Day-3

Python DSA

1. **Missing Number**

https://leetcode.com/problems/missing-number/description/

Optimal Solution

```
class Solution:
    def missingNumber(self, nums: List[int]) -> int:
        n= len(nums)
        original_total = (n * (n+1))//2
        return original_total - sum(nums)
TC -O(N)
SC- O(1)
```

2. **485 Max Consecutive Ones**

https://leetcode.com/problems/max-consecutive-ones/

Optimal

```
class Solution:
    def findMaxConsecutiveOnes(self, nums: List[int]) -> 
int:
        cnt = 0
        maxi = 0
        for i in range(len(nums)):
            if nums[i] == 1:
                cnt += 1
            else:
                maxi = max(maxi, cnt)
                cnt = 0
        return max(maxi, cnt)
```

```
TC – O(N)
SC- 0(1)
```

3. **136 Single Number**

https://leetcode.com/problems/single-number/description/

Optimal

```
class Solution:
    def singleNumber(self, nums: List[int]) -> int:
        ans=0
        for i in range(0, len(nums)):
            ans ^= nums[i]
        return ans
```

4. **Longest Array with Sum K**

https://www.geeksforgeeks.org/problems/longest-sub-array-with-sum-k0809/1

Optimal Solution

```
class Solution:
   def longestSubarray(self, arr, k):
      hash_map={}
      res=0
      Sum=0
      for i in range(len(arr)):
         Sum +=arr[i]
```

```
            if Sum==k:
                res= i+1
            elif(Sum -k) in hash_map:
                res= max(res,i-hash_map[Sum-k])
            if Sum not in hash_map:
                hash_map[Sum]=i

        return res
```

TC- O(N)

SC- O(N)

Dry Run

arr = [10, 5, 2, 7, 1, -10]

k = 15

Prefix sums:

Index 0: 10

Index 1: 15 → entire prefix is valid, res = 2

Index 2: 17 → 17 - 15 = 2 not in map

Index 3: 24 → 24 - 15 = 9 not in map

Index 4: 25 → 25 - 15 = 10 in map at index 0 → subarray [1 to 4], length 4 → res = 4

Index 5: 15 → again full prefix from 0 to 5, res = 6

Final result: 6, which is correct (entire array sums to 15)