TT DS PYTHON MODULE-24

Started on Tuesday, 20 May 2025, 3:28 PM

State Finished

Completed on Tuesday, 20 May 2025, 3:51 PM

Time taken 23 mins 3 secs

Grade 80.00 out of 100.00

Question 1
Correct
Mark 20.00 out of 20.00

Flag question

Create a python program using dynamic programming for 0/1 knapsack problem.

For example:

Test	Input	Result	
knapSack(W, wt, val, n)	3 50 60 100 120 10 20 30	The maximum value that can be put in a knapsack of capacity W is:	220

Answer: (penalty regime: 0 %)

Reset answer

```
def knapSack(W, wt, val, n):
    dp=[[-1]*(W+1) for _ in range(len(val)+1)]
    for i in range(W+1):
 2
 3
 4
             dp[0][i]=0
 5
         for i in range(n+1):
 6
             dp[i][0]=0
         for i in range(1,n+1):
              for j in range(1,W+1):
 8
                  if j<wt[i-1]:</pre>
 9
10
                       dp[i][j]=dp[i-1][j]
11
12
                       dp[i][j]=max(dp[i-1][j],dp[i-1][j-wt[i-1]]+val[i-1])
         return dp[n][W]
13
14
15
     x=int(input())
    y=int(input())
16
17
    W=int(input())
18
    val=[]
19
    wt=[]
20
     for i in range(x):
21
         val.append(int(input()))
22 | for y in range(y):
```

Test	Input	Expected	Got
knapSack(W, wt, val, n)	3 3 50 60 100 120 10 20 30	The maximum value that can be put in a knapsack of capacity W is: 220	The maximum
knapSack(W, wt, val, n)	3 40 50 90 110 10 20 30	The maximum value that can be put in a knapsack of capacity W is: 160	The maximum

Passed all tests!

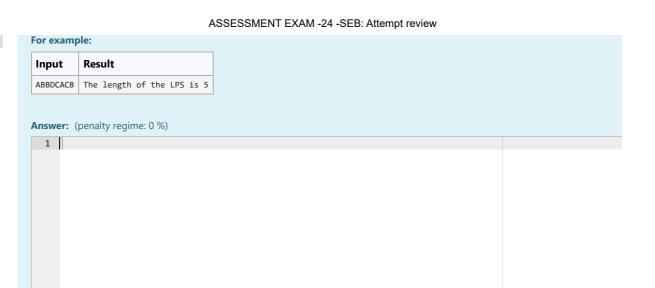
4 @

Marks for this submission: 20.00/20.00.

Question **2**Not answered
Mark 0.00 out of 20.00

LONGEST PALINDROMIC SUBSEQUENCE

Given a sequence, find the length of the longest palindromic subsequence in it.



Question **3**Correct
Mark 20.00 out of 20.00

Flag question

Create a python program to for the following problem statement.

You are given an n x n grid representing a field of cherries, each cell is one of three possible integers.

- @ means the cell is empty, so you can pass through,
- 1 means the cell contains a cherry that you can pick up and pass through, or
- -1 means the cell contains a thorn that blocks your way.

Return the maximum number of cherries you can collect by following the rules below:

- Starting at the position (0, 0) and reaching (n 1, n 1) by moving right or down through valid path cells (cells with value 0 or 1
- After reaching (n 1, n 1), returning to (0, 0) by moving left or up through valid path cells.
- When passing through a path cell containing a cherry, you pick it up, and the cell becomes an empty cell 0.
- If there is no valid path between (0, 0) and (n 1, n 1), then no cherries can be collected.

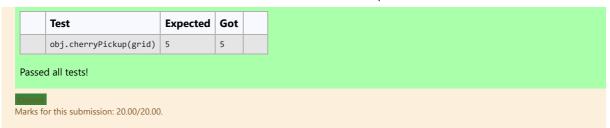
For example:

Test	Result
obj.cherryPickup(grid)	5

Answer: (penalty regime: 0 %)

Reset answer

```
class Solution:
 1
 2
         def cherryPickup(self, grid):
 3
             n = len(grid)
             dp = [[[-1] * n for _ in range(n)] for _ in range(n)]
 4
 5
              def f(x1, y1, x2):
 6
 7
                  y2 = x1 + y1 - x2
 8
                  if x1 < 0 or y1 < 0 or x2 < 0 or y2 < 0 or grid[x1][y1] == -1 or grid[x2][y2] == -1:
                      return float('-inf')
10
                  if x1 == 0 and y1 == 0 and x2 == 0 and y2 == 0:
                      return grid[0][0]
11
12
                  if dp[x1][y1][x2] != -1:
13
                      return dp[x1][y1][x2]
                  cherries = grid[x1][y1]
14
                  if x1 != x2 or y1 != y2:
cherries += grid[x2][y2]
15
16
17
                  cherries += max(
                       f(x1 - 1, y1, x2 - 1),
18
                      f(x1, y1 - 1, x2 - 1),
f(x1 - 1, y1, x2),
f(x1, y1 - 1, x2))
19
20
21
22
```



Question **4**Correct
Mark 20.00 out of 20.00

Flag question

Given a 2D matrix tsp[][], where each row has the array of distances from that indexed city to all the other cities and -1 denotes that there doesn't exist a path between those two indexed cities. The task is to print minimum cost in TSP cycle.

```
tsp[[] = {{-1, 30, 25, 10}, {15, -1, 20, 40}, {10, 20, -1, 25}, {30, 10, 20, -1}};
```

Answer: (penalty regime: 0 %)

```
Reset answer
```

```
from typing import DefaultDict
 2
 3
    INT_MAX = 2147483647
4
 5
 6
    def findMinRoute(tsp):
 7
8
        sum = 0
9
        counter = 0
10
        j = 0
11
        i = 0
        min = INT MAX
12
13
        visitedRouteList = DefaultDict(int)
14
15
16
        visitedRouteList[0] = 1
        route = [0] * len(tsp)
17
18
19
20
        while i < len(tsp) and j < len(tsp[i]):</pre>
            if counter >= len(tsp[i]) - 1:
21
22
                break
```

```
Expected

Minimum Cost is: 50 Minimum Cost is: 50

Passed all tests!

Marks for this submission: 20.00/20.00.
```

Question **5**Correct
Mark 20.00 out of 20.00

Flag question

Create a python program to find the maximum value in linear search.

For example:

Test	Input	Result
<pre>find_maximum(test_scores)</pre>	10 88 93 75 100 80 67 71 92 90 83	Maximum value is 100

Answer: (penalty regime: 0 %)

Reset answer

1 def find_maximum(lst):
2 if len(lst)--0:

```
notice
           return 0
         max_=lst[0]
for i in lst:
 4
 5
          if i>max_:
    max_=i
 6
 7
         return max_
 8
 9
10
    test_scores = []
11  n=int(input())
    for i in range(n):
12
    test_scores.append(int(input()))
print("Maximum value is ",find_maximum(test_scores))
13
14
15
16
```

Test	Input	Expected	Got
<pre>find_maximum(test_scores)</pre>	10 88 93 75 100 80 67 71 92 90 83	Maximum value is 100	Maximum value is 100
<pre>find_maximum(test_scores)</pre>	5 45 86 95 76 28	Maximum value is 95	Maximum value is 95

Passed all tests!

Marks for this submission: 20.00/20.00.

Finish ı