



|              |                                    |
|--------------|------------------------------------|
| Started on   | Wednesday, 8 October 2025, 3:08 PM |
| State        | Finished                           |
| Completed on | Wednesday, 8 October 2025, 3:12 PM |
| Time taken   | 3 mins 39 secs                     |
| Grade        | 10.00 out of 10.00 (100%)          |

**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 the position of the top left white rook. He is been 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 max(int a, int b) {
4      return (a > b) ? a : b;
5  }
6
7  int main() {
8      int n;
9      scanf("%d", &n);
10
11     int board[n][n];
12     long long dp[n][n];
13
14     for (int i = 0; i < n; i++)
15         for (int j = 0; j < n; j++)
16             scanf("%d", &board[i][j]);
17
18     for (int i = 0; i < n; i++) {
19         for (int j = 0; j < n; j++) {
20             if (i == 0 && j == 0)
21                 dp[i][j] = board[i][j];
22             else if (i == 0)
23                 dp[i][j] = dp[i][j - 1] + board[i][j];
24             else if (j == 0)
25                 dp[i][j] = dp[i - 1][j] + board[i][j];
26             else
27                 dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]) + board[i][j];
28         }
29     }
30
31     printf("%lld\n", dp[n - 1][n - 1]);
32     return 0;
33 }
34
```

|   | Input                                         | Expected | Got |   |
|---|-----------------------------------------------|----------|-----|---|
| ✓ | 3<br>1 2 4<br>2 3 4<br>8 7 1                  | 19       | 19  | ✓ |
| ✓ | 3<br>1 3 1<br>1 5 1<br>4 2 1                  | 12       | 12  | ✓ |
| ✓ | 4<br>1 1 3 4<br>1 5 7 8<br>2 3 4 6<br>1 6 9 0 | 28       | 28  | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

[Back to Course](#)