

---

**Started on** Wednesday, 15 October 2025, 10:40 PM

---

**State** Finished

---

**Completed on** Wednesday, 15 October 2025, 11:06 PM

---

**Time taken** 25 mins 16 secs

---

**Grade** 10.00 out of 10.00 (100%)

### 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 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 long long countWays(int n) {
3     if (n == 0) return 1;
4     if (n == 1 || n == 2) return 1;
5     if (n == 3) return 2;
6
7     long long a = 1, b=1, c = 2, current;
8     for (int i = 4; i <= n; i++) {
9         current = c + a;
10        a = b;
11        b = c;
12        c = current;
13    }
14    return c;
15 }
16
17 int main() {
18     int n;
19     scanf("%d", &n);
20     printf("%lld", countWays(n));
21     return 0;
22 }
```

	<b>Input</b>	<b>Expected</b>	<b>Got</b>	
✓	6	6	6	✓
✓	25	8641	8641	✓
✓	100	24382819596721629	24382819596721629	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

[Back to Course](#)

---

**Started on** Saturday, 25 October 2025, 9:44 AM

---

**State** Finished

---

**Completed on** Saturday, 25 October 2025, 9:44 AM

---

**Time taken** 35 secs

---

**Grade** 10.00 out of 10.00 (100%)

**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 #define MAX 100
4
5 int max(int a, int b) {
6     return (a > b) ? a : b;
7 }
8
9 int main() {
10    int n;
11    int board[MAX][MAX];
12    int dp[MAX][MAX];
13
14    scanf("%d", &n);
15
16    for (int i = 0; i < n; i++)
17        for (int j = 0; j < n; j++)
18            scanf("%d", &board[i][j]);
19
20    dp[0][0] = board[0][0];
21
22    for (int j = 1; j < n; j++)
23        dp[0][j] = dp[0][j - 1] + board[0][j];
24
25    for (int i = 1; i < n; i++)
26        dp[i][0] = dp[i - 1][0] + board[i][0];
27
28    for (int i = 1; i < n; i++) {
29        for (int j = 1; j < n; j++) {
30            dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]) + board[i][j];
31        }
32    }
33}
```

```
32     }
33     printf("%d\n", dp[n - 1][n - 1]);
34
35     return 0;
36 }
37 }
```

	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.



[Back to Course](#)

---

**Started on** Saturday, 25 October 2025, 9:45 AM

---

**State** Finished

---

**Completed on** Saturday, 25 October 2025, 9:45 AM

---

**Time taken** 18 secs

---

**Marks** 1.00/1.00

---

**Grade** **10.00** out of 10.00 (**100%**)

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	<b>g</b>	t	a	b	
s2	<b>g</b>	x	<b>t</b>	x	a	y	b

**The length is 4**

Solving it using Dynamic Programming

For example:

Input	Result
aab	2
azb	

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

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

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)



---

**Started on** Saturday, 25 October 2025, 9:45 AM

---

**State** Finished

---

**Completed on** Saturday, 25 October 2025, 9:45 AM

---

**Time taken** 19 secs

---

**Marks** 1.00/1.00

---

**Grade** **10.00** out of 10.00 (**100%**)

## 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 #define MAX 1000
4
5 int max(int a, int b) {
6     return (a > b) ? a : b;
7 }
8
9 int longestNonDecreasingSubsequence(int arr[], int n) {
10    int dp[MAX];
11
12    for (int i = 0; i < n; i++)
13        dp[i] = 1;
14
15    for (int i = 1; i < n; i++) {
16        for (int j = 0; j < i; j++) {
17            if (arr[i] >= arr[j])
18                dp[i] = max(dp[i], dp[j] + 1);
19        }
20    }
21
22    int result = 0;
23    for (int i = 0; i < n; i++)
24        result = max(result, dp[i]);
25
26    return result;
27 }
28
29 int main() {
30    int n;
31    scanf("%d", &n);
32
33    int arr[MAX];
34    for (int i = 0; i < n; i++)
35        scanf("%d", &arr[i]);
36
37    int length = longestNonDecreasingSubsequence(arr, n);
38    printf("%d\n", length);
39
40    return 0;
41 }
42

```

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

	Input	Expected	Got	
✓	7 1 2 2 4 5 7 6	6	6	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)