



1-DP-Playing with Numbers

Started on Sunday, 12 October 2025, 10:05 AM

State Finished

Completed on Sunday, 12 October 2025, 10:19 AM

Time taken 13 mins 51 secs

Grade 10.00 out of 10.00 (100%)

Question 1 | Correct Mark 10.00 out of 10.00  [Flag question](#)

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

Sample Input

6

Sample Output

6

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 long long calculate_ways(int n) {
3     if(n < 0) {
4         return 0;
5     }
6     if(n == 0) {
7         return 1;
8     }
9     long long dp3 = 1;
10    long long dp2 = 1;
11    long long dp1 = 1;
12    long long current_dp = 0;
13    if(n == 1)
14        return dp2;
15    if(n == 2)
16        return dp1;
17    for(int i = 3; i <= n; i++) {
18        current_dp = dp1 + dp3;
19        dp3 = dp2;
20        dp2 = dp1;
21        dp1 = current_dp;
22    }
23    return dp1;
24 }
25 int main() {
26     int n;
27     if(scanf("%d", &n) != 1) {
28         return 1;
29     }
30     long long result = calculate_ways(n);
31     printf("%lld\n", result);
32     return 0;
33 }
```

	Input	Expected	Got	
✓	6	6	6	✓
✓	25	8641	8641	✓
✓	100	24382819596721629	24382819596721629	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

[Finish review](#)

[Back to Course](#)


[Data retention summary](#)

CS23331-DAA-2024-CSE / 2-DP-Playing with chessboard



2-DP-Playing with chessboard

Started on	Saturday, 18 October 2025, 8:11 AM
State	Finished
Completed on	Saturday, 18 October 2025, 8:19 AM
Time taken	8 mins 30 secs
Grade	10.00 out of 10.00 (100%)

Question 1 | Correct Mark 10.00 out of 10.00  [Flag question](#)

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 int main() {
3     int n;
4     scanf("%d", &n);
5     int a[n][n];
6     for(int i = 0; i < n; i++) {
7         for(int j = 0; j < n; j++) {
8             scanf("%d", &a[i][j]);
9         }
10    }
11    int dp[n][n];
12    dp[0][0] = a[0][0];
13    for(int j = 1; j < n; j++) {
14        dp[0][j] = dp[0][j - 1] + a[0][j];
15    }
16    for(int i = 1; i < n; i++) {
17        dp[i][0] = dp[i - 1][0] + a[i][0];
18    }
19    for(int i = 1; i < n; i++) {
20        for(int j = 1; j < n; j++) {
21            if(dp[i - 1][j] > dp[i][j - 1]) {
22                dp[i][j] = dp[i - 1][j] + a[i][j];
23            }
24            else {
25                dp[i][j] = dp[i][j - 1] + a[i][j];
26            }
27        }
28    }
29    printf("%d", dp[n - 1][n - 1]);
30    return 0;
31 }
```

	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	28	28	✓

1	5	7	8			
2	3	4	6			
1	6	9	0			

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

[Finish review](#)

[Back to Course](#)

[Data retention summary](#)



3-DP-Longest Common Subsequence

Started on Saturday, 18 October 2025, 8:35 AM


State Finished

Completed on Saturday, 18 October 2025, 8:44 AM

Time taken 8 mins 38 secs

Marks 1.00/1.00

Grade 10.00 out of 10.00 (100%)

Question 1 | Correct Mark 1.00 out of 1.00  [Flag question](#)

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

```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Finish review](#)

[Back to Course](#)

Data retention summary

CS23331-DAA-2024-CSE / 4-DP-Longest non-decreasing Subsequence



4-DP-Longest non-decreasing Subsequence

Started on	Saturday, 18 October 2025, 8:47 AM
State	Finished
Completed on	Saturday, 18 October 2025, 8:52 AM
Time taken	5 mins 44 secs
Marks	1.00/1.00
Grade	10.00 out of 10.00 (100%)

Question 1 | Correct Mark 1.00 out of 1.00 [Flag question](#)

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 int max(int a, int b) {
3     return (a > b) ? a : b;
4 }
5 int main() {
6     int n;
7     scanf("%d", &n);
8     int arr[n], dp[n];
9     for(int i = 0; i < n; i++) {
10        scanf("%d", &arr[i]);

```

```

10     scanf("%d", &arr[i]);
11     dp[i] = 1;
12 }
13 int result = 1;
14 for(int i = 1; i < n; i++) {
15     for(int j = 0; j < i; j++) {
16         if(arr[i] >= arr[j]) {
17             dp[i] = max(dp[i], dp[j] + 1);
18         }
19     }
20     result = max(result, dp[i]);
21 }
22 printf("%d", result);
23 return 0;
24 }

```

	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.

[Finish review](#)

[Back to Course](#)

[Data retention summary](#)