Started on	Tuesday, 13 May 2025, 3:39 PM
State	Finished
Completed on	Tuesday, 13 May 2025, 4:15 PM
Time taken	35 mins 49 secs
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Flag question

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): ## travel from 1 to end
        for j in range(m):
            x=table[i-S[j]][j] if S[j]<=i else 0##include sj if < i or else 0 if < - with i & add to x

            y=table[i][j-1] if j>=1 else 0  ##solutions excluding S[j]

            # total count
            table[i][j]=x+y  ##add both

    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**

Correct

Mark 20.00 out of 20.00

Flag question

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:
    def maxSubArray(a,size):
        result =0
        mm=-1000
        for i in A: ## travel through array
            result=result+i ## adding i value to res
            mm=max(mm,result) ## finding the max
```

```
            mm=max(mm,result) ## finding the max
            if result<0:  ## if negative 0
                result=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 **3**

Correct

Mark 20.00 out of 20.00

Flag question

Write a python program to implement quick sort on the given values and print the sorted list and pivot value of each iteration.

For example:

Input	Result
5 41 21 6 34 8	Input List [41, 21, 6, 34, 8] pivot: 41 pivot: 8 pivot: 21 Sorted List [6, 8, 21, 34, 41]
4 5 2 49 3	Input List [5, 2, 49, 3] pivot: 5 pivot: 3 Sorted List [2, 3, 5, 49]

Answer: (penalty regime: 0 %)

```
def partition(list1,start,end):
    i=start+1
    j=end-1
    pivot=list1[start]
    print("pivot: ",pivot)
    while True:
        while (i<=j and list1[i]<=pivot):
            i+=1
        while (i<=j and list1[j]>pivot):
            j-=1
        if i<=j:
            list1[i],list1[j]=list1[j],list1[i]
        else:
            list1[start],list1[j]=list1[j],list1[start]
            return j
def quick_sort(list1,start,end):
    if (end-start)>1:
        pi=partition(list1,start,end)
        quick_sort(list1,start,pi)
        quick_sort(list1,pi+1,end)
array = []
n=int(input())
```

	Input	Expected	Got	
	5 41 21 6 34 8	Input List [41, 21, 6, 34, 8] pivot: 41 pivot: 8 pivot: 21 Sorted List [6, 8, 21, 34, 41]	Input List [41, 21, 6, 34, 8] pivot: 41 pivot: 8 pivot: 21 Sorted List [6, 8, 21, 34, 41]	
	4 5 2 49 3	Input List [5, 2, 49, 3] pivot: 5 pivot: 3 Sorted List [2, 3, 5, 49]	Input List [5, 2, 49, 3] pivot: 5 pivot: 3 Sorted List [2, 3, 5, 49]	
	6 41 5 69 83 27 10	Input List [41, 5, 69, 83, 27, 10] pivot: 41 pivot: 27 pivot: 10 pivot: 83 Sorted List [5, 10, 27, 41, 69, 83]	Input List [41, 5, 69, 83, 27, 10] pivot: 41 pivot: 27 pivot: 10 pivot: 83 Sorted List [5, 10, 27, 41, 69, 83]	

Passed all tests!

Correct

Marks for this submission: 20.00/20.00.

Question **4**

Correct

Mark 20.00 out of 20.00

Flag question

Write a Python program to Implement Minimum cost path in a Directed Graph

For example:

Test	Result
getMinPathSum(graph, visited, necessary, source, dest, 0);	12

Answer: (penalty regime: 0 %)

Reset answer

```
minsum = 1000000000
def getMinPathSum(graph, visited, necessary,
                  src, dest, currSum):
    global minsum
    if(src==dest): ## if already reached
        flag=True;
        for i in necessary:  ## checking condition 2,4 visit or not
            if(not visited[i]):
                flag = False;
                break;
        if(flag):
            minsum=min(minsum,currSum);   ## update minsum
            return;
    else:
        visited[src]=True;
        for node in graph[src]: ## visiting neighbour nodes
            if not visited[node[0]]:
                visited[node[0]]=True;
                getMinPathSum(graph,visited,necessary,node[0],dest,currSum+node[1]);
                visited[node[0]]=False;
        visited[src]=False;

```

	Test	Expected	Got	
	getMinPathSum(graph, visited, necessary, source, dest, 0);	12	12	

Passed all tests!

Correct

Marks for this submission: 20.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

Flag question

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
## Start
def minJumps(arr):

    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]
```

	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	
	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.