Started on	Wednesday, 21 May 2025, 1:23 PM
State	Finished
Completed on	Wednesday, 21 May 2025, 1:38 PM
Time taken	14 mins 38 secs
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 5.00 out of 5.00

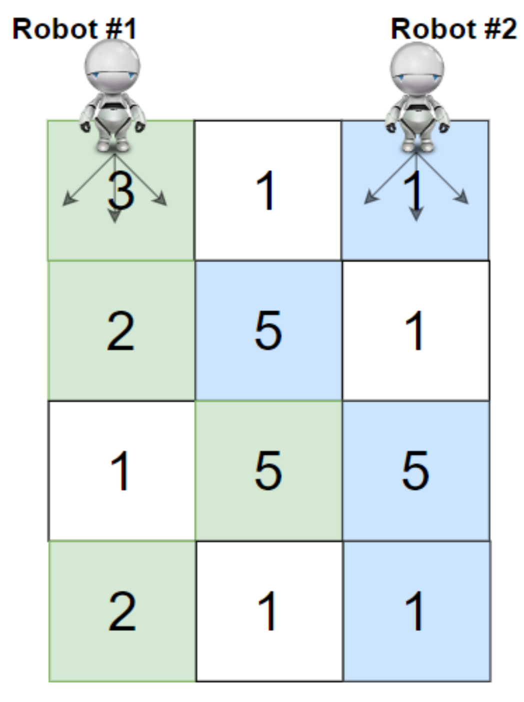
You are given a `rows x cols` matrix `grid` representing a field of cherries where `grid[i][j]` represents the number of cherries that you can collect from the `(i, j)` cell.

You have two robots that can collect cherries for you:

- **Robot #1** is located at the **top-left corner** `(0, 0)`, and
- **Robot #2** is located at the **top-right corner** `(0, cols - 1)`.

Return *the maximum number of cherries collection using both robots by following the rules below*:

- From a cell `(i, j)`, robots can move to cell `(i + 1, j - 1)`, `(i + 1, j)`, or `(i + 1, j + 1)`.
- When any robot passes through a cell, It picks up all cherries, and the cell becomes an empty cell.
- When both robots stay in the same cell, only one takes the cherries.
- Both robots cannot move outside of the grid at any moment.
- Both robots should reach the bottom row in `grid`.



3	1	1
2	5	1
1	5	5
2	1	1

For example:

Test	Result
ob.cherryPickup(grid)	24

Answer: (penalty regime: 0 %)

Reset answer

```
class Solution(object):
    def cherryPickup(self, grid):
        ROW_NUM = len(grid)
        COL_NUM = len(grid[0])
        dp = [[[float('-inf')] * COL_NUM for _ in range(COL_NUM)] for _ in range(R
        dp[0][0][COL_NUM - 1] = grid[0][0] + grid[0][COL_NUM - 1]
        for i in range(1, ROW_NUM):
            for j1 in range(COL_NUM):
                for j2 in range(COL_NUM):
                    curr_cherries = grid[i][j1]
                    if j1 != j2:
                        curr_cherries+=grid[i][j2]
                    for prev_j1 in range(j1 - 1, j1 + 2):
                        for prev_j2 in range(j2 - 1, j2 + 2):
                            if 0 <= prev_j1 < COL_NUM and 0 <= prev_j2 < COL_NUM:
                                prev_cherries = dp[i - 1][prev_j1][prev_j2]
                                dp[i][j1][j2] = max(dp[i][j1][j2], curr_cherries +

        return max(0, dp[ROW_NUM - 1][0][COL_NUM - 1])
```

```

grid=[[3,1,1],
```

	Test	Expected	Got	
✓	ob.cherryPickup(grid)	24	24	✓

Passed all tests! ✓

Correct

Marks for this submission: 5.00/5.00.

Question **2**

Correct

Mark 5.00 out of 5.00

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.

- `0` 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:
    def cherryPickup(self, grid):
        n = len(grid)
        ### add code here
        dp=[[[-1]*n for _ in range(n)] for _ in range(n)]
        def f(x1,y1,x2):
            y2=x1+y1-x2
            if x1<0 or y1<0 or x2<0 or y2<0 or grid[x1][y1]==-1 or grid[x2][y2]==-
                return float('-inf')
            if x1==0 and y1==0 and x2==0 and y2==0:
                return grid[0][0]
            if dp[x1][y1][x2]!=-1:
                return dp[x1][y1][x2]
            cherries=grid[x1][y1]
            if x1!=x2 or y1!=y2:
                cherries+=grid[x2][y2]
            cherries+=max(
                          f(x1-1,y1,x2-1),
                           f(x1,y1-1,x2-1),
                           f(x1-1,y1,x2),
                           f(x1,y1-1,x2))
```

	Test	Expected	Got	
✓	obj.cherryPickup(grid)	5	5	✓

Passed all tests! ✓

Correct

Marks for this submission: 5.00/5.00.