

**Answer:**

#..CODE START FROM HERE..

#Python is used as coding language Here.

def printLeaders(arr,size):

for i in range(0, size):

for j in range(i+1, size):

if arr[i]<=arr[j]:

break

if j == size-1:

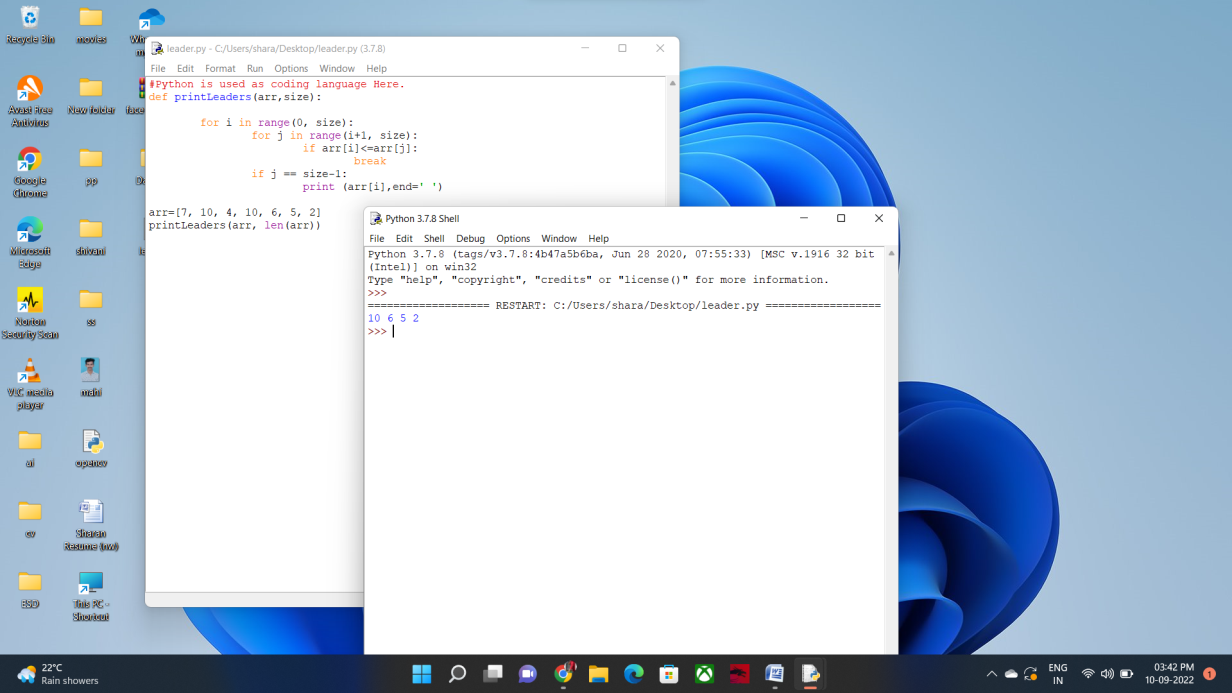
print (arr[i],end=' ')

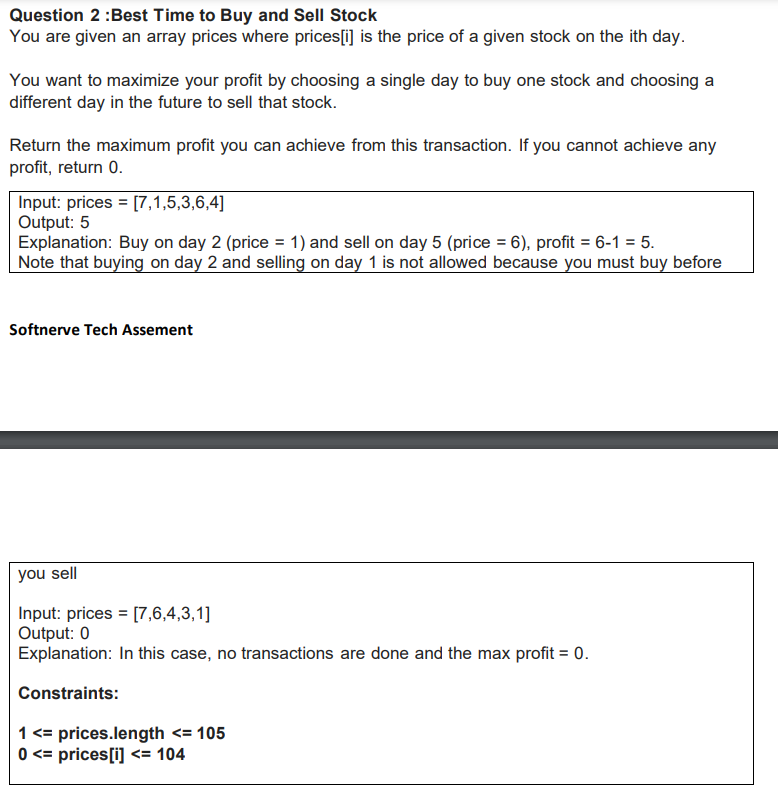
arr=[7, 10, 4, 10, 6, 5, 2]

printLeaders(arr, len(arr))

#..CODE ENDS HERE..

**Output Screenshot:**





**Answer:**

#..CODE START FROM HERE..

#Python is used as coding language Here.

def maxProfit(prices, n):

buy = prices[0]

max\_profit = 0

for i in range(1, n):

## Checking for lower buy value

if (buy > prices[i]):

buy = prices[i]

## Checking for higher profit

elif (prices[i] - buy > max\_profit):

max\_profit = prices[i] - buy;

return max\_profit;

## Driver code

if \_\_name\_\_=='\_\_main\_\_':

prices = [ 7, 1, 5, 6, 4 ];

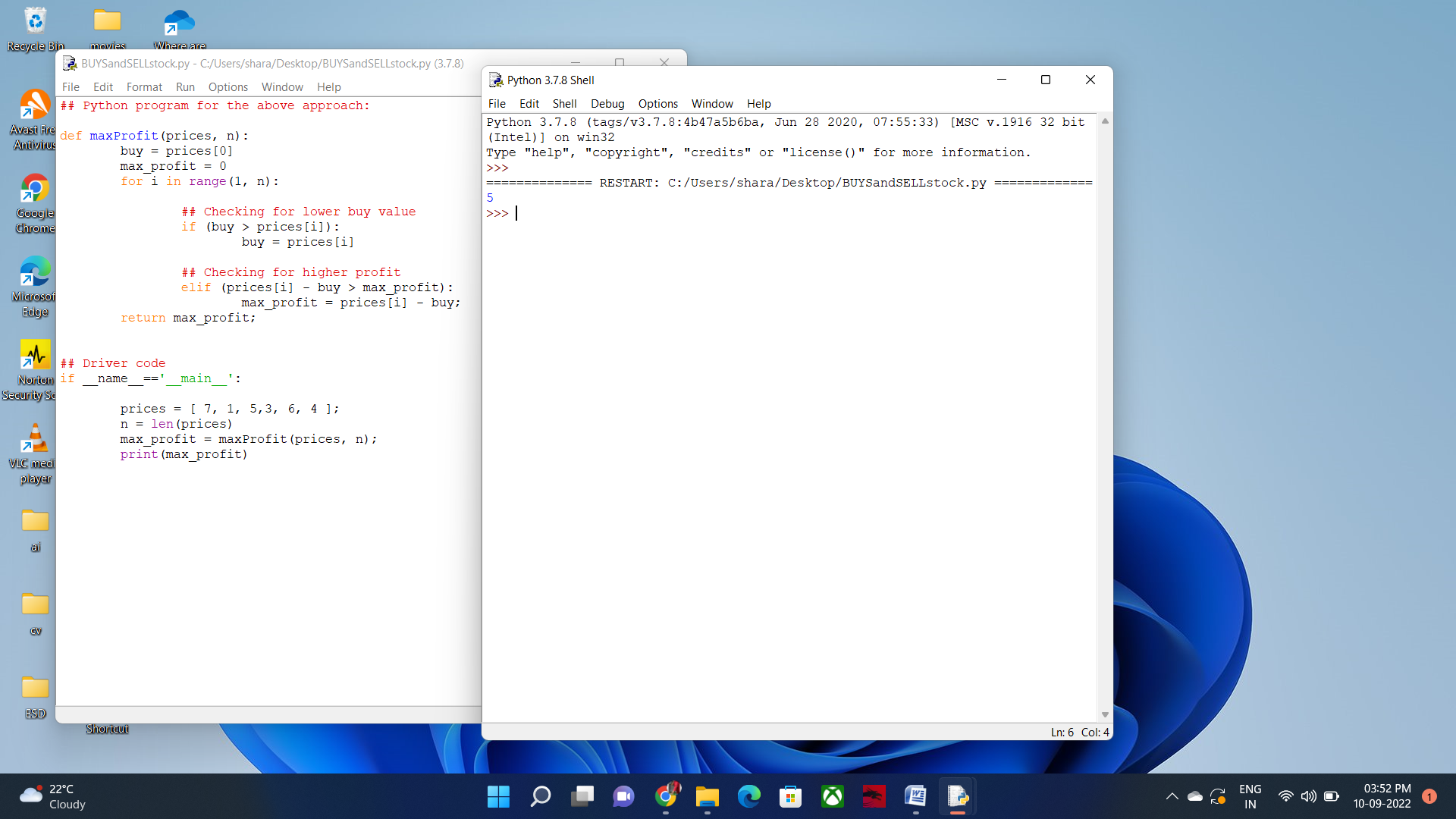
n = len(prices)

max\_profit = maxProfit(prices, n);

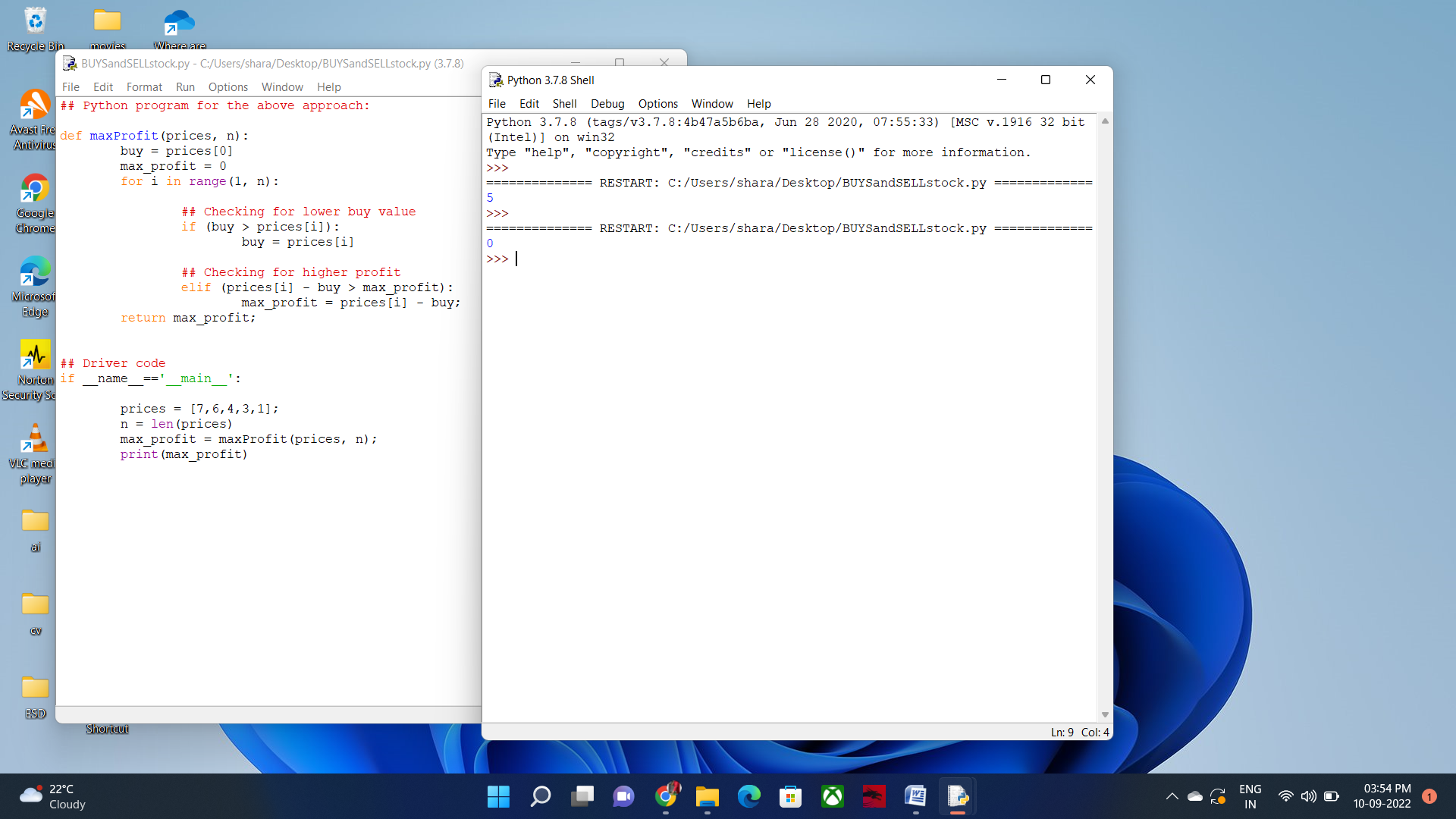
print(max\_profit)

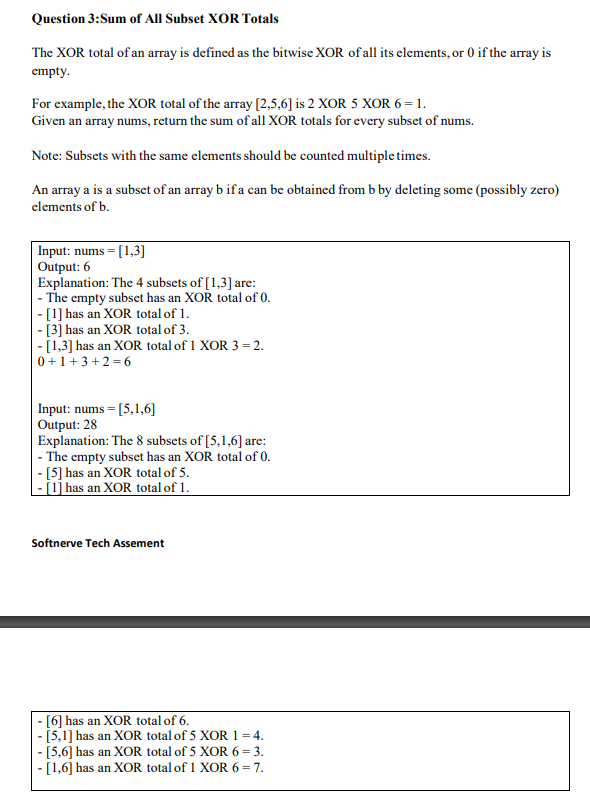
#..CODE ENDS HERE..

**TESTCASE 1:**



**TEST CASE 2:**





**Answer:**

#..CODE START FROM HERE..

#Python is used as coding language Here.

def rec(i, x, arr, size):

# return the current xor sum if we reach the end of

# array

if (i == size):

return x

# first choice can be to include the i-th element in

# the subset and thus we take its xor

choice1 = rec(i + 1, x ^ arr[i], arr, size)

# second choice can be to include the i-th element in

# the subset and thus we take its xor

choice2 = rec(i + 1, x, arr, size)

# return sum of both the choices as we need to find the

# sum of xor of all subsets

return choice1 + choice2

def xorSum(arr, size):

return rec(0, 0, arr, size)

arr = [1,3]

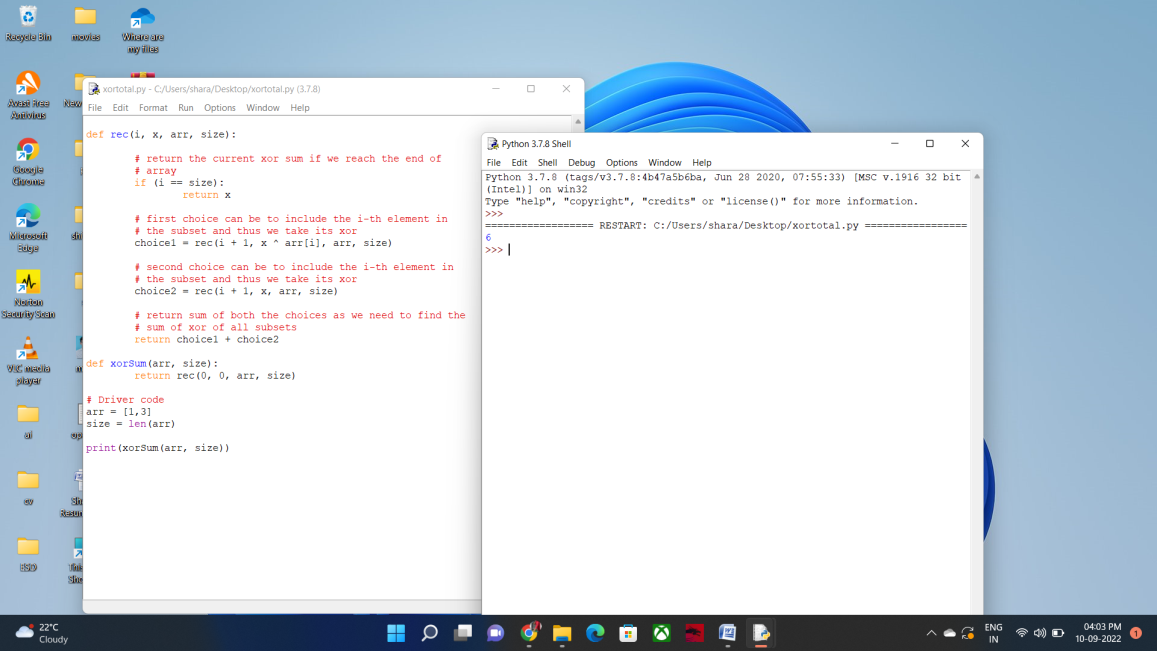
size = len(arr)

print(xorSum(arr, size))

#..CODE ENDS HERE..

**Output Screenshot:**

Testcase 1:



Testcase 2:

