DAY-1 :leetcode DSA →

Q.1 two sum:

Solving Approch 👍:
Simple two loop i and j
For i in range(len(arr)-1):
    For j in range(i+1 ,len(arr)):

Check by arr[left]+arr[right]==target

Code 👏:

```
def find_two_sum(arr ,target):
    for i in range(len(arr)-1):
        for j in range(i+1):
            if arr[i]+arr[j]==target:
                return i ,j
    return None
print(find_two_sum([2,4,5] ,6))
```

complexity-o(n2)

Code -2

Solving approch
Create the has table and check whether the need value is present in the hash table if it present then return values if not then add in the hash table
Needed_value =target -num

code

```
def two_sum_optimal(arr ,target):
    ht={}
    for num in arr:
        needed_value=target-num

        if needed_value in ht:
            return num , needed_value
        ht[num]=True
    return None
print(two_sum_optimal([2,4,5,6,3] ,8)),complexity=o(n)
```

Q2.check isomorphism

Solving approach

Two str if have same size then they are isomorphic also next take to value
s ,t hash table={} ,,{} and there two pointer char_s =str1[i], char_t=str2[i]

Two condition is chars in s and s[hash_s]!=char_t return false same char_t in t an t[char_t]!=char_s return false
Check this two condition if ture then the given string is isomorphic

CODE

```
def check_two_isomorphic(str1 ,str2):

    if len(str1)!=len(str2):
        return False
    s_hash,t_hash={} ,{}

    for i in range(len(str1)):
        char_s =str1[i]
        char_t=str2[i]

        if char_s in s_hash and s_hash[char_s]!=char_t:
            return False
        if char_t in t_hash and t_hash[char_t]!=char_s:
            return False

        s_hash[char_s]=char_t
        t_hash[char_t]=char_s

    return True

print(check_two_isomorphic("abba" ,"atta"))
```

Q 3 to 6 : implement bubble insertion selection sort:

```
def selection_sort(arr):
    for i in range(len(arr)-1):
        min_index=i
        for j in range(1, len(arr)):
            if arr[j]<arr[min_index]:
                min_index=j

        arr[i] , arr[min_index]=arr[min_index] ,arr[i]

    return arr
print(selection_sort([32,3,5]))

def insersion_osrt(arr):
    for i in range(1, len(arr)):
        j=i
        while(j>0 and arr[j]<arr[j-1]):
            arr[j] ,arr[j-1]=arr[j-1] ,arr[j]
            j-=1
    return arr
print(insersion_sort([3,4,5,3]))


def bubblesort(arr):
    for i in range(len(arr)-1):
        for j in range(len(arr)-i-1):
            if arr[j]>arr[j+1]:
                arr[j] ,arr[j+1] =arr[j+1] ,arr[j]
    return arr
print(bubblesort([3,4,5,2]))
```

Q7
Rotated array:

```
def rotates_array(arr ,k):

    left=0
    right=len(arr)-1
    mid=(left+right)//2
    k=k%len(arr)

    while(left<right):
        arr[left] ,arr[right] =arr[right] ,arr[left]
        left+=1
        right-=1

    left ,right=0 ,k-1
    while(left<right):
        arr[left] ,arr[right] =arr[right] ,arr[left]
        left+=1
        right-=1

    left ,right=k, len(arr)-1
    while(left<right):
        arr[left] ,arr[right] =arr[right] ,arr[left]
        left+=1
        right-=1

    return arr
print(rotates_array([2,4,5,6,3,1] ,3))
```

Simple three loop while left to right swap ,0 , k-1 ,and k to len(array)-1 and same swap just

Three sum:
Soln approach:
Simple sort the array first : select the to pointer first and last index avoid the duplication by check arr[i]==arr[i]-1 if yes then continue means skip after that while loop in which left and right pointer
Calculate the total by arr[i] arr[left] arr[right] and if total==0 append in the result list also to avoid the repetition os again while loop if the left==left+1 and right==right-1 then update the pointer left+=1 and right-=1

```
def three_sum(arr):

    arr.sort()
    result=[]
    for i in range(len(arr)-2):

        if i>0 and arr[i]==arr[i-1]:
            continue

        left ,right=arr[i+1] ,len(arr)-1

        while(left<right):
                total_sum=arr[i]+arr[left]+arr[right]
                if(total_sum==0):
                    result.append([arr[i] ,arr[left] ,arr[right]])

                    while(left<right and arr[left]==arr[left+1]):
                        left+=1

                    while(left<right and arr[right]==arr[right-1]):
                        right-=1
                    left+=1
                    right-=1

                elif total_sum<0:
                    left+=1
                else:
                    right-=1
    return result
print(three_sum([-1,-2,1,2,3,-4,-5,-2,-1,-1,2,3]))
```