Started on	Saturday, 10 May 2025, 1:22 PM
State	Finished
Completed on	Saturday, 10 May 2025, 1:41 PM
Time taken	18 mins 36 secs
Grade	**80.00** out of 100.00

Question **1**

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.

Question **2**

Incorrect

Mark 0.00 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.



def possiblePath(arr, dp):

    queue = Queue(maxsize = 0)
    p1 = Pair(0, "0", dp[0])
    queue.put(p1)
```

Syntax Error(s)

```
Sorry: IndentationError: expected an indented block (__tester__.python3, line 18)
```

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.

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):
    max=a[0]
    sum=0
    for i in range(0,n):
        sum=sum+a[i]
        if(sum<0):
            sum=0
        elif(sum>max):
            max=sum
    return max

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	✔

	Test	Input	Expected	Got	
✔	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 **4**

Correct

Mark 20.00 out of 20.00

LONGEST COMMON SUBSTRING PROBLEM

Given two strings 'X' and 'Y', find the length of the longest common substring.

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()
m=len(x)
n=len(y)
print("Length of Longest Common Substring is",lcs(x,y,m,n))
```

	Input	Expected	Got	
✔	ABC BABA	Length of Longest Common Substring is 2	Length of Longest Common Substring is 2	✔
✔	abcdxyz xyzabcd	Length of Longest Common Substring is 4	Length of Longest Common Substring is 4	✔

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 for printing Minimum Cost Simple Path between two given nodes in a directed and weighted graph

For example:

Test	Result
`minimumCostSimplePath(s, t, visited, graph)`	`-3`

Answer: (penalty regime: 0 %)

Reset answer

```
import sys
V = 5
INF = sys.maxsize
def minimumCostSimplePath(u, destination,
                          visited, graph):
    if(u==destination):
        return 0
    visited[u]=1
    ans=INF
    for i in range(V):
        if(graph[u][i]!=INF and not visited[i]):
            curr=minimumCostSimplePath(i,destination,visited,graph)
            if(curr<INF):
                ans=min(ans,graph[u][i]+curr)
    visited[u]=0
    return ans

if __name__=="__main__":
    graph = [[INF for j in range(V)]
                  for i in range(V)]
    visited = [0 for i in range(V)]
    graph[0][1] = -1
```

	Test	Expected	Got	
✔	`minimumCostSimplePath(s, t, visited, graph)`	`-3`	`-3`	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.