**DSA:Day-1**

**#start with the simple question of the DSA:**

[**https://leetcode.com/problems/remove-duplicates-from-sorted-array/**](https://leetcode.com/problems/remove-duplicates-from-sorted-array/)

**Question 1:**

**1.two-sum:**

**Given an array of integers, return indices of the two numbers such that they add up to a specific target:**

**Technic used:**

**Now apply the 2 loops each represent the i and j by that find the sum of the two number**

**Input: nums = [2,7,11,15], target = 9**

**Output: [0,1]**

**Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].**

**Code output:**

def two\_sum(arr ,target):

for i in range(len(arr)):

for j in range(len(arr)-1):

if(arr[i]+arr[j]==target):

print(i ,j)

arr=[2,5,3,2,4]

target=9

two\_sum(arr,target)

0

QUE-2

<https://leetcode.com/problems/remove-duplicates-from-sorted-array/description/>

Remove duplicate from the array :

**Input:** nums = [1,1,2]

**Output:** 2, nums = [1,2,\_]

**Explanation:** Your function should return k = 2, with the first two elements of nums being 1 and 2 respectively.

It does not matter what you leave beyond the returned k (hence they are underscores).

soln:

def remove\_duplicate(array):

left=1

for i in range(1,len(array)):

if(array[i]!=array[i-1]):

array[left]=array[i]

left+=1

return left

arr=[1,1,2,3]

print(remove\_duplicate(arr))

Technic which used to slove the problem: the now start with the left and right corner with the same whenever the new unique element is coming the left increment and condition for the is the arr[i]!=arr[i-1] only this condition and inside that when this happen then arrar[[lett]=arr[i] and the left+=1

Day-1 question-3 #DSA

Rotate Array (Medium):

<https://leetcode.com/problems/rotate-array/>

**Input:** nums = [1,2,3,4,5,6,7], k = 3

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

**Explanation:**

rotate 1 steps 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]

soln:

def rotated\_array(array ,k):

l ,r = 0 , len(array)-1

k=k%len(array)

while(l<r):

array[l] ,array[r]=array[r] , array[l]

l,r=l+1 ,r-1

l ,r = 0 , k-1

while(l<r):

array[l] ,array[r]=array[r] , array[l]

l,r=l+1,r-1

l ,r = k , len(array)-1

while(l<r):

array[l] ,array[r]=array[r] , array[l]

l,r=l+1,r-1

return array

print(rotated\_array([2,3,4,5,6],3))

Day-1 question-4 #DSA

DSA#4:

#MOVE ZEROS

link=<https://leetcode.com/problems/move-zeroes/>

Given an integer array nums, move all 0's to the end of it while maintaining the relative order of the non-zero elements.

**Note** that you must do this in-place without making a copy of the array.

**Example 1:**

**Input:** nums = [0,1,0,3,12]

**Output:** [1,3,12,0,0]

soln:

def move\_zeros(array):

left=0

for r in range(len(array)):

if(array[r]):

array[left],array[r] ,array[r] , array[left]

left+=1

return array

Now the technique used to solve this particular problem is we want to move the left pointer its simple whenever the zero value is counter we swap the left and the right pointer

Day-1 question-5#DSA

#best time to buy and sell the stock

#HINT

The main aim of the stock is the we have to but the stcok at low price and we have to sell the stock at high lets solve the quetion

#now in the given problem the left represent the buy the stock and the right represent the sell th stock

link=<https://leetcode.com/problems/best-time-to-buy-and-sell-stock/>

Soln :

Def best\_time \_to\_sell(prices):

l ,r=0,1

max\_profit=0

while(r<len(prices)):

if(l<r):

profit=prices[r]-prices[l]

max\_profit=max(profit , max\_profit)

else:

l+=1

r+=1