Task 10

Registration ID : SIRSS1038

Leela Satya Kartheek Raja

Q1) Given an array arr[] of positive and negative numbers only.

The task is to find the length of the longest alternating

(means negative-positive-negative or positive-negative-positive) subsequence present in the array.

Examples: Input: arr[] = {-4, 3, -5, 9, 10, 12, 2, -1} Output: 5

Explanation: The longest sequence is {-4, 3, -5, 9, -1}, which is of length 5. There can be many more subsequences of this length.

In [4]:

```
def FLS(x):

    maxLen = 1
    endIndex = 0
    currLen = 1
    for i in range(1, len(x)):
        if x[i] * x[i - 1] < 0:
            currLen = currLen + 1
            if currLen > maxLen:
                maxLen = currLen
                endIndex = i
        else:
            currLen = 1

    sublist = x[endIndex - maxLen + 1: endIndex + 1]
    print("The longest alternating sublist is:- ", sublist)
    print("length of longest alternating sequences:- ",len(sublist))
```

In [5]:

```
if __name__ == '__main__':

    x = [-4, 3, -5, 9, 10, 12, 2, -1]
    FLS(x)
```

```
The longest alternating sublist is:-  [-4, 3, -5, 9]
length of longest alternating sequences:-  4
```

Q2) Given a sorted array of positive integers, rearrange the array alternately i.e first

element should be the maximum value, second minimum value, third-second max, fourth-

second min and so on.

Examples: Input: arr[] = {1, 2, 3, 4, 5, 6, 7} Output: arr[] = {7, 1, 6, 2, 5, 3, 4}

In [6]:

```
def Rrange(arr, n):

    temp = n*[None]
    small, large = 0, n-1

    flag = True

    for i in range(n):
        if flag is True:
            temp[i] = arr[large]
            large -= 1
        else:
            temp[i] = arr[small]
            small += 1

        flag = bool(1-flag)

    for i in range(n):
        arr[i] = temp[i]
    return arr
```

In [9]:

```
arr = [4, 7, 3, 1, 5, 6, 2]
n = len(arr)
print("Original Array:- ",arr)
print("\n")
print("Modified Array:- ",Rrange(arr, n))
print("\n")
```

```
Original Array:-  [4, 7, 3, 1, 5, 6, 2]


Modified Array:-  [2, 4, 6, 7, 5, 3, 1]
```

Q3. Given an array arr of N integers.

Find the contiguous sub-array with maximum sum.

Example 1: Input: N = 5

arr[] = {1,2,3,-2,5} Output: 9

Explanation: Max subarray sum is 9 of elements (1, 2, 3, -2, 5) which is a contiguous subarray.

In [10]:

```
def MSubArray(a,size):

    max_so_far =a[0]
    curr_max = a[0]

    for i in range(1,size):
        curr_max = max(a[i], curr_max + a[i])
        max_so_far = max(max_so_far,curr_max)

    return max_so_far
```

In [11]:

```
a = [1, 2, 3, -2, 5]
print("Maximum contiguous sum is : -" , MSubArray(a,len(a)))
```

```
Maximum contiguous sum is : - 9
```

Q4) Chain Marketing Organization has a scheme for income generation, through which its members generate income for themselves.

The scheme is such that suppose A joins the scheme and makes R and V join this scheme then A is Parent Member of R and V who are child Members. When any member joins the scheme then the parent gets a total commission of 10% from each of its child members.Child members receive a commission of 5% respectively. If a parent member does not have any member joined under him, then he gets a commission of 5%. Take the name of the members joining the scheme as input. Display how many members joined the scheme including parent members. Calculate the Total commission gained by each member in the scheme. The fixed amount for joining the scheme is Rs.5000 on which commission will be generated

SchemeAmount = 5000

In [16]:

```
parent = input()
Yes_No = input()
if Yes_No == "N":
    print("TOTAL MEMBERS:1\nCOMMISSION DETAILS\n{0}: 250 INR".format(parent))
elif Yes_No == "Y":
    child=list(map(str,input().split(",")))
    print("TOTAL MEMBERS:{}".format(len(child)+1))
    print("COMMISSION DETAILS \n{0}:{1} INR".format(parent,len(child)*500))
    for i in child:
        print("{0}:250 INR".format(i))
```

```
Leela
Y
Satya
TOTAL MEMBERS:2
COMMISSION DETAILS
Leela :500 INR
Satya:250 INR
```