

<b>Started on</b>	Thursday, 10 October 2024, 8:29 AM
<b>State</b>	Finished
<b>Completed on</b>	Thursday, 10 October 2024, 8:29 AM
<b>Time taken</b>	14 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 Numbers:**

Ram and Sita are playing with numbers by giving puzzles to each other. Now it was Ram term, so he gave Sita a positive integer 'n' and two numbers 1 and 3. He asked her to find the possible ways by which the number n can be represented using 1 and 3. Write any efficient algorithm to find the possible ways.

**Example 1:****Input:** 6**Output:** 6**Explanation:** There are 6 ways to 6 represent number with 1 and 3

1+1+1+1+1+1

3+3

1+1+1+3

1+1+3+1

1+3+1+1

3+1+1+1

**Input Format**

First Line contains the number n

**Output Format****Print: The number of possible ways 'n' can be represented using 1 and 3**

Sample Input

6

Sample Output

6

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

```

1  #include<stdio.h>
2  int main(){
3      int n;
4      scanf("%d",&n);
5      long long d[n+1];
6      for(int i=0;i<=n;i++){
7          d[i]=0;
8      }
9      d[0]=1;
10     for(int i=1;i<=n;i++){
11         if(i>=1){
12             d[i]=d[i]+d[i-1];
13         }
14         if(i>=3){
15             d[i]=d[i]+d[i-3];
16         }
17     }
18     }
19     printf("%lld",d[n]);
20 }
21

```

	Input	Expected	Got	
✓	6	6	6	✓
✓	25	8641	8641	✓

	Input	Expected	Got	
✓	100	24382819596721629	24382819596721629	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

◀ 5-G-Product of Array elements-Minimum

Jump to...

2-DP-Playing with chessboard ▶

<b>Started on</b>	Thursday, 10 October 2024, 8:30 AM
<b>State</b>	Finished
<b>Completed on</b>	Thursday, 10 October 2024, 8:30 AM
<b>Time taken</b>	33 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 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 maxMonetaryPath(int n, int board[n][n]) {
8     int dp[n][n];
9     dp[0][0] = board[0][0];
10
11     for (int j = 1; j < n; j++) {
12         dp[0][j] = dp[0][j-1] + board[0][j];
13     }
14
15     for (int i = 1; i < n; i++) {
16         dp[i][0] = dp[i-1][0] + board[i][0];
17     }
18
19     for (int i = 1; i < n; i++) {
20         for (int j = 1; j < n; j++) {
21             dp[i][j] = board[i][j] + max(dp[i-1][j], dp[i][j-1]);
22         }
23     }
24
25     return dp[n-1][n-1];
26 }
27
28 int main() {
29     int n;
30
31     scanf("%d", &n);
32
33     int board[n][n];
34
35     for (int i = 0; i < n; i++) {
36         for (int j = 0; j < n; j++) {
```

```

37         scanf("%d", &board[i][j]);
38     }
39 }
40
41 int result = maxMonetaryPath(n, board);
42 printf("%d\n", result);
43
44 return 0;
45 }
46

```

	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 ▶](#)

<b>Started on</b>	Thursday, 10 October 2024, 8:30 AM
<b>State</b>	Finished
<b>Completed on</b>	Thursday, 10 October 2024, 8:31 AM
<b>Time taken</b>	27 secs
<b>Marks</b>	1.00/1.00
<b>Grade</b>	<b>10.00</b> out of 10.00 ( <b>100%</b> )

## Question 1

Correct

Mark 1.00 out of 1.00

Given two strings find the length of the common longest subsequence(need not be contiguous) between the two.

Example:

s1: ggtabe

s2: tgatasb

s1	a	g	g	t	a	b	
s2	g	x	t	x	a	y	b

**The length is 4**

Solveing it using Dynamic Programming

For example:

Input	Result
aab	2
azb	

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

```
1 #include<stdio.h>
2 #include <stdio.h>
3 #include <string.h>
4
5 int max(int a, int b) {
6     return (a > b) ? a : b;
7 }
8
9 int lcs(char *s1, char *s2) {
10     int m = strlen(s1);
11     int n = strlen(s2);
12     int dp[m+1][n+1];
13
14     for (int i = 0; i <= m; i++) {
15         for (int j = 0; j <= n; j++) {
16             if (i == 0 || j == 0) {
17                 dp[i][j] = 0;
18             } else if (s1[i-1] == s2[j-1]) {
19                 dp[i][j] = dp[i-1][j-1] + 1;
20             } else {
21                 dp[i][j] = max(dp[i-1][j], dp[i][j-1]);
22             }
23         }
24     }
25
26     return dp[m][n];
27 }
28
29 int main() {
30     char s1[100], s2[100];
31
32     scanf("%s", s1);
33
34     scanf("%s", s2);
35
36     printf("%d\n", lcs(s1, s2));
37     return 0;
38 }
39
```



	Input	Expected	Got	
✓	aab azb	2	2	✓
✓	ABCD ABCD	4	4	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

◀ 2-DP-Playing with chessboard

Jump to...

4-DP-Longest non-decreasing Subsequence ▶

<b>Started on</b>	Thursday, 10 October 2024, 8:31 AM
<b>State</b>	Finished
<b>Completed on</b>	Thursday, 10 October 2024, 8:31 AM
<b>Time taken</b>	12 secs
<b>Marks</b>	1.00/1.00
<b>Grade</b>	<b>10.00</b> out of 10.00 ( <b>100%</b> )

## Question 1

Correct

Mark 1.00 out of 1.00

Problem statement:

Find the length of the Longest Non-decreasing Subsequence in a given Sequence.

Eg:

Input:9

Sequence: [-1,3,4,5,2,2,2,2,3]

the subsequence is [-1,2,2,2,2,3]

Output:6

Answer: (penalty regime: 0 %)

```

1 #include <stdio.h>
2
3 int longestNonDecreasingSubsequence(int arr[], int n) {
4     int lengths[n];
5     for (int i = 0; i < n; i++) {
6         lengths[i] = 1;
7     }
8
9     for (int i = 1; i < n; i++) {
10        for (int j = 0; j < i; j++) {
11            if (arr[i] >= arr[j] && lengths[i] < lengths[j] + 1) {
12                lengths[i] = lengths[j] + 1;
13            }
14        }
15    }
16
17    int maxLength = 0;
18    for (int i = 0; i < n; i++) {
19        if (maxLength < lengths[i]) {
20            maxLength = lengths[i];
21        }
22    }
23
24    return maxLength;
25 }
26
27 int main() {
28     int arr[] = {-1, 3, 4, 5, 2, 2, 2, 2, 3};
29     int n = sizeof(arr) / sizeof(arr[0]);
30     int result = longestNonDecreasingSubsequence(arr, n);
31     printf("%d\n", result);
32     return 0;
33 }
34

```

	Input	Expected	Got	
✓	9 -1 3 4 5 2 2 2 2 3	6	6	✓
✓	7 1 2 2 4 5 7 6	6	6	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Jump to...

[1-Finding Duplicates- \$O\(n^2\)\$  Time Complexity, \$O\(1\)\$  Space Complexity](#) ►