Started on	Thursday, 1 May 2025, 11:15 AM
State	Finished
Completed on	Thursday, 1 May 2025, 11:51 AM
Time taken	35 mins 40 secs
Grade	**80.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Create a python program to find the minimum number of jumps needed to reach end of the array using Dynamic Programming.

For example:

Test	Input	Result
minJumps(arr,n)	6 1 3 6 1 0 9	Minimum number of jumps to reach end is 3

Answer: (penalty regime: 0 %)

Reset answer

```
def minJumps(arr, n):
    jumps = [0 for i in range(n)]

    if (n == 0) or (arr[0] == 0):
        return float('inf')

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

	Test	Input	Expected	Got	
✔	minJumps(arr,n)	6 1 3 6 1 0 9	Minimum number of jumps to reach end is 3	Minimum number of jumps to reach end is 3	✔
✔	minJumps(arr,n)	7 2 3 -8 9 5 6 4	Minimum number of jumps to reach end is 3	Minimum number of jumps to reach end is 3	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **2**

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 **3**

Correct

Mark 20.00 out of 20.00

Create a python function to compute the fewest number of coins that we need to make up the amount given.

For example:

Test	Input	Result
ob1.coinChange(s,amt)	3 11 1 2 5	3

Answer: (penalty regime: 0 %)

Reset answer

```
class Solution(object):
   def coinChange(self, coins, amount):
      ####################      Add your Code Here ###########
       dp = [float('inf')] * (amount + 1)
       dp[0]=0
       for coin in coins:
           for i in range(coin, amount + 1):
               dp[i] = min(dp[i], dp[i - coin] + 1)
       return dp[amount] if dp[amount]!=float('inf') else -1

ob1 = Solution()
n=int(input())
s=[]
amt=int(input())
for i in range(n):
    s.append(int(input()))


print(ob1.coinChange(s,amt))
```

	Test	Input	Expected	Got	
✔	ob1.coinChange(s,amt)	3 11 1 2 5	3	3	✔
✔	ob1.coinChange(s,amt)	3 12 1 2 5	3	3	✔
✔	ob1.coinChange(s,amt)	3 22 1 2 5	5	5	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

Incorrect

Mark 0.00 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:
    def maxSubArray(a,size):
        ####################  Add your Code here #############
        max_sum = A[8]
        current_sum = A[5]
        for i in range(1, len(A)):
            current_sum = max(A[i], current_sum + A[i])
             max_sum = max(max_sum, current_sum)
        return max_sum

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)))
```

Syntax Error(s)

```
Sorry: IndentationError: unexpected indent (__tester__.python3, line 8)
```

Incorrect

Marks for this submission: 0.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

Write a python program to implement merge sort using iterative approach on the given list of float values.

For example:

Test	Input	Result
Merge_Sort(S)	5 10.2 21.3 3.5 7.8 9.8	The Original array is: [10.2, 21.3, 3.5, 7.8, 9.8] Array after sorting is: [3.5, 7.8, 9.8, 10.2, 21.3]
Merge_Sort(S)	6 20.3 41.2 5.3 6.2 8.1 65.2	The Original array is: [20.3, 41.2, 5.3, 6.2, 8.1, 65.2] Array after sorting is: [5.3, 6.2, 8.1, 20.3, 41.2, 65.2]

Answer: (penalty regime: 0 %)

```
def Merge_Sort(S):
    if(len(S)>1):
        mid = len(S)//2
        left = S[:mid]
        right = S[mid:]
        Merge_Sort(left)
        Merge_Sort(right)
        i = j = k = 0
        while(i < len(left) and j < len(right)):
            if(left[i] < right[j]):
                S[k] = left[i]
                i = i +1
            else:
                S[k] = right[j]
                j = j+1
            k = k+1
        while(i<len(left)):
            S[k] = left[i]
            i = i+1
            k = k+1
        while(j<len(right)):
            S[k] = right[j]
```

	Test	Input	Expected	Got	
✔	Merge_Sort(S)	5 10.2 21.3 3.5 7.8 9.8	The Original array is: [10.2, 21.3, 3.5, 7.8, 9.8] Array after sorting is: [3.5, 7.8, 9.8, 10.2, 21.3]	The Original array is: [10.2, 21.3, 3.5, 7.8, 9.8] Array after sorting is: [3.5, 7.8, 9.8, 10.2, 21.3]	✔
✔	Merge_Sort(S)	6 20.3 41.2 5.3 6.2 8.1 65.2	The Original array is: [20.3, 41.2, 5.3, 6.2, 8.1, 65.2] Array after sorting is: [5.3, 6.2, 8.1, 20.3, 41.2, 65.2]	The Original array is: [20.3, 41.2, 5.3, 6.2, 8.1, 65.2] Array after sorting is: [5.3, 6.2, 8.1, 20.3, 41.2, 65.2]	✔

	Test	Input	Expected	Got	
✔	Merge_Sort(S)	4 2.3 6.1 4.5 96.5	The Original array is: [2.3, 6.1, 4.5, 96.5] Array after sorting is: [2.3, 4.5, 6.1, 96.5]	The Original array is: [2.3, 6.1, 4.5, 96.5] Array after sorting is: [2.3, 4.5, 6.1, 96.5]	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.