0
        high = n - 1
        while low <= high:
            mid = (low + high) // 2
            if nums[mid] == target:
                return True
            if nums[low] == nums[mid] == nums[high]:
                high -= 1
                low += 1
                continue
            if nums[low] <= nums[mid]:
                if nums[low] <= target <= nums[mid]:
                    high = mid - 1
                else:
```

```
                    low = mid + 1
            else:
                if nums[mid] <= target <= nums[high]:
                    low = mid + 1
                else:
                    high = mid - 1

        return False
```

TC- O(log N)

SC- O(1)