Started on	Wednesday, 16 April 2025, 3:05 PM
State	Finished
Completed on	Wednesday, 16 April 2025, 5:44 PM
Time taken	2 hours 39 mins
Overdue	39 mins 28 secs
Grade	**80.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Print All Paths With Minimum Jumps

```
1. You are given a number N representing number of elements.
2. You are given N space separated numbers (ELE : elements).
3. Your task is to find & print
    3.1) "MINIMUM JUMPS" need from 0th step to (n-1)th step.
    3.2) all configurations of "MINIMUM JUMPS".
NOTE: Checkout sample question/solution video inorder to have more insight.
```

For example:

Test	Input	Result
minJumps(arr)	10 3 3 0 2 1 2 4 2 0 0	0 -> 3 -> 5 -> 6 -> 9 0 -> 3 -> 5 -> 7 -> 9

Answer: (penalty regime: 0 %)

Reset answer

```
from queue import Queue
import sys
class Pair(object):
    idx = 0
    psf = ""
    jmps = 0
    def __init__(self, idx, psf, jmps):

        self.idx = idx
        self.psf = psf
        self.jmps = jmps
def minJumps(arr):
    ##################  Add your Code here.
    #Start here
    MAX_VALUE = sys.maxsize
    dp = [MAX_VALUE for i in range(len(arr))]
    n = len(dp)
    dp[n - 1] = 0
    for i in range(n - 2, -1, -1):
        steps = arr[i]
        minimum = MAX_VALUE
        for j in range(1, steps + 1, 1):
```

	Test	Input	Expected	Got	
✔	minJumps(arr)	10 3 3 0 2 1 2 4 2 0 0	0 -> 3 -> 5 -> 6 -> 9 0 -> 3 -> 5 -> 7 -> 9	0 -> 3 -> 5 -> 6 -> 9 0 -> 3 -> 5 -> 7 -> 9	✔

	Test	Input	Expected	Got	
✔	minJumps(arr)	7 5 5 0 3 2 3 6	0 -> 1 -> 6 0 -> 3 -> 6 0 -> 4 -> 6 0 -> 5 -> 6	0 -> 1 -> 6 0 -> 3 -> 6 0 -> 4 -> 6 0 -> 5 -> 6	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Incorrect

Mark 0.00 out of 20.00

SUBSET SUM PROBLEM

COUNT OF SUBSETS WITH SUM EQUAL TO X

Given an array arr[] of length N and an integer X, the task is to find the number of subsets with a sum equal to X.

Examples:

Input: *arr[] = {1, 2, 3, 3}, X = 6*
Output: *3*
All the possible subsets are {1, 2, 3},
{1, 2, 3} and {3, 3}

Input: *arr[] = {1, 1, 1, 1}, X = 1*
Output: *4*

THE INPUT

1.No of numbers

2.Get the numbers

3.Sum Value

For example:

Input	Result
4 2 4 5 9 15	1
6 3 34 4 12 3 2 7	2

Answer: (penalty regime: 0 %)

Reset answer

```
def subsetSum(arr, n, i,sum, count):
#Write your code here




arr=[]
size=int(input())
for j in range(size):
    value=int(input())
    arr.append(value)
sum = int(input())
n = len(arr)

print(subsetSum(arr, n, 0, sum, 0))
```

Syntax Error(s)

```
Sorry: IndentationError: expected an indented block (__tester__.python3, line 9)
```

Incorrect

Marks for this submission: 0.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Write a python program to find the maximum contiguous subarray on the given float array using kadane's algorithm.

For example:

Test	Input	Result
s.maxSubArray(A)	5 -9.6 -3.5 6.3 8.31 9.2	The sum of contiguous sublist with the largest sum is 23.8

Answer: (penalty regime: 0 %)

Reset answer

```
class Solution:
    ######################  Add your Code here ##############
    def maxSubArray(self,A):
        res=0
        mm= -10000
        for v in A:
            res+=v
            mm=max(mm,res)
            if res<0:
                res=0
        return mm
A =[]
n=int(input())
for i in range(n):
    A.append(float(input()))
s=Solution()
print("The sum of contiguous sublist with the largest sum is {:.1f}".format(s.maxSubArray(A)))
```

	Test	Input	Expected	Got	
✔	s.maxSubArray(A)	5 -9.6 -3.5 6.3 8.31 9.2	The sum of contiguous sublist with the largest sum is 23.8	The sum of contiguous sublist with the largest sum is 23.8	✔
✔	s.maxSubArray(A)	7 2.3 6.5 4.6 -7.8 -2.8 -1.6 9.8	The sum of contiguous sublist with the largest sum is 13.4	The sum of contiguous sublist with the largest sum is 13.4	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

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):
            # Count of solutions including S[j]
            #Start here
            x = table[i - S[j]][j] if i-S[j] >= 0 else 0
            # Count of solutions excluding S[j]
            y = table[i][j-1] if j >= 1 else 0
            # total count
            table[i][j] = x + y
    return table[n][m-1]
    #End here
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.

Question **5**

Correct

Mark 20.00 out of 20.00

Write a python program to Implement Minimum cost path using Dynamic Programming.

For example:

Input	Result
3 3	8

Answer: (penalty regime: 0 %)

```
R = int(input())
C = int(input())
def minCost(cost, m, n):
    tc = [[0 for x in range(C)] for x in range(R)]
    tc[0][0] = cost[0][0]
    for i in range(1, m+1):
        tc[i][0] = tc[i-1][0] + cost[i][0]
    for j in range(1, n+1):
        tc[0][j] = tc[0][j-1] + cost[0][j]
    for i in range(1, m+1):
        for j in range(1, n+1):
            tc[i][j] = min(tc[i-1][j-1], tc[i-1][j], tc[i][j-1]) + cost[i][j]

    return tc[m][n]

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.