DSA DAY 4

#tw osum two midium sorted array
"""
```
def two_sum_two(arr , target):
    n=len(arr)
    left=1
    right=n-1
    while(left<right):
        ans=arr[left-1]+arr[right-1]
        if ans==target:
            return [left ,right]
        if ans>target:
            right-=1
        if ans< target:
            left+=1
print(two_sum_two([2,3,4,6,7,7] ,9))
```

#explanation :
in this quetion we have to return the actual sequence of means the actual posittion but as we know the index start from the zero therefore we are using the while loop start from the one and also decrement the left by one so it become easy and at last check ans with the target is less left+=1 else right-=1

#quetion 2

""""

```
def find_plant_water(arr ,capecityAlis , capecityBob):
    left , right=0 , len(arr)-1
    refill=0
    currentalis ,currentBob=capecityAlis, capecityBob
    while(left<=right):
        if left==right:
            if max(currentalis , currentbob)< arr[left]:
                refill+=1
            break

        if currentalis <arr[left]:
            refill+=1
            currentalis=capecityAlis
        currentalis-=arr[left]
        left+=1
        if currentBob <arr[right]:
            refill+=1
            currentbob=capecityBob
        currentBob-=arr[right]
        right-=1
    return refill
print(find_plant_water([2,2,3,3],3,4))

def rearrene_the_sequence(arr):
    res=[0]*len(arr)
    positive=0
    negative=1
    for num in  arr:
        if num<0:
            if(negative<len(arr)):
                res[negative]=num
                negative+=2
        else:
            if(positive<len(arr)):
                res[positive]=num
                positive+=2
    return res

print(rearrene_the_sequence([-1,2,3,-4]))

def pivot_arrenge(nums , pivot):
 lCount, pCount = 0, 0
        for num in nums:
            if num < pivot:
                lCount += 1
            elif num == pivot:
                pCount += 1

        res = [0] * len(nums)
        left, mid, right = 0, lCount, lCount + pCount

        for num in nums:
            if num < pivot:
                res[left] = num
                left += 1
            elif num > pivot:
                res[right] = num
                right += 1
            else:
```

```
            res[mid] = num
            mid += 1

        return res
arrenge_pivot([3,4,5,6,7] ,4)
```