

[Dashboard](#) / [My courses](#) / [CS23331-DAA-2023-CSE](#) / [Dynamic Programming](#) / [2-DP-Playing with chessboard](#)

<b>Started on</b>	Wednesday, 20 November 2024, 1:50 PM
<b>State</b>	Finished
<b>Completed on</b>	Wednesday, 20 November 2024, 2:07 PM
<b>Time taken</b>	16 mins 38 secs
<b>Grade</b>	<b>10.00</b> out of 10.00 ( <b>100%</b> )

## Question 1

Correct

Mark 10.00 out of 10.00

**Playing with Chessboard:**

Ram is given with an  $n \times n$  chessboard with each cell with a monetary value. Ram stands at the (0,0), that is the position of the top left white rook. He is given a task to reach the bottom right black rook position ( $n-1, n-1$ ) constrained that he needs to reach the position by traveling the maximum monetary path under the condition that he can only travel one step right or one step down the board. Help Ram to achieve it by providing an efficient DP algorithm.

**Example:****Input**

```
3
1 2 4
2 3 4
8 7 1
```

**Output:**

```
19
```

**Explanation:**

Totally there will be 6 paths among that the optimal is  
Optimal path value:  $1+2+8+7+1=19$

**Input Format**

First Line contains the integer  $n$

The next  $n$  lines contain the  $n \times n$  chessboard values

**Output Format**

Print Maximum monetary value of the path

**Answer:** (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6     int chessboard[n][n];
7     for (int i = 0; i < n; i++) {
8         for (int j = 0; j < n; j++) {
9             scanf("%d", &chessboard[i][j]);
10        }
11    }
12    long long dp[n][n];
13    dp[0][0] = chessboard[0][0];
14    for (int j = 1; j < n; j++) {
15        dp[0][j] = dp[0][j - 1] + chessboard[0][j];
16    }
17    for (int i = 1; i < n; i++) {
18        dp[i][0] = dp[i - 1][0] + chessboard[i][0];
19    }
20    for (int i = 1; i < n; i++) {
21        for (int j = 1; j < n; j++) {
22            dp[i][j] = chessboard[i][j] + (dp[i - 1][j] > dp[i][j - 1] ? dp[i - 1][j] : dp[i][j - 1]);
23        }
24    }
25    printf("%lld\n", dp[n - 1][n - 1]);
26    return 0;
27 }
```

	Input	Expected	Got	
✓	3 1 2 4 2 3 4 8 7 1	19	19	✓
✓	3 1 3 1 1 5 1 4 2 1	12	12	✓
✓	4 1 1 3 4 1 5 7 8 2 3 4 6 1 6 9 0	28	28	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

[◀ 1-DP-Playing with Numbers](#)

Jump to...

[3-DP-Longest Common Subsequence ▶](#)