Started on	Tuesday, 29 April 2025, 10:16 AM
State	Finished
Completed on	Tuesday, 29 April 2025, 10:34 AM
Time taken	17 mins 51 secs
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Write a python program to find the maximum contiguous subarray.

For example:

Test	Input	Result
maxSubArraySum(a,n)	8 -2 -3 4 -1 -2 1 5 -3	Maximum contiguous sum is 7

Answer: (penalty regime: 0 %)

Reset answer

```
def maxSubArraySum(a,size):
    ######################  Add your Code here ##############
    max_till_now = a[0]
    max_ending = 0
    for i in range(0, size):
        max_ending = max_ending + a[i]
        if max_ending < 0:
            max_ending = 0
        elif (max_till_now < max_ending):
            max_till_now = max_ending
    return max_till_now
n=int(input())
a =[] #[-2, -3, 4, -1, -2, 1, 5, -3]
for i in range(n):
    a.append(int(input()))
print("Maximum contiguous sum is", maxSubArraySum(a,n))
```

	Test	Input	Expected	Got	
✔	maxSubArraySum(a,n)	8 -2 -3 4 -1 -2 1 5 -3	Maximum contiguous sum is 7	Maximum contiguous sum is 7	✔
✔	maxSubArraySum(a,n)	5 1 -2 -3 4 5	Maximum contiguous sum is 9	Maximum contiguous sum is 9	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

Create a python program to find Minimum number of jumps to reach end of the array using naive method(recursion) using float values

For example:

Test	Input	Result
minJumps(arr, 0, n-1)	6 2.3 7.4 6.3 1.5 8.2 0.1	Minimum number of jumps to reach end is 2

Answer: (penalty regime: 0 %)

Reset answer

```
def minJumps(arr, l, h):
    if (h == l):
        return 0
    if (arr[l] == 0):
        return float('inf')
    min = float('inf')
    for i in range(l + 1, h + 1):
        if (i < l + arr[l] + 1):
            jumps = minJumps(arr, i, h)
            if (jumps != float('inf') and
                    jumps + 1 < min):
                min = jumps + 1
    return min
arr = []
n = int(input())
for i in range(n):
    arr.append(float(input()))
print('Minimum number of jumps to reach','end is', minJumps(arr, 0, n-1))

```

	Test	Input	Expected	Got	
✔	minJumps(arr, 0, n-1)	6 2.3 7.4 6.3 1.5 8.2 0.1	Minimum number of jumps to reach end is 2	Minimum number of jumps to reach end is 2	✔
✔	minJumps(arr, 0, n-1)	10 3.2 3.2 5 6.2 4.9 1.2 5.0 7.3 4.6 6.2	Minimum number of jumps to reach end is 2	Minimum number of jumps to reach end is 2	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Create a python program to find the length of longest common subsequence using naive recursive method

For example:

Input	Result
AGGTAB GXTXAYB	Length of LCS is 4

Answer: (penalty regime: 0 %)

```
def lcs(x,y,m,n):
    if m==0 or n==0:
        return 0
    elif x[m-1]==y[n-1]:
        return 1+lcs(x,y,m-1,n-1)
    else:
        return max(lcs(x,y,m,n-1),lcs(x,y,m-1,n))
X = input()
Y = input()
print ("Length of LCS is ", lcs(X , Y, len(X), len(Y)) )
```

	Input	Expected	Got	
✔	AGGTAB GXTXAYB	Length of LCS is 4	Length of LCS is 4	✔
✔	saveetha engineering	Length of LCS is 2	Length of LCS is 2	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

Correct

Mark 20.00 out of 20.00

Write a Python program using A Naive recursive implementation of Minimum Cost Path Problem.

For example:

Input	Result
3 3	8

Answer: (penalty regime: 0 %)

Reset answer

```
R = int(input())
C = int(input())
import sys
def minCost(cost, m, n):
    ############  Add your Code Here ##################
    if (n < 0 or m < 0):
        return sys.maxsize
    elif (m == 0 and n == 0):
        return cost[m][n]
    else:
        return cost[m][n] + min( minCost(cost, m-1, n-1),
                                 minCost(cost, m-1, n),
                                 minCost(cost, m, n-1) )
def min(x, y, z):
    if (x < y):
        return x if (x < z) else z
    else:
        return y if (y < z) else z
cost= [ [1, 2, 3],
        [4, 8, 2],
        [1, 5, 3] ]
print(minCost(cost, R-1, C-1))
```

	Input	Expected	Got	
✔	3 3	8	8	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

Create a Dynamic Programming python Implementation of Coin Change Problem.

For example:

Test	Input	Result
count(arr, m, n)	3 4 1 2 3	4

Answer: (penalty regime: 0 %)

Reset answer

```
def count(S, m, n):
    table = [[0 for x in range(m)] for x in range(n+1)]
    for i in range(m):
        table[0][i] = 1
    for i in range(1, n+1):
        for j in range(m):
            x = table[i - S[j]][j] if i-S[j] >= 0 else 0
            y = table[i][j-1] if j >= 1 else 0
            table[i][j] = x + y
    return table[n][m-1]
arr = []
m = int(input())
n = int(input())
for i in range(m):
    arr.append(int(input()))
print(count(arr, m, n))
```

	Test	Input	Expected	Got	
✔	count(arr, m, n)	3 4 1 2 3	4	4	✔
✔	count(arr, m, n)	3 16 1 2 5	20	20	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.