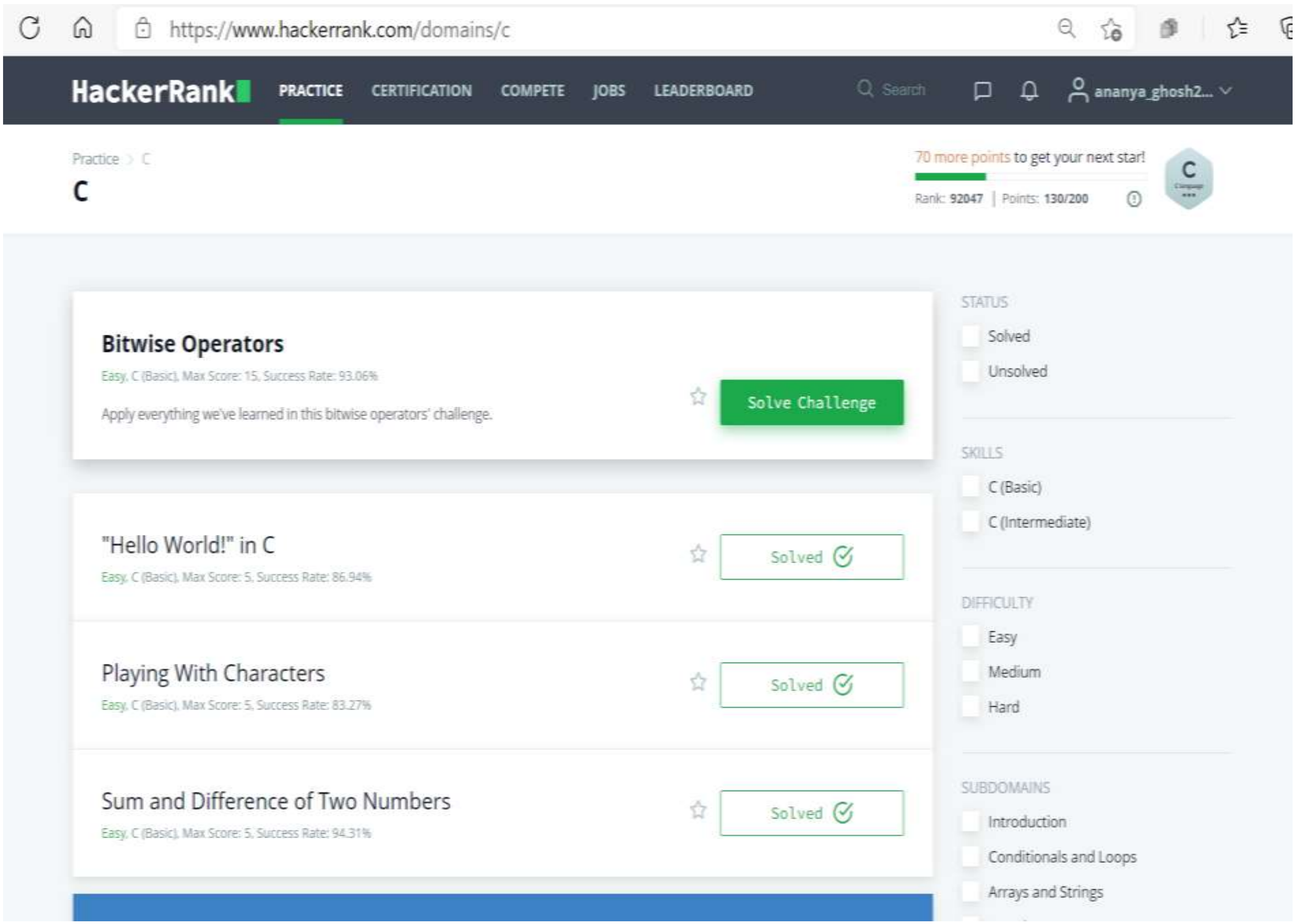
Ananya Ghosh

20MIC0063

Task 2


https://www.hackerrank.com/domains/c

HackerRank PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search ananya_ghosh2...

Practice > C

C

70 more points to get your next star!

Rank: 92047 | Points: 130/200

Bitwise Operators

Easy, C (Basic), Max Score: 15, Success Rate: 93.06%

Apply everything we've learned in this bitwise operators' challenge.

Solve Challenge

"Hello World!" in C

Easy, C (Basic), Max Score: 5, Success Rate: 86.94%

Solved

Playing With Characters

Easy, C (Basic), Max Score: 5, Success Rate: 83.27%

Solved

Sum and Difference of Two Numbers

Easy, C (Basic), Max Score: 5, Success Rate: 94.31%

Solved

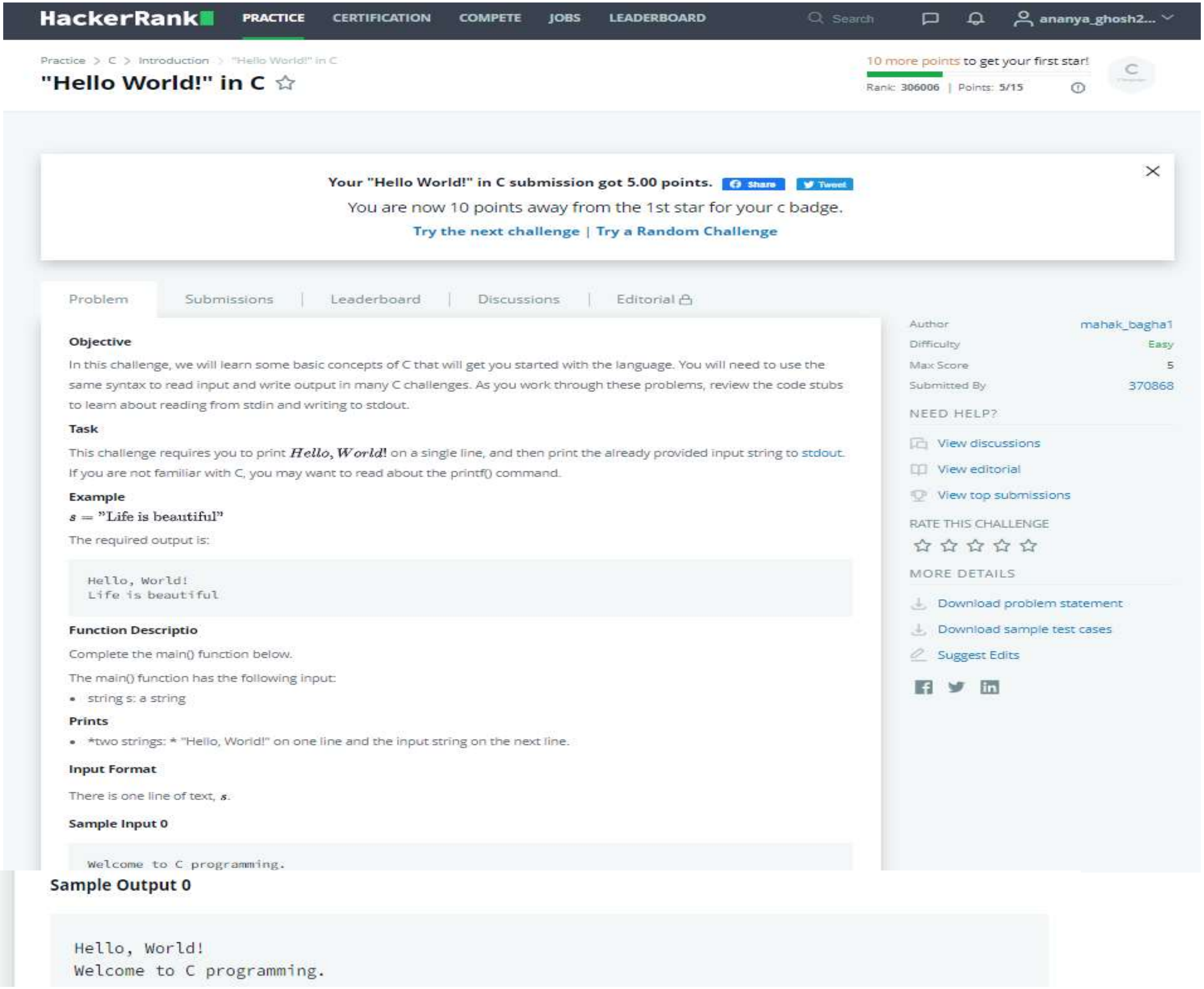
STATUS
- Solved
- Unsolved

SKILLS
- C (Basic)
- C (Intermediate)

DIFFICULTY
- Easy
- Medium
- Hard

SUBDOMAINS
- Introduction
- Conditionals and Loops
- Arrays and Strings

1.


HackerRank PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search ananya_ghosh2...

Practice > C > Introduction > "Hello World!" in C

"Hello World!" in C

10 more points to get your first star!
Rank: 306006 | Points: 5/15

Your "Hello World!" in C submission got 5.00 points. Share Tweet

You are now 10 points away from the 1st star for your c badge.

Try the next challenge | Try a Random Challenge

Problem | Submissions | Leaderboard | Discussions | Editorial

Objective

In this challenge, we will learn some basic concepts of C that will get you started with the language. You will need to use the same syntax to read input and write output in many C challenges. As you work through these problems, review the code stubs to learn about reading from stdin and writing to stdout.

Task

This challenge requires you to print $Hello, World!$ on a single line, and then print the already provided input string to stdout. If you are not familiar with C, you may want to read about the printf() command.

Example

$s =$ "Life is beautiful"

The required output is:

```
Hello, World!
Life is beautiful
```

Function Descriptio

Complete the main() function below.

The main() function has the following input:

- string s: a string

Prints

- *two strings: * "Hello, World!" on one line and the input string on the next line.

Input Format

There is one line of text, s.

Sample Input 0

```
Welcome to C programming.
```

Sample Output 0

```
Hello, World!
Welcome to C programming.
```

Author	mahak_bagha1
Difficulty	Easy
Max Score	5
Submitted By	370868

NEED HELP?

View discussions

View editorial

View top submissions

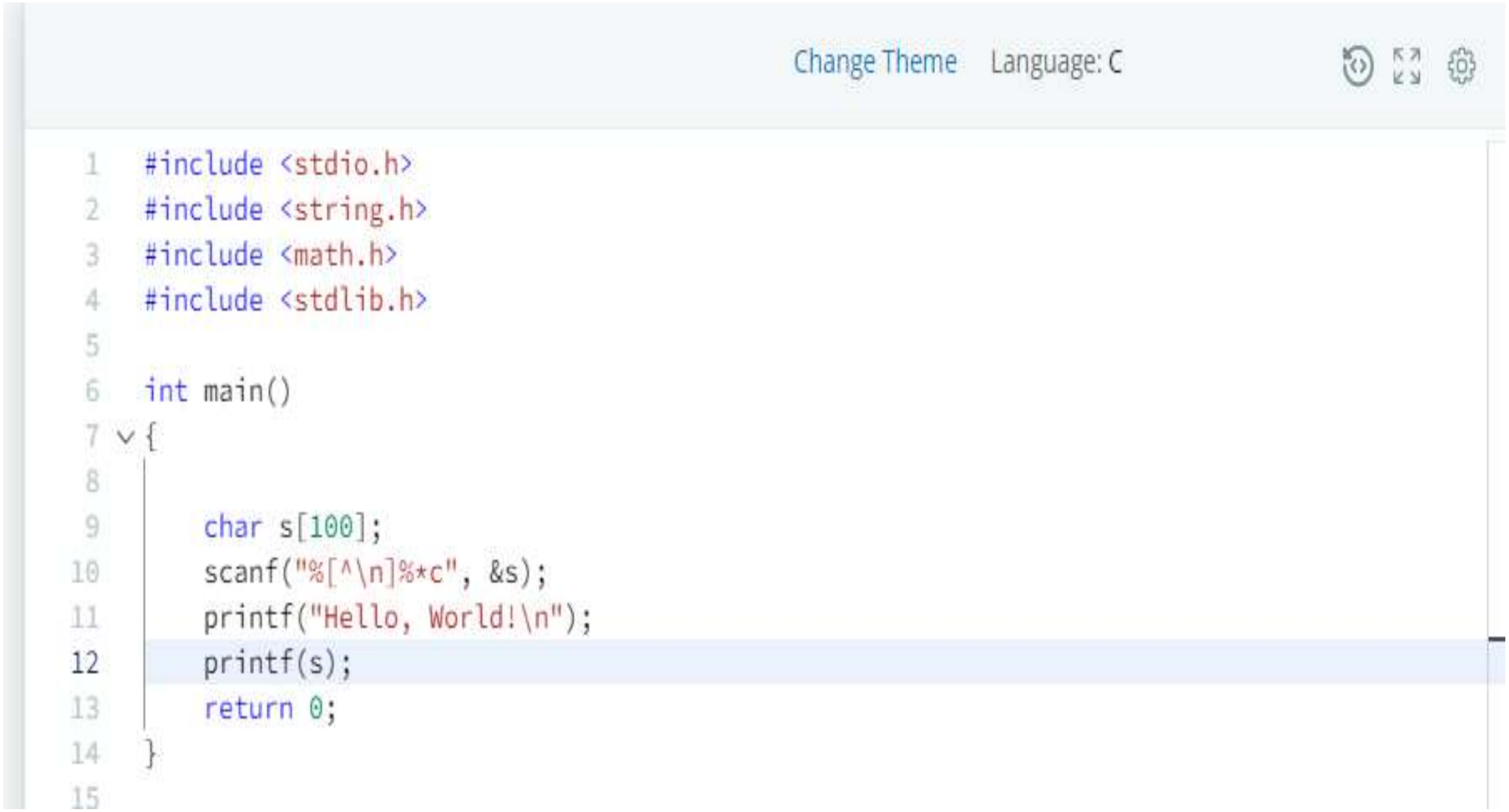
RATE THIS CHALLENGE

MORE DETAILS

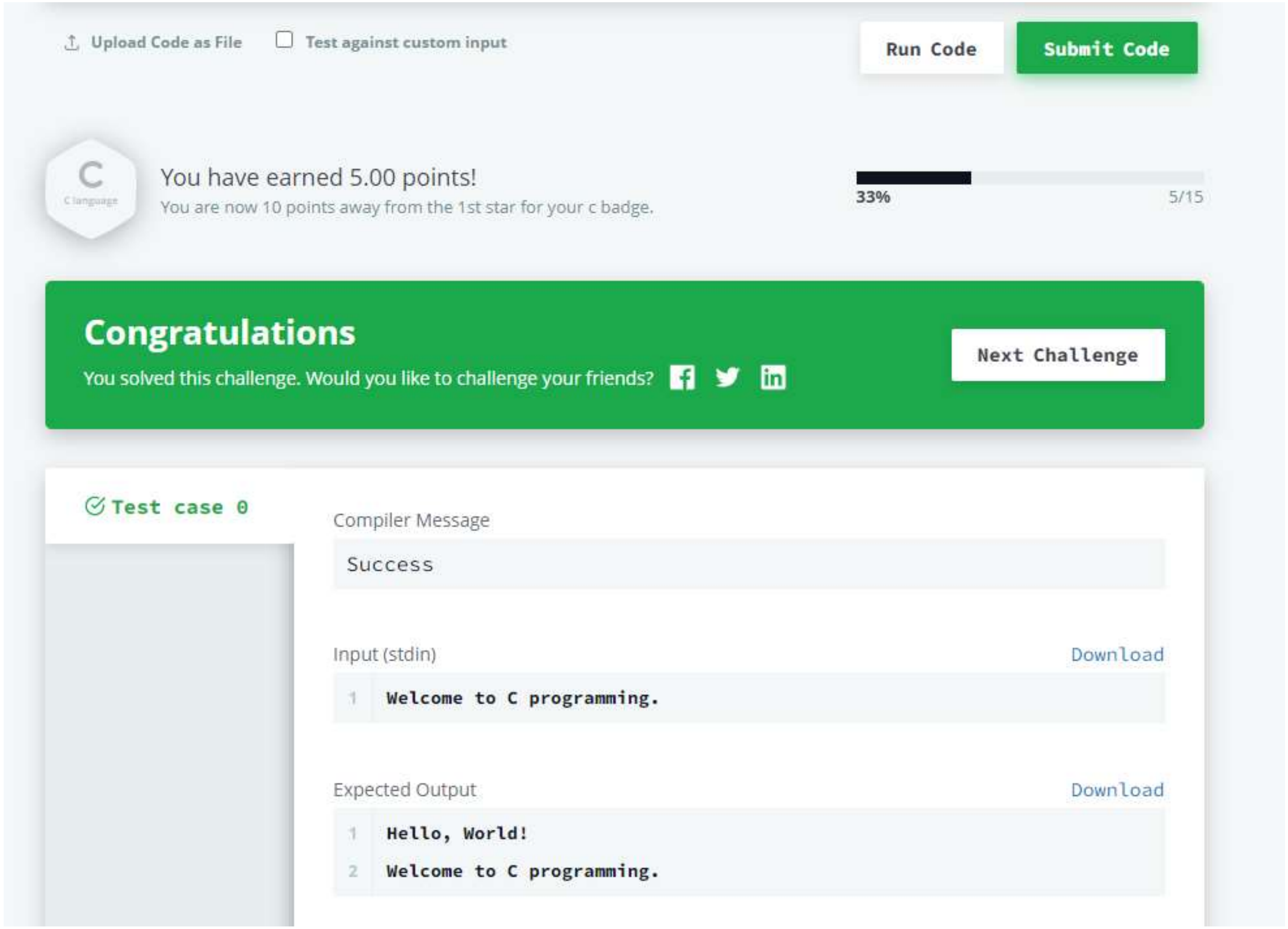
Download problem statement

Download sample test cases

Suggest Edits



```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

int main()
{

    char s[100];
    scanf("%[^\n]%*c", &s);
    printf("Hello, World!\n");
    printf(s);
    return 0;
}

```



You have earned 5.00 points!

You are now 10 points away from the 1st star for your c badge.

33% 5/15

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

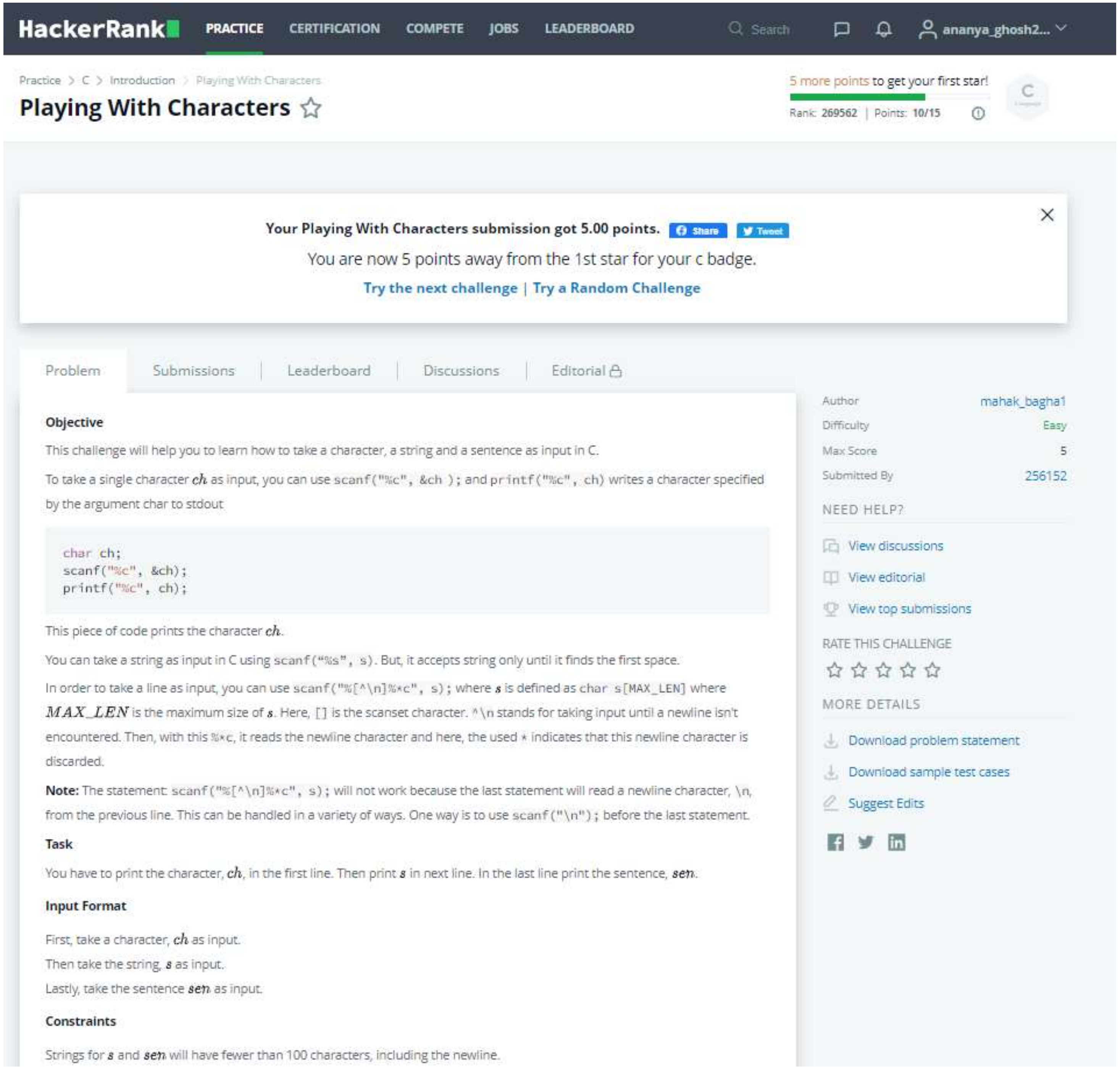
Input (stdin) Download

```
Welcome to C programming.
```

Expected Output Download

```
Hello, World!
Welcome to C programming.
```

2.



HackerRank PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search ananya_ghosh2...

Practice > C > Introduction > Playing With Characters

Playing With Characters

5 more points to get your first star!
Rank: 269562 | Points: 10/15

Your Playing With Characters submission got 5.00 points. Share Tweet

You are now 5 points away from the 1st star for your c badge.

Try the next challenge | Try a Random Challenge

Problem Submissions Leaderboard Discussions Editorial

Objective

This challenge will help you to learn how to take a character, a string and a sentence as input in C.

To take a single character ch as input, you can use `scanf("%c", &ch );` and `printf("%c", ch)` writes a character specified by the argument char to stdout

```
char ch;
scanf("%c", &ch);
printf("%c", ch);
```

This piece of code prints the character ch.

You can take a string as input in C using `scanf("%s", s)`. But, it accepts string only until it finds the first space.

In order to take a line as input, you can use `scanf("%[^\n]%*c", s);` where s is defined as `char s[MAX_LEN]` where MAX_LEN is the maximum size of s. Here, `[]` is the scanset character. `^\n` stands for taking input until a newline isn't encountered. Then, with this `%*c`, it reads the newline character and here, the used `*` indicates that this newline character is discarded.

Note: The statement: `scanf("%[^\n]%*c", s);` will not work because the last statement will read a newline character, `\n`, from the previous line. This can be handled in a variety of ways. One way is to use `scanf("\n");` before the last statement.

Task

You have to print the character, ch, in the first line. Then print s in next line. In the last line print the sentence, sen.

Input Format

First, take a character, ch as input.

Then take the string, s as input.

Lastly, take the sentence sen as input.

Constraints

Strings for s and sen will have fewer than 100 characters, including the newline.

Author	mahak_bagha1
Difficulty	Easy
Max Score	5
Submitted By	256152

NEED HELP?
View discussions
View editorial
View top submissions

RATE THIS CHALLENGE

MORE DETAILS
Download problem statement
Download sample test cases
Suggest Edits

from the previous line. This can be handled in a variety of ways. One way is to use `scanf("\n");` before the last statement.

Task

You have to print the character, ch, in the first line. Then print s in next line. In the last line print the sentence, sen.

Input Format

First, take a character, ch as input.

Then take the string, s as input.

Lastly, take the sentence sen as input.

Constraints

Strings for s and sen will have fewer than 100 characters, including the newline.

Output Format

Print three lines of output. The first line prints the character, ch.

The second line prints the string, s.

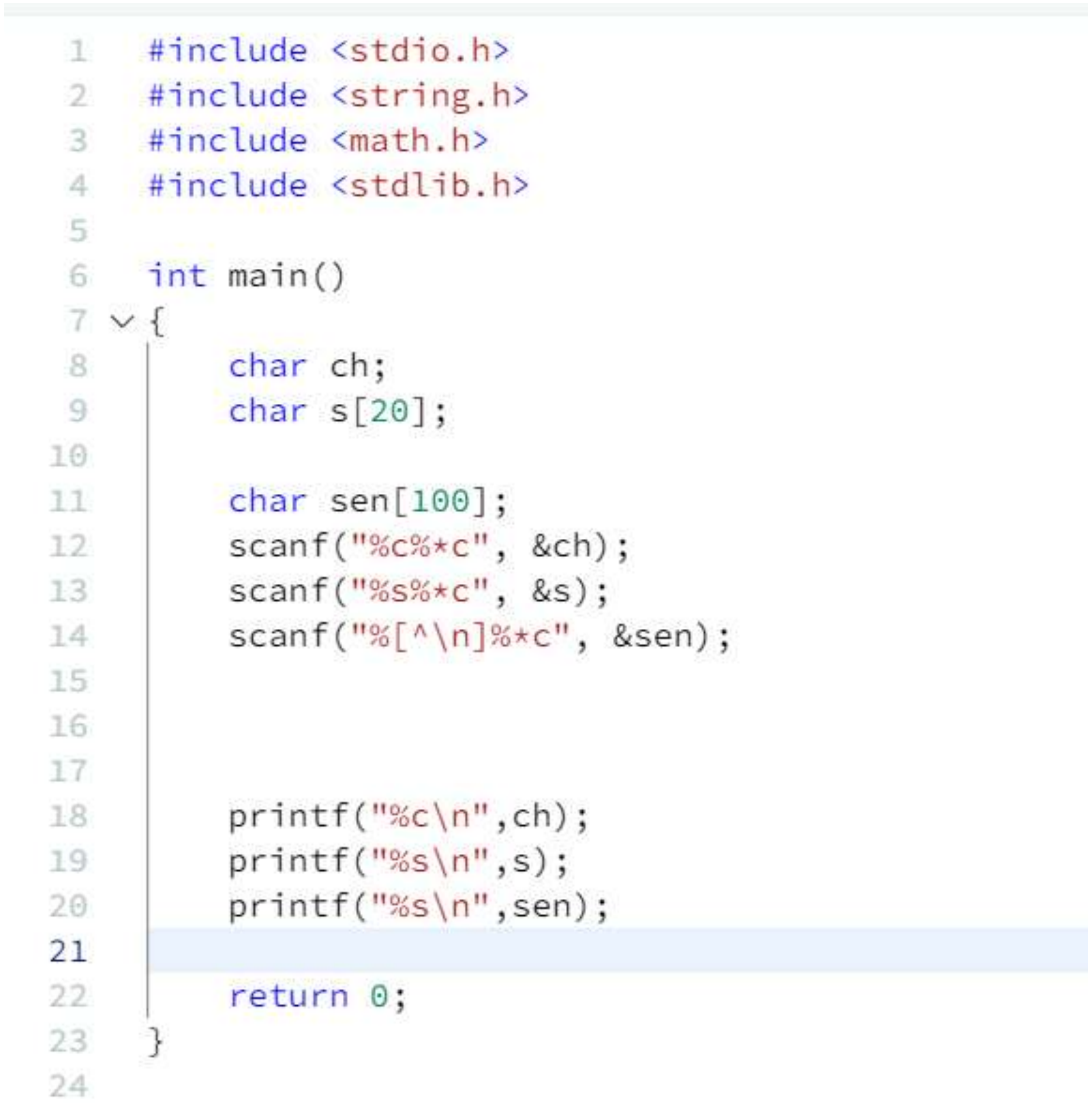
The third line prints the sentence, sen.

Sample Input 0

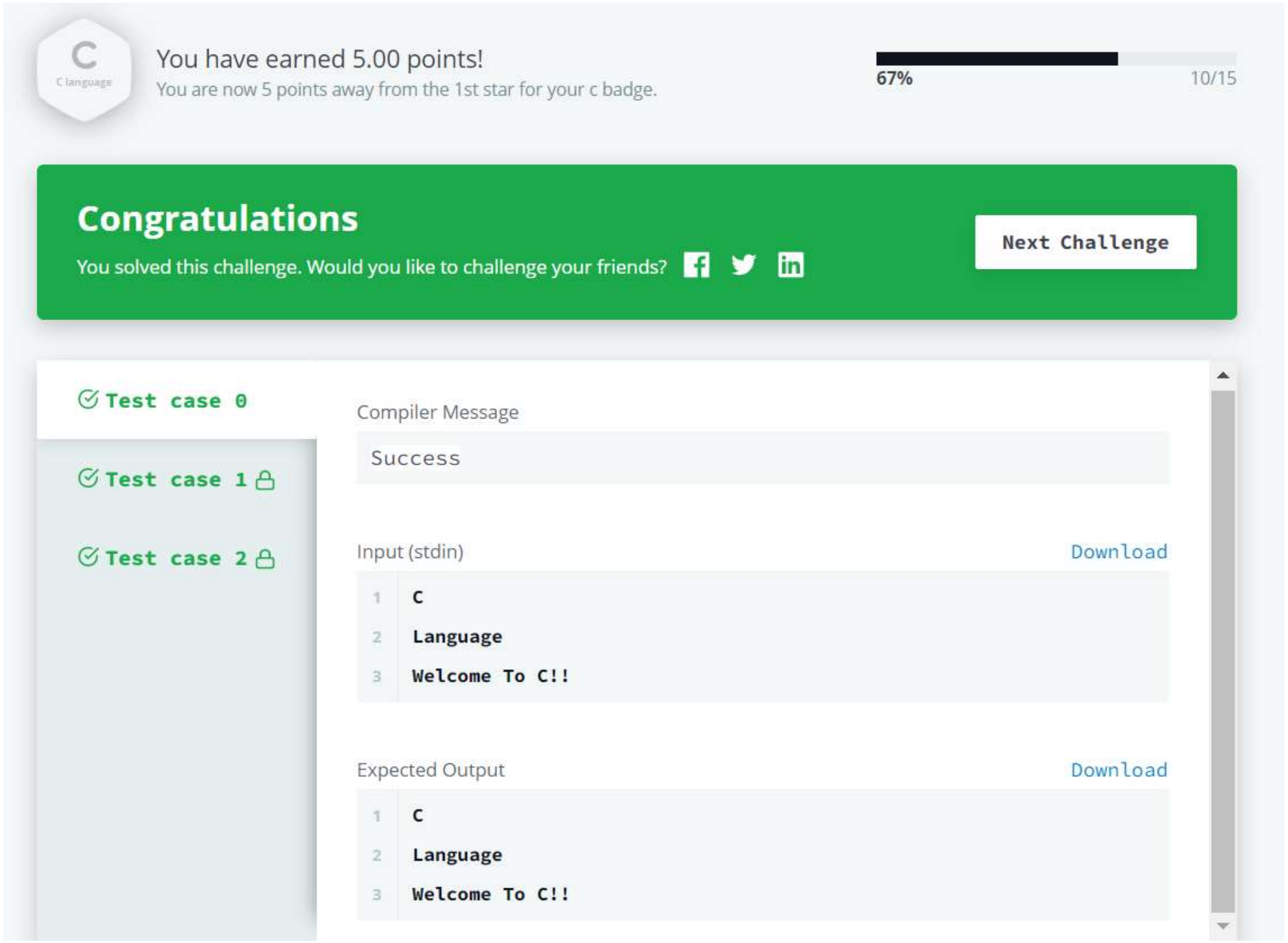
```
C
Language
Welcome To C!!
```

Sample Output 0

```
C
Language
Welcome To C!!
```



```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

int main()
{
    char ch;
    char s[20];

    char sen[100];
    scanf("%c%*c", &ch);
    scanf("%s%*c", &s);
    scanf("%[^\n]%*c", &sen);



    printf("%c\n",ch);
    printf("%s\n",s);
    printf("%s\n",sen);

    return 0;
}

```



You have earned 5.00 points!

You are now 5 points away from the 1st star for your c badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

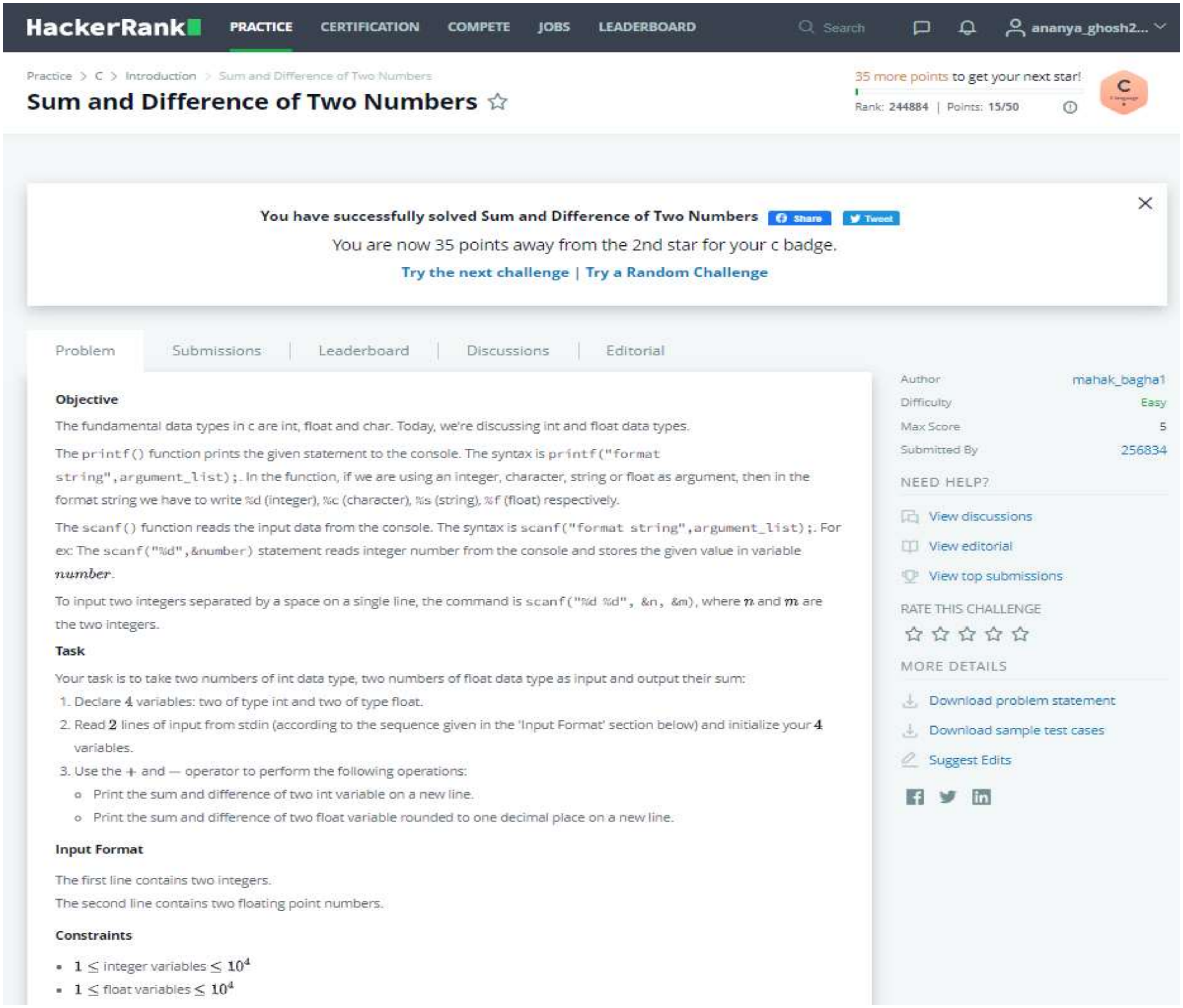
Input (stdin)

```
C
Language
Welcome To C!!
```

Expected Output

```
C
Language
Welcome To C!!
```

3.



Practice > C > Introduction > Sum and Difference of Two Numbers

Sum and Difference of Two Numbers

35 more points to get your next star!
Rank: 244884 | Points: 15/50

You have successfully solved Sum and Difference of Two Numbers Share Tweet

You are now 35 points away from the 2nd star for your c badge.

Try the next challenge | Try a Random Challenge

Objective

The fundamental data types in c are int, float and char. Today, we're discussing int and float data types.

The `printf()` function prints the given statement to the console. The syntax is `printf("format string",argument_list);`. In the function, if we are using an integer, character, string or float as argument, then in the format string we have to write %d (integer), %c (character), %s (string), %f (float) respectively.

The `scanf()` function reads the input data from the console. The syntax is `scanf("format string",argument_list);`. For ex: The `scanf("%d",&number)` statement reads integer number from the console and stores the given value in variable $number$.

To input two integers separated by a space on a single line, the command is `scanf("%d %d", &n, &m)`, where n and m are the two integers.

Task

Your task is to take two numbers of int data type, two numbers of float data type as input and output their sum:

1. Declare 4 variables: two of type int and two of type float.
2. Read 2 lines of input from stdin (according to the sequence given in the 'Input Format' section below) and initialize your 4 variables.
3. Use the $+$ and $-$ operator to perform the following operations:
   - Print the sum and difference of two int variable on a new line.
   - Print the sum and difference of two float variable rounded to one decimal place on a new line.

Input Format

The first line contains two integers.
The second line contains two floating point numbers.

Constraints

- $1 \leq$ integer variables $\leq 10^4$
- $1 \leq$ float variables $\leq 10^4$

Author	mahak_bagha1
Difficulty	Easy
Max Score	5
Submitted By	256834

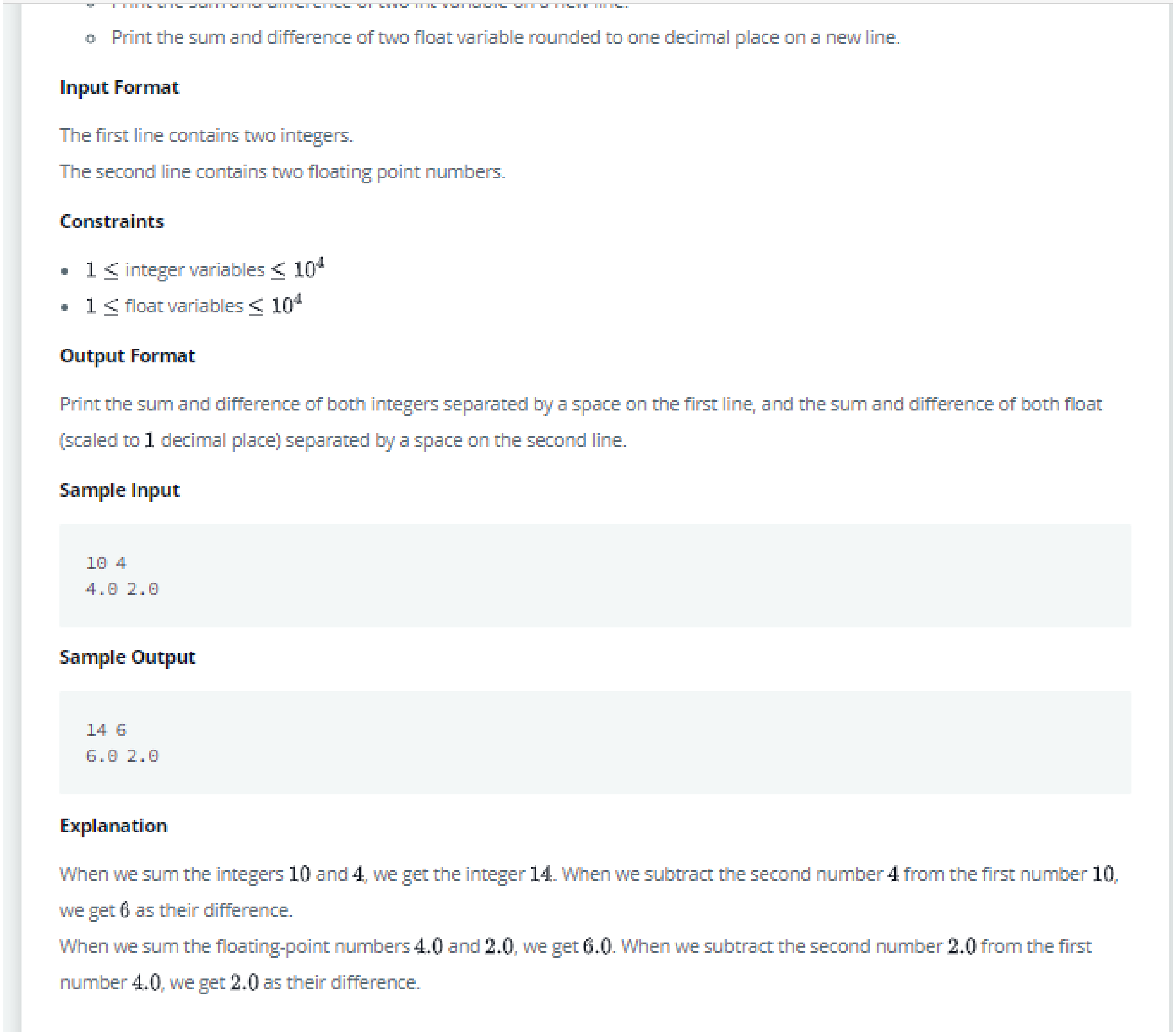
NEED HELP?

View discussions
View editorial
View top submissions

RATE THIS CHALLENGE

MORE DETAILS

Download problem statement
Download sample test cases
Suggest Edits



  - Print the sum and difference of two int variable on a new line.
  - Print the sum and difference of two float variable rounded to one decimal place on a new line.

Input Format

The first line contains two integers.

The second line contains two floating point numbers.

Constraints

- $1 \leq$ integer variables $\leq 10^4$
- $1 \leq$ float variables $\leq 10^4$

Output Format

Print the sum and difference of both integers separated by a space on the first line, and the sum and difference of both float (scaled to 1 decimal place) separated by a space on the second line.

Sample Input

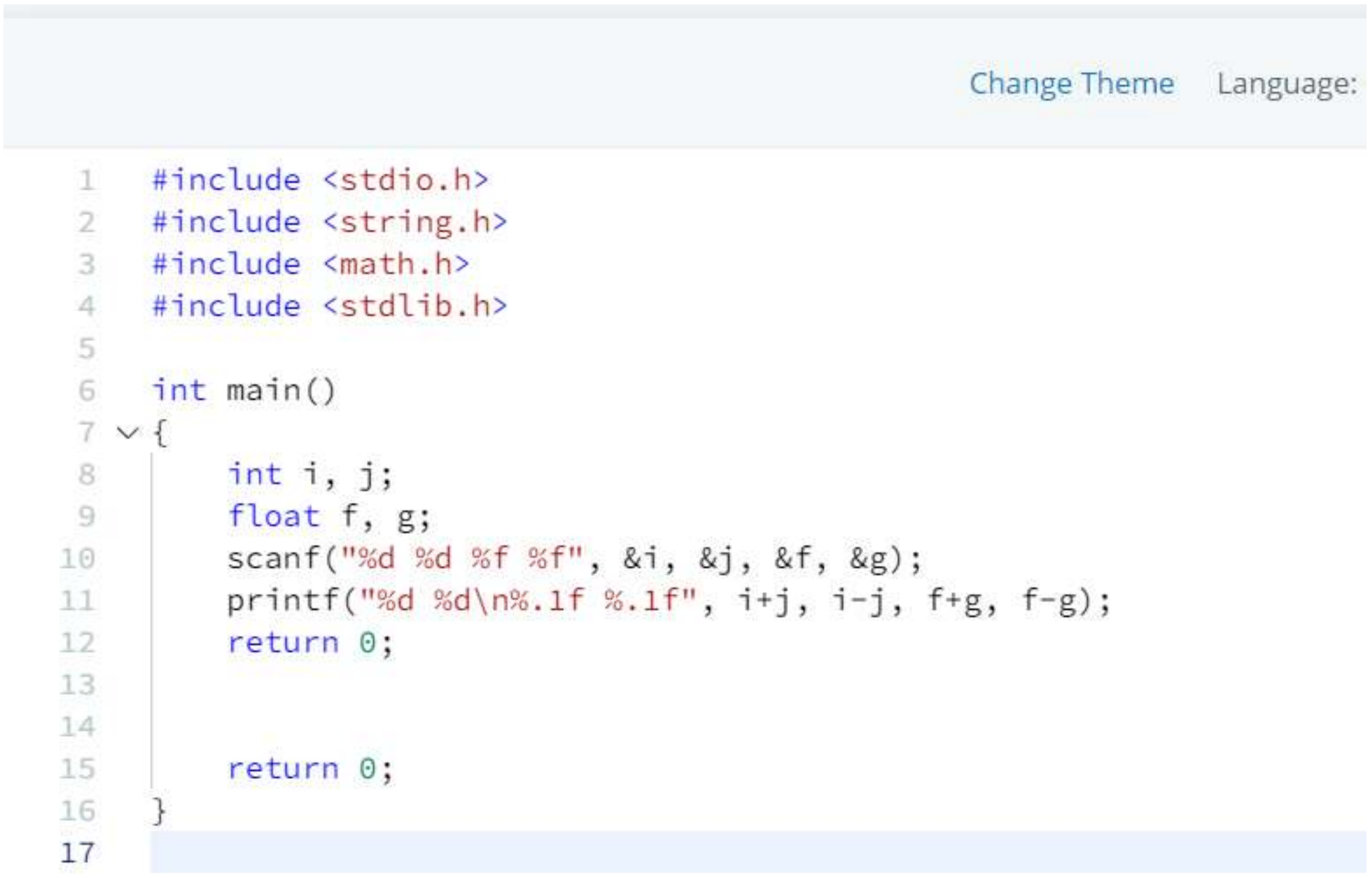
```
10 4
4.0 2.0
```

Sample Output

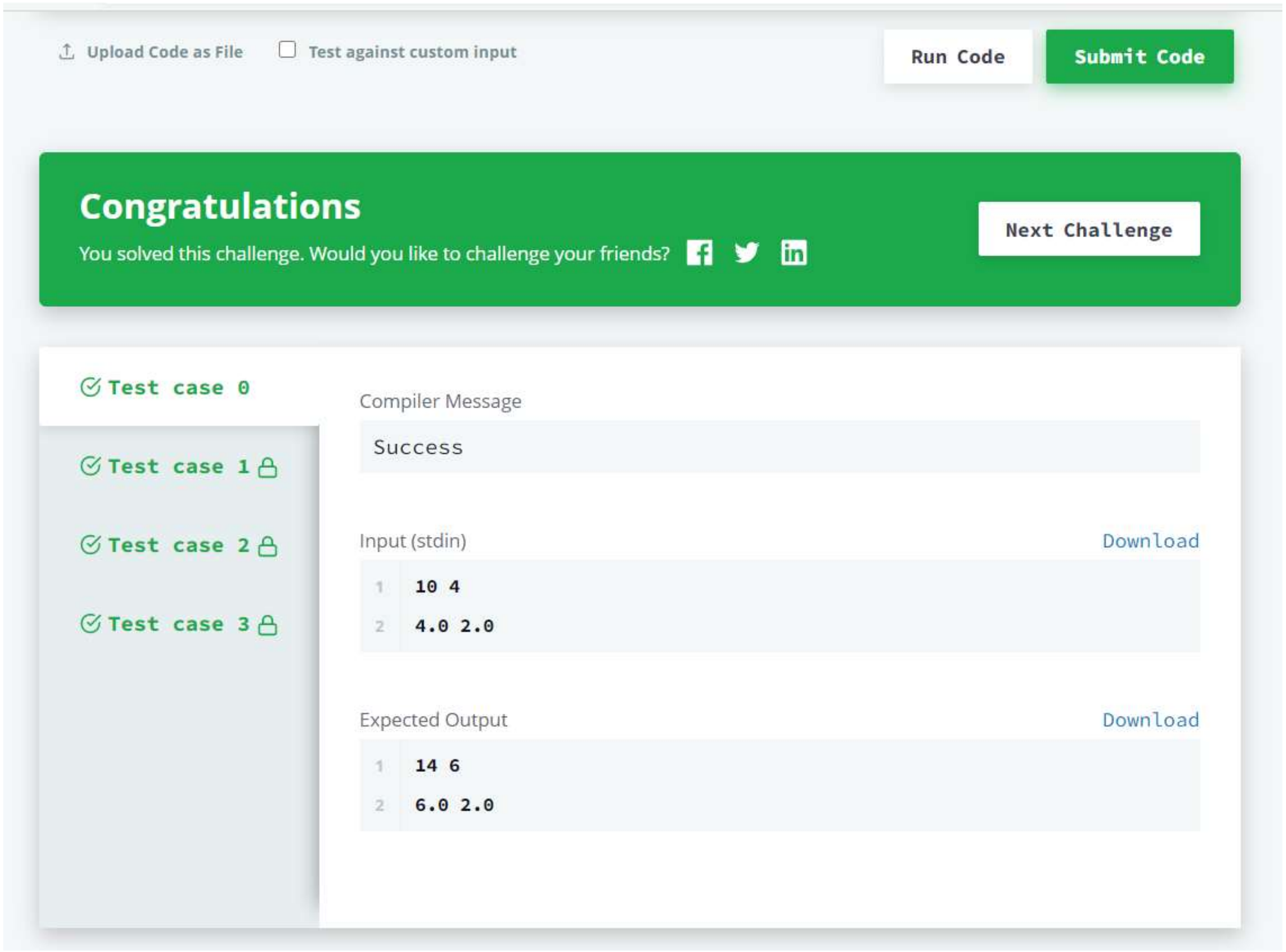
```
14 6
6.0 2.0
```

Explanation

When we sum the integers 10 and 4, we get the integer 14. When we subtract the second number 4 from the first number 10, we get 6 as their difference.
When we sum the floating-point numbers 4.0 and 2.0, we get 6.0. When we subtract the second number 2.0 from the first number 4.0, we get 2.0 as their difference.



```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

int main()
{
    int i, j;
    float f, g;
    scanf("%d %d %f %f", &i, &j, &f, &g);
    printf("%d %d\n%.1f %.1f", i+j, i-j, f+g, f-g);
    return 0;


    return 0;
}

```



Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

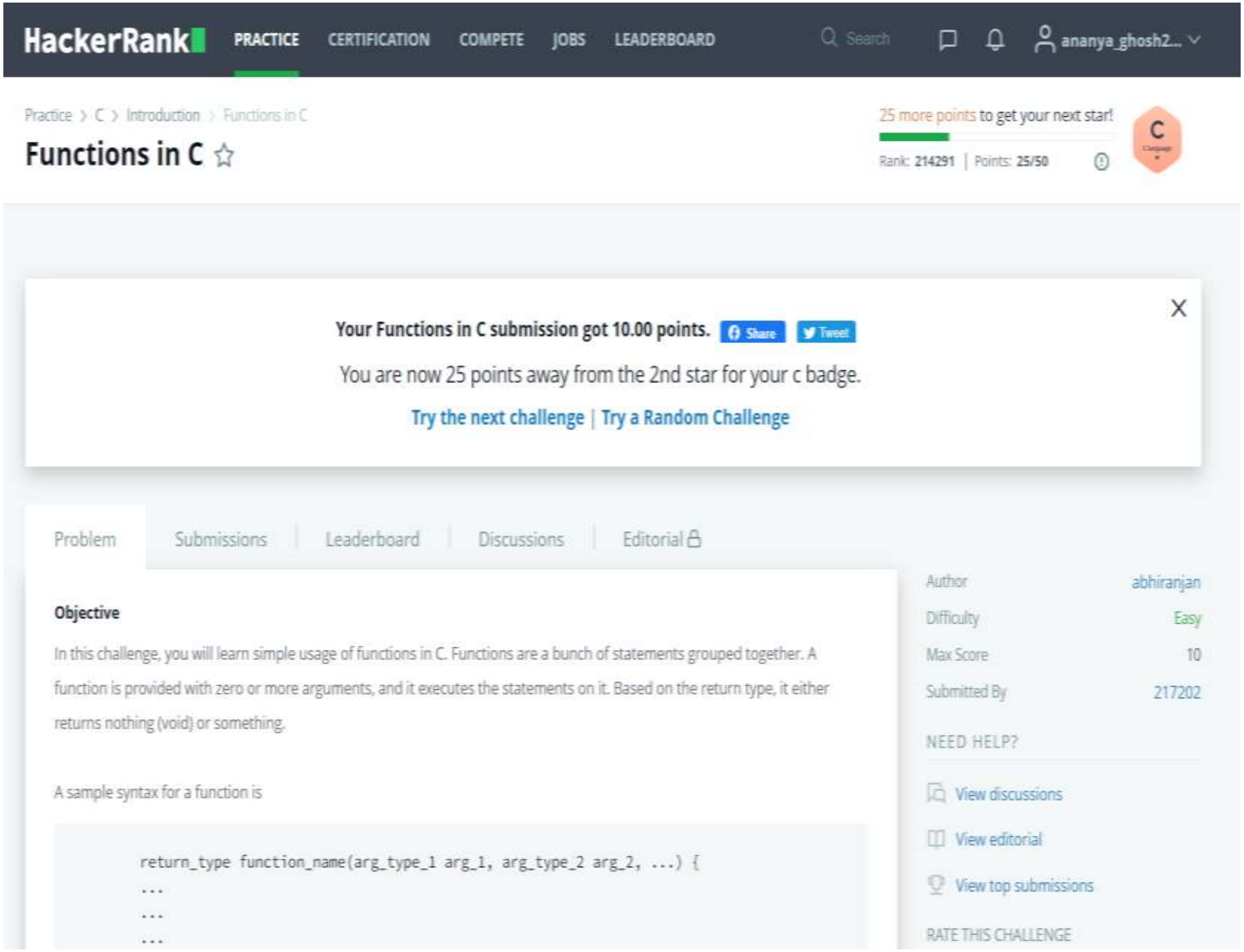
Input (stdin)

```
10 4
4.0 2.0
```

Expected Output

```
14 6
6.0 2.0
```

4.



Practice > C > Introduction > Functions in C

Functions in C

25 more points to get your next star!

Rank: 214291 | Points: 25/50

Your Functions in C submission got 10.00 points.

You are now 25 points away from the 2nd star for your c badge.

Try the next challenge | Try a Random Challenge

Objective

In this challenge, you will learn simple usage of functions in C. Functions are a bunch of statements grouped together. A function is provided with zero or more arguments, and it executes the statements on it. Based on the return type, it either returns nothing (void) or something.

A sample syntax for a function is

```
return_type function_name(arg_type_1 arg_1, arg_type_2 arg_2, ...) {
    ...
    ...
    ...
```

Author	abhiranjan
Difficulty	Easy
Max Score	10
Submitted By	217202

NEED HELP?

function is provided with zero or more arguments, and it executes the statements on it. Based on the return type, it either returns nothing (void) or something.

A sample syntax for a function is

```
        return_type function_name(arg_type_1 arg_1, arg_type_2 arg_2, ...) {
        ...
        ...
        ...
        [if return_type is non void]
                return something of type `return_type`;
    }
```

For example, a function to read four variables and return the sum of them can be written as

```
        int sum_of_four(int a, int b, int c, int d) {
        int sum = 0;
        sum += a;
        sum += b;
        sum += c;
        sum += d;
        return sum;
    }
```

```
+= : Add and assignment operator. It adds the right operand to the left operand and assigns the

a += b is equivalent to a = a + b;
```

Task

Write a function `int max_of_four(int a, int b, int c, int d)` which reads four arguments and returns the greatest of

Task

Write a function `int max_of_four(int a, int b, int c, int d)` which reads four arguments and returns the greatest of them.

Note

There is not built in `max` function in C. Code that will be reused is often put in a separate function, e.g. `int max(x, y)` that returns the greater of the two values.

Input Format

Input will contain four integers - a, b, c, d , one on each line.

Output Format

Print the greatest of the four integers.

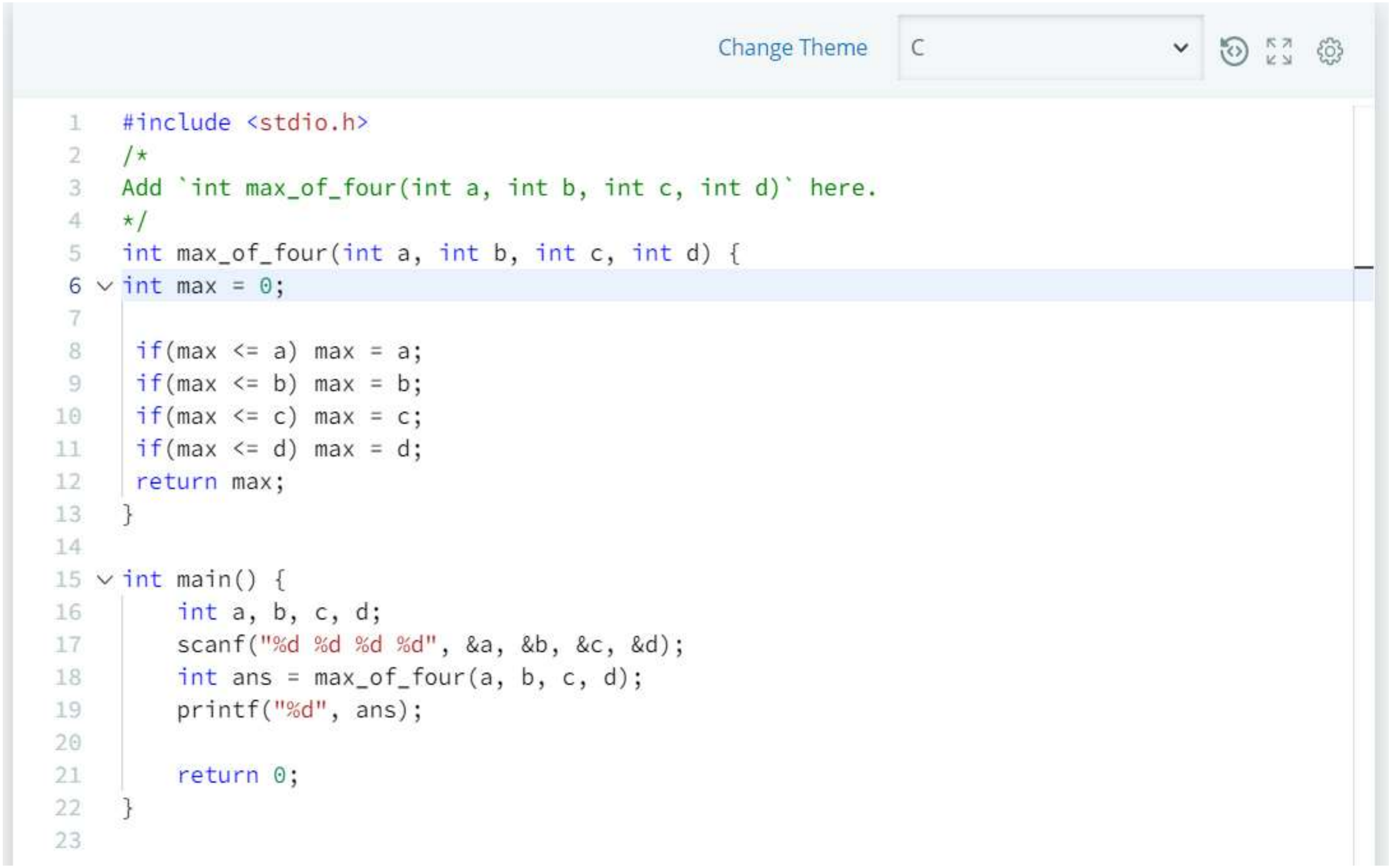
Note: I/O will be automatically handled.

Sample Input

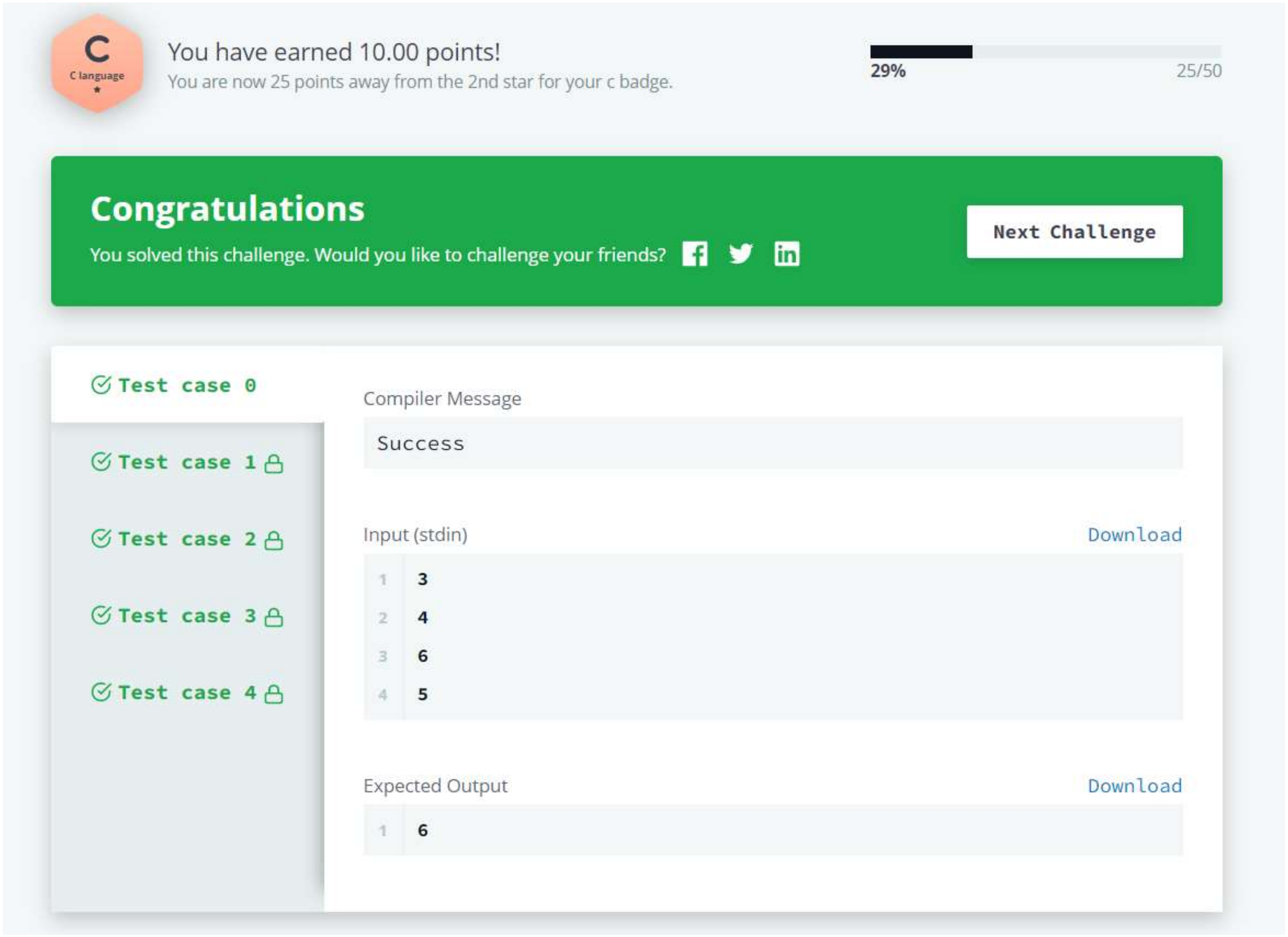
```
3
4
6
5
```

Sample Output

```
6
```



```c
#include <stdio.h>
/*
Add `int max_of_four(int a, int b, int c, int d)` here.
*/
int max_of_four(int a, int b, int c, int d) {
int max = 0;

 if(max <= a) max = a;
 if(max <= b) max = b;
 if(max <= c) max = c;
 if(max <= d) max = d;
 return max;
}

int main() {
    int a, b, c, d;
    scanf("%d %d %d %d", &a, &b, &c, &d);
    int ans = max_of_four(a, b, c, d);
    printf("%d", ans);

    return 0;
}

```



C
C language
★

You have earned 10.00 points!

You are now 25 points away from the 2nd star for your c badge.

29% 25/50

Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

Compiler Message

```
Success
```

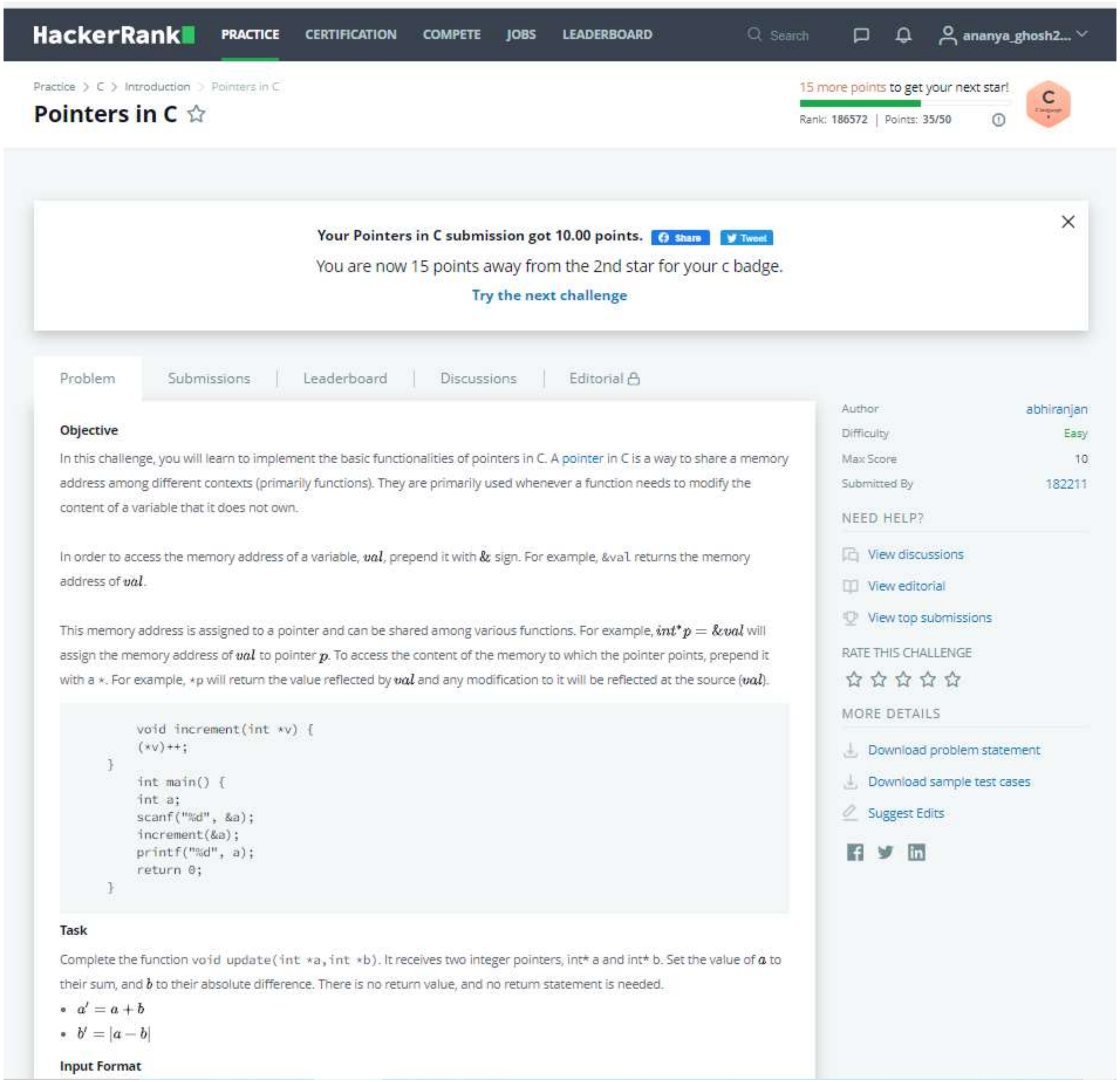
Input (stdin) Download

```
3
4
6
5
```

Expected Output Download

```
6
```

5.



Pointers in C

15 more points to get your next star!
Rank: 186572 | Points: 35/50

Your Pointers in C submission got 10.00 points.

You are now 15 points away from the 2nd star for your c badge.

Try the next challenge

Objective

In this challenge, you will learn to implement the basic functionalities of pointers in C. A pointer in C is a way to share a memory address among different contexts (primarily functions). They are primarily used whenever a function needs to modify the content of a variable that it does not own.

In order to access the memory address of a variable, val, prepend it with $\&$ sign. For example, &val returns the memory address of val.

This memory address is assigned to a pointer and can be shared among various functions. For example, $int^*p = \&val$ will assign the memory address of val to pointer p. To access the content of the memory to which the pointer points, prepend it with a *. For example, *p will return the value reflected by val and any modification to it will be reflected at the source (val).

```
        void increment(int *v) {
        (*v)++;
    }

        int main() {
        int a;
        scanf("%d", &a);
        increment(&a);
        printf("%d", a);
        return 0;
    }
```

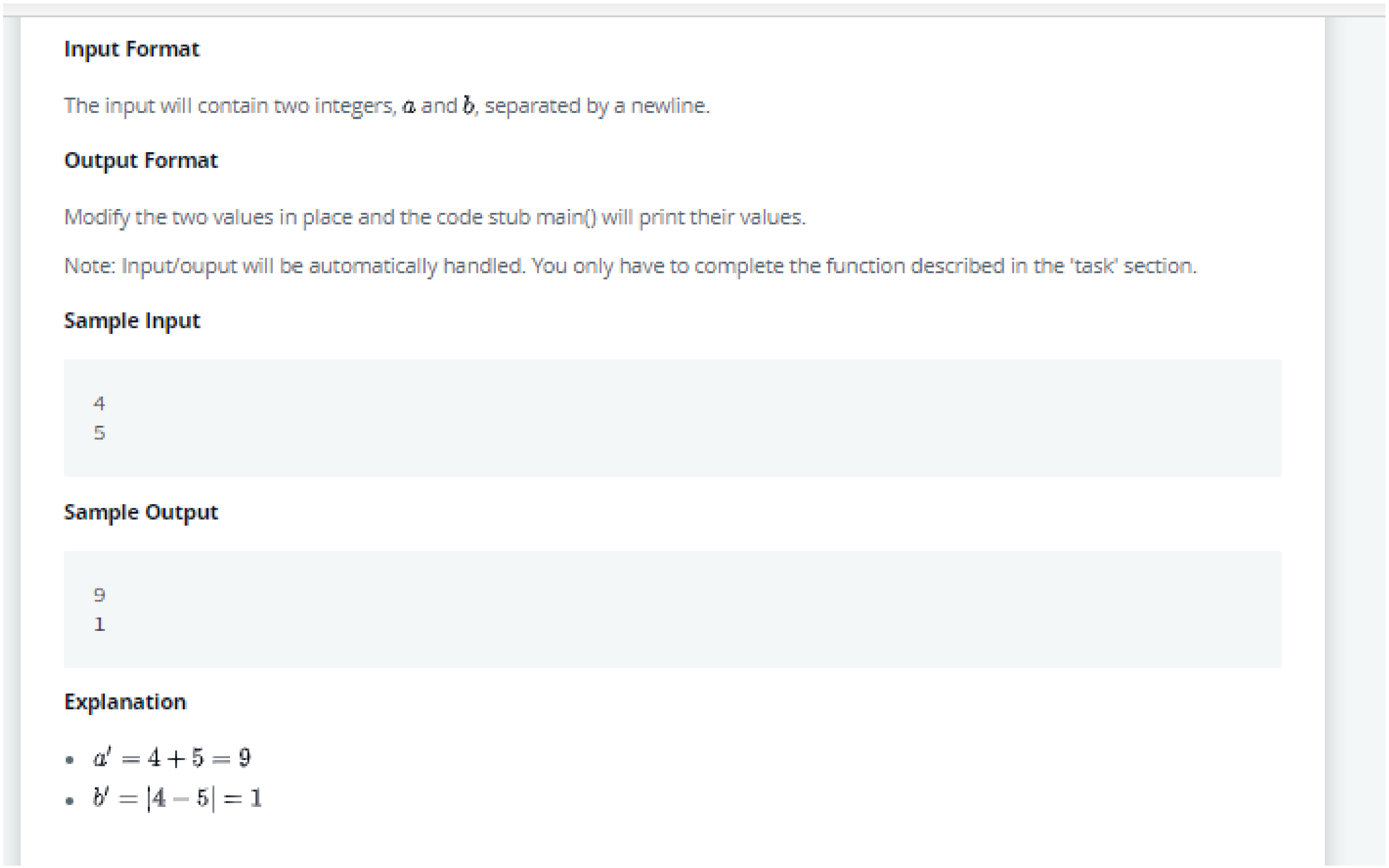
Task

Complete the function void update(int *a,int *b). It receives two integer pointers, int* a and int* b. Set the value of a to their sum, and b to their absolute difference. There is no return value, and no return statement is needed.

- $a' = a + b$
- $b' = |a - b|$

Input Format

Author	abhiranjan
Difficulty	Easy
Max Score	10
Submitted By	182211



Input Format

The input will contain two integers, a and b, separated by a newline.

Output Format

Modify the two values in place and the code stub main() will print their values.

Note: Input/ouput will be automatically handled. You only have to complete the function described in the 'task' section.

Sample Input

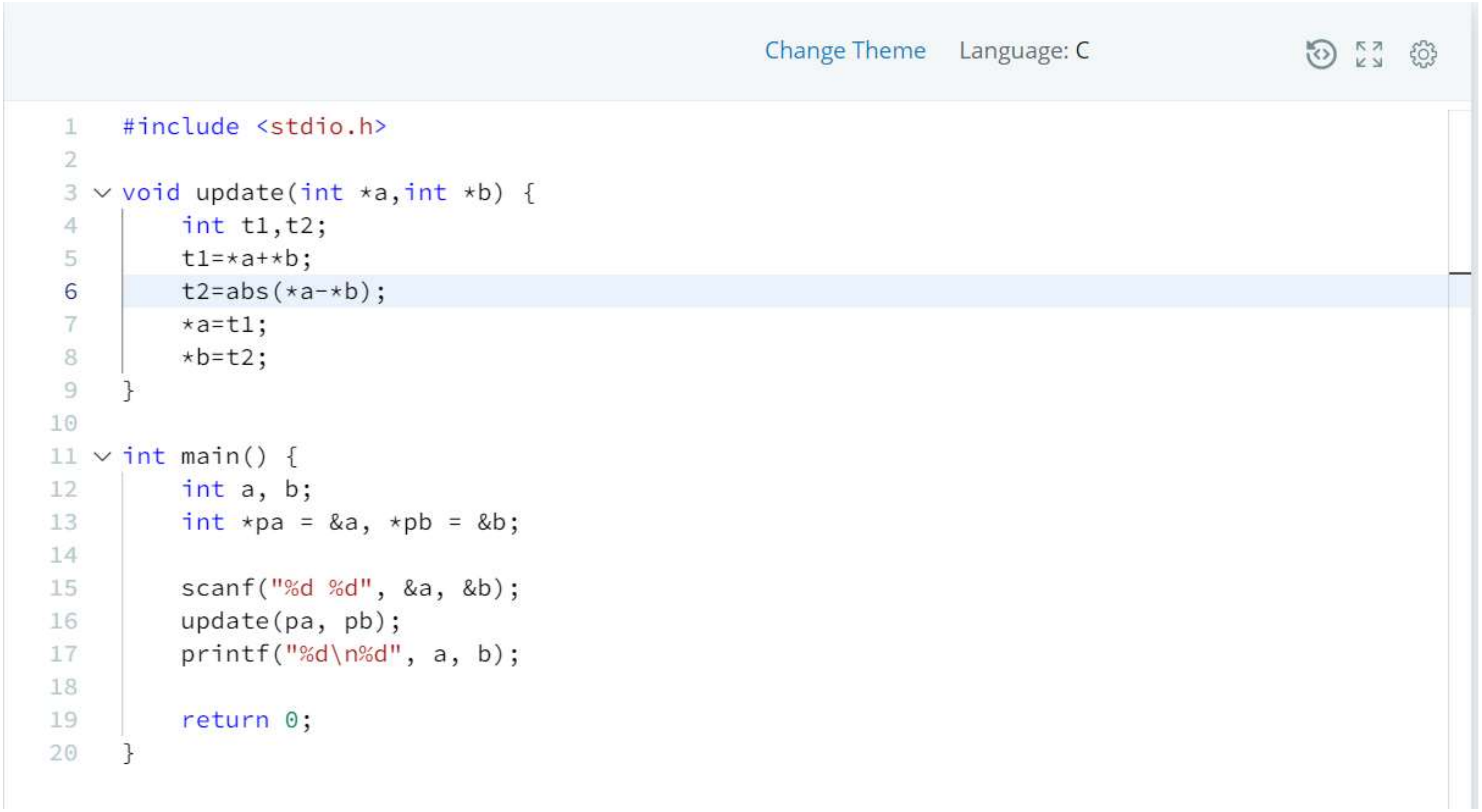
```
4
5
```

Sample Output

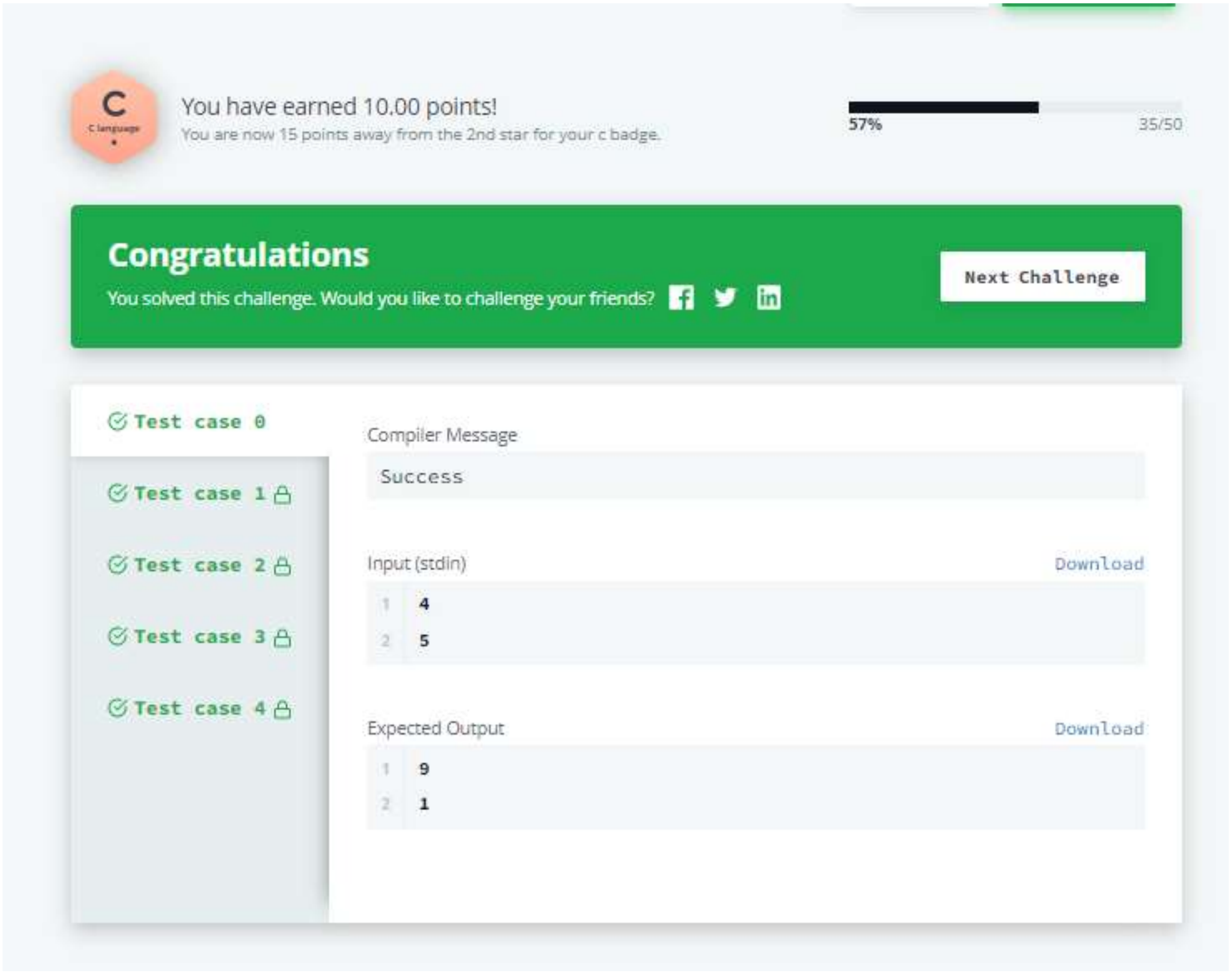
```
9
1
```

Explanation

- $a' = 4 + 5 = 9$
- $b' = |4 - 5| = 1$



```c
#include <stdio.h>

void update(int *a,int *b) {
    int t1,t2;
    t1=*a+*b;
    t2=abs(*a-*b);
    *a=t1;
    *b=t2;
}

int main() {
    int a, b;
    int *pa = &a, *pb = &b;

    scanf("%d %d", &a, &b);
    update(pa, pb);
    printf("%d\n%d", a, b);

    return 0;
}
```



You have earned 10.00 points!

You are now 15 points away from the 2nd star for your c badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

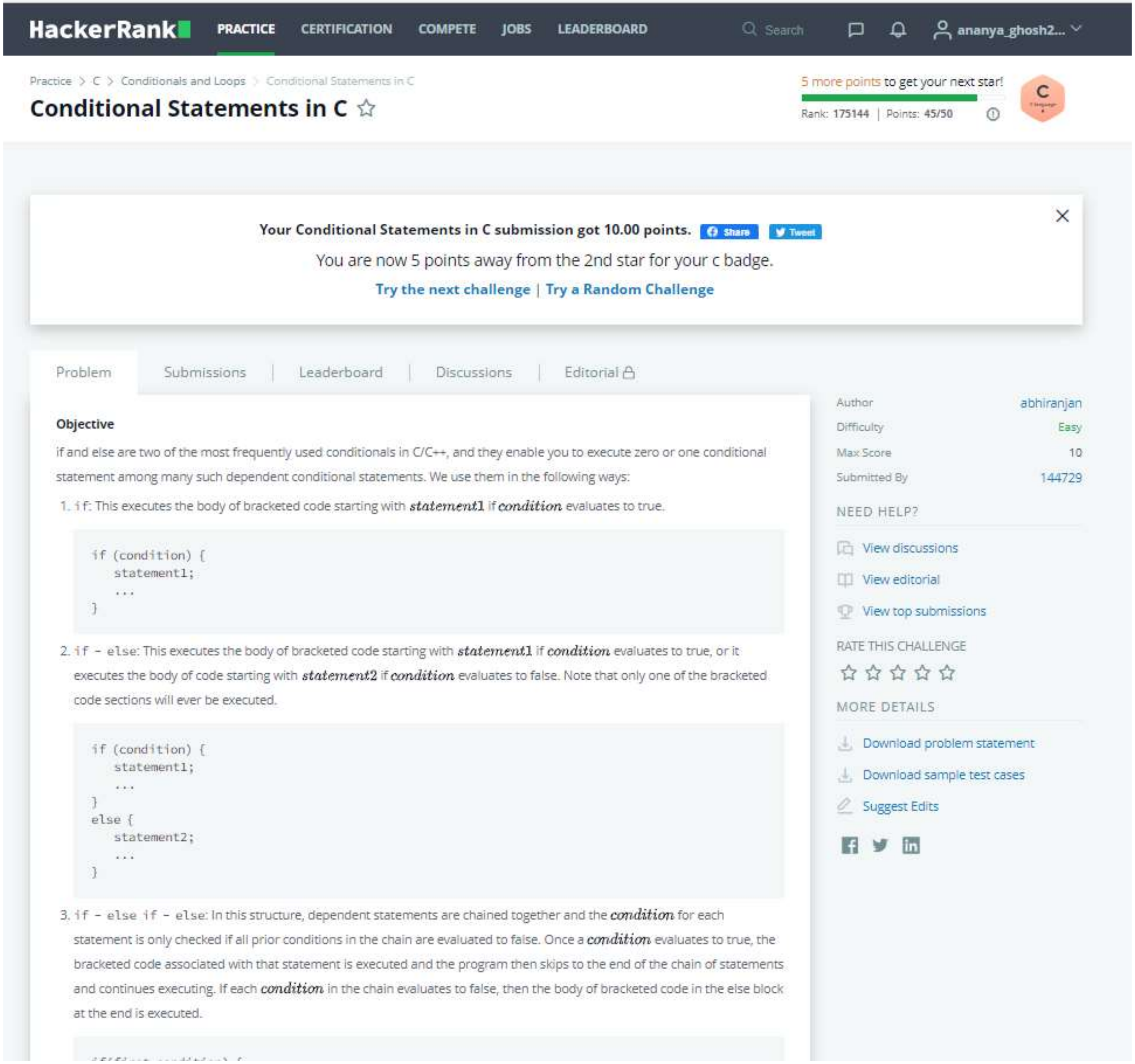
Input (stdin)

```
4
5
```

Expected Output

```
9
1
```

6.



HackerRank PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search ananya_ghosh2...

Practice > C > Conditionals and Loops > Conditional Statements in C

Conditional Statements in C

5 more points to get your next star!

Rank: 175144 | Points: 45/50

Your Conditional Statements in C submission got 10.00 points. Share Tweet

You are now 5 points away from the 2nd star for your c badge.

Try the next challenge | Try a Random Challenge

Problem | Submissions | Leaderboard | Discussions | Editorial

Objective

if and else are two of the most frequently used conditionals in C/C++, and they enable you to execute zero or one conditional statement among many such dependent conditional statements. We use them in the following ways:

1. `if`: This executes the body of bracketed code starting with $statement1$ if $condition$ evaluates to true.

```
if (condition) {
    statement1;
    ...
}
```

2. `if - else`: This executes the body of bracketed code starting with $statement1$ if $condition$ evaluates to true, or it executes the body of code starting with $statement2$ if $condition$ evaluates to false. Note that only one of the bracketed code sections will ever be executed.

```
if (condition) {
    statement1;
    ...
}
else {
    statement2;
    ...
}
```

3. `if - else if - else`: In this structure, dependent statements are chained together and the $condition$ for each statement is only checked if all prior conditions in the chain are evaluated to false. Once a $condition$ evaluates to true, the bracketed code associated with that statement is executed and the program then skips to the end of the chain of statements and continues executing. If each $condition$ in the chain evaluates to false, then the body of bracketed code in the else block at the end is executed.

Author	abhiranjan
Difficulty	Easy
Max Score	10
Submitted By	144729

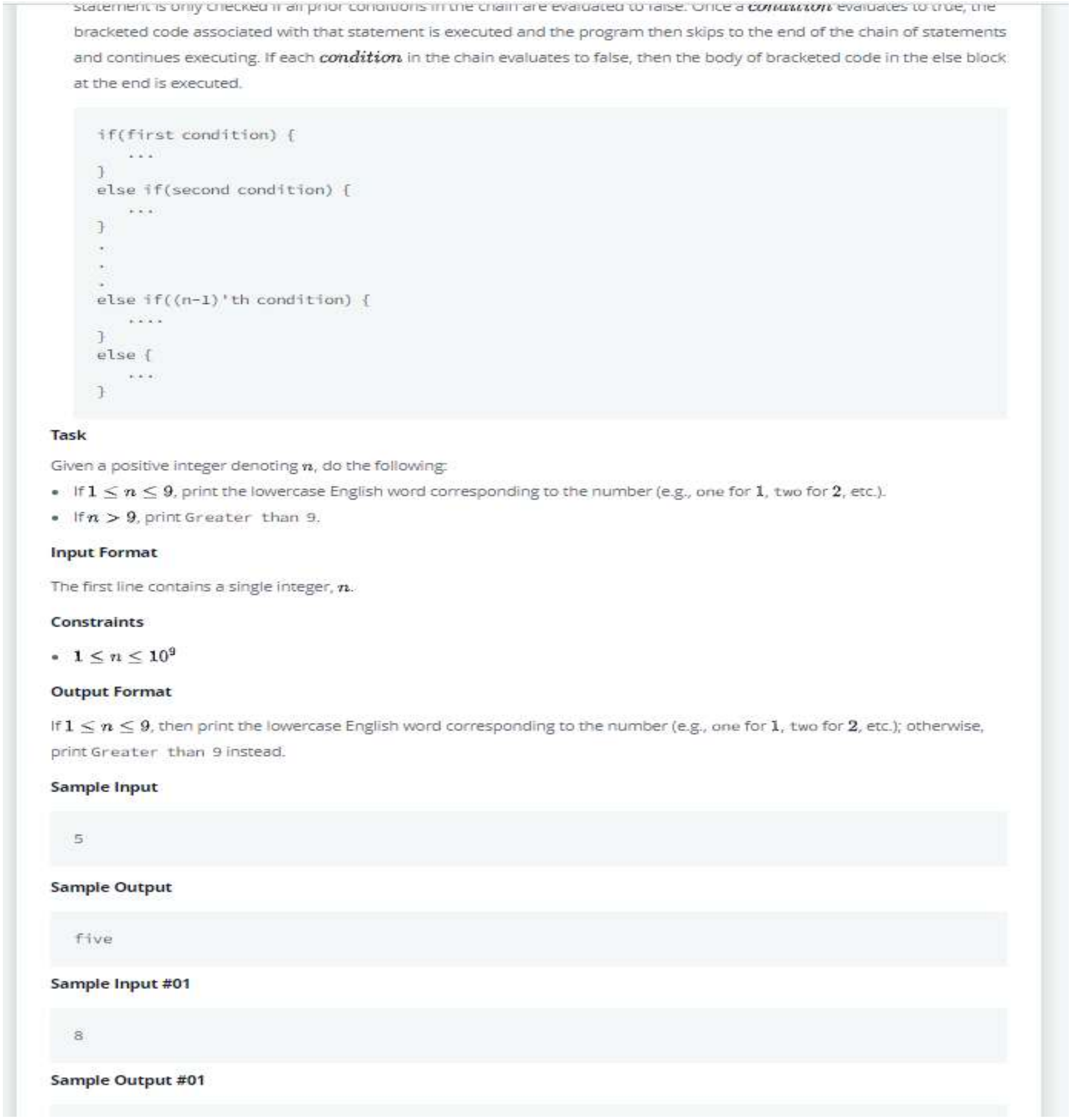
NEED HELP?

View discussions
View editorial
View top submissions

RATE THIS CHALLENGE

MORE DETAILS

Download problem statement
Download sample test cases
Suggest Edits



statement is only checked if all prior conditions in the chain are evaluated to false. Once a *condition* evaluates to true, the bracketed code associated with that statement is executed and the program then skips to the end of the chain of statements and continues executing. If each *condition* in the chain evaluates to false, then the body of bracketed code in the else block at the end is executed.

```
if(first condition) {
    ...
}
else if(second condition) {
    ...
}
.
.
.
else if((n-1)'th condition) {
    ....
}
else {
    ...
}
```

Task

Given a positive integer denoting n, do the following:

- If $1 \leq n \leq 9$, print the lowercase English word corresponding to the number (e.g., `one` for 1, `two` for 2, etc.).
- If $n > 9$, print `Greater than 9`.

Input Format

The first line contains a single integer, n.

Constraints

- $1 \leq n \leq 10^9$

Output Format

If $1 \leq n \leq 9$, then print the lowercase English word corresponding to the number (e.g., `one` for 1, `two` for 2, etc.); otherwise, print `Greater than 9` instead.

Sample Input

```
5
```

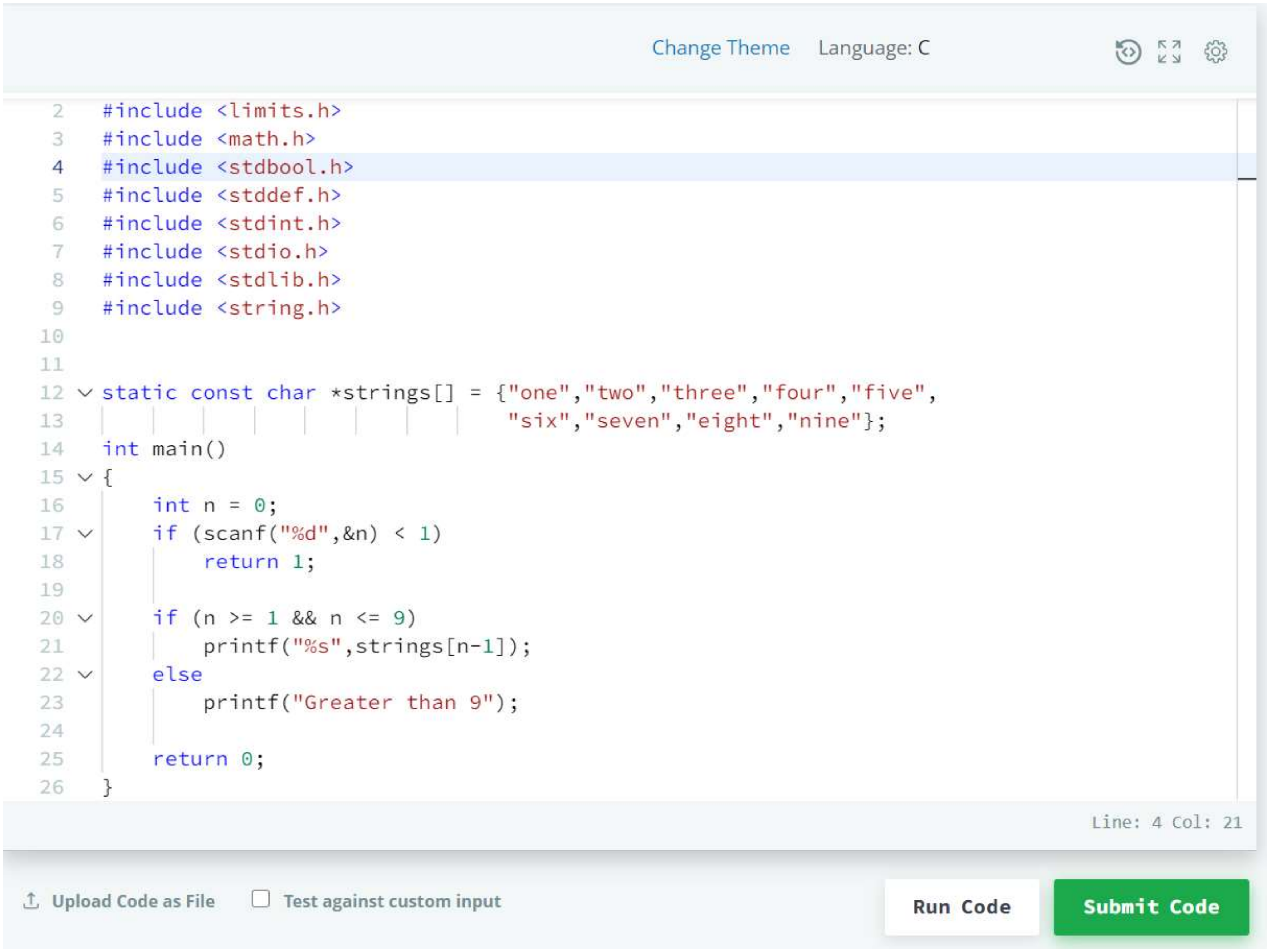
Sample Output

```
five
```

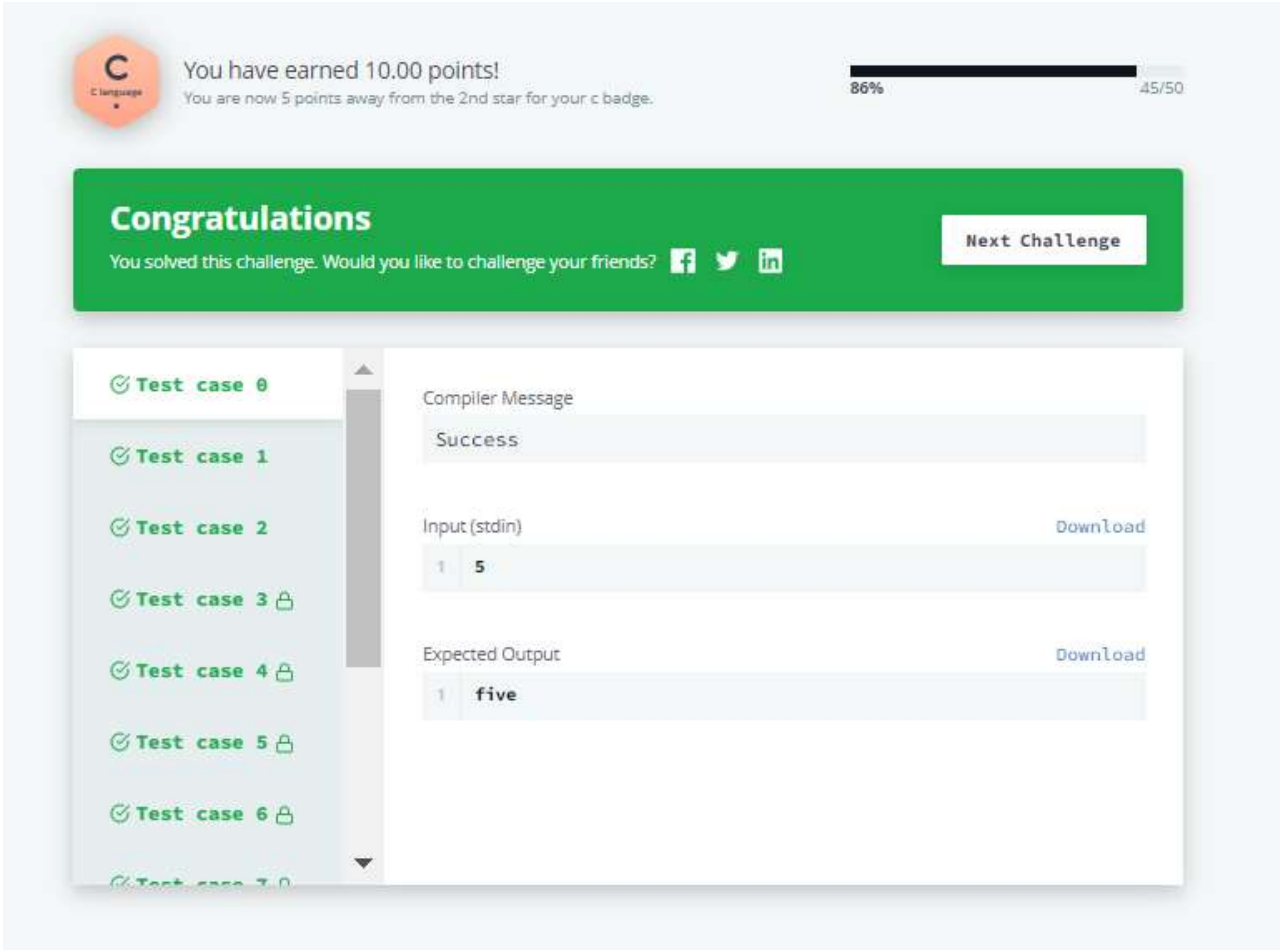
Sample Input #01

```
8
```

Sample Output #01



```c
#include <limits.h>
#include <math.h>
#include <stdbool.h>
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


static const char *strings[] = {"one","two","three","four","five",
                                "six","seven","eight","nine"};
int main()
{
    int n = 0;
    if (scanf("%d",&n) < 1)
        return 1;

    if (n >= 1 && n <= 9)
        printf("%s",strings[n-1]);
    else
        printf("Greater than 9");

    return 0;
}
```



You have earned 10.00 points!

You are now 5 points away from the 2nd star for your c badge.

86% 45/50

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

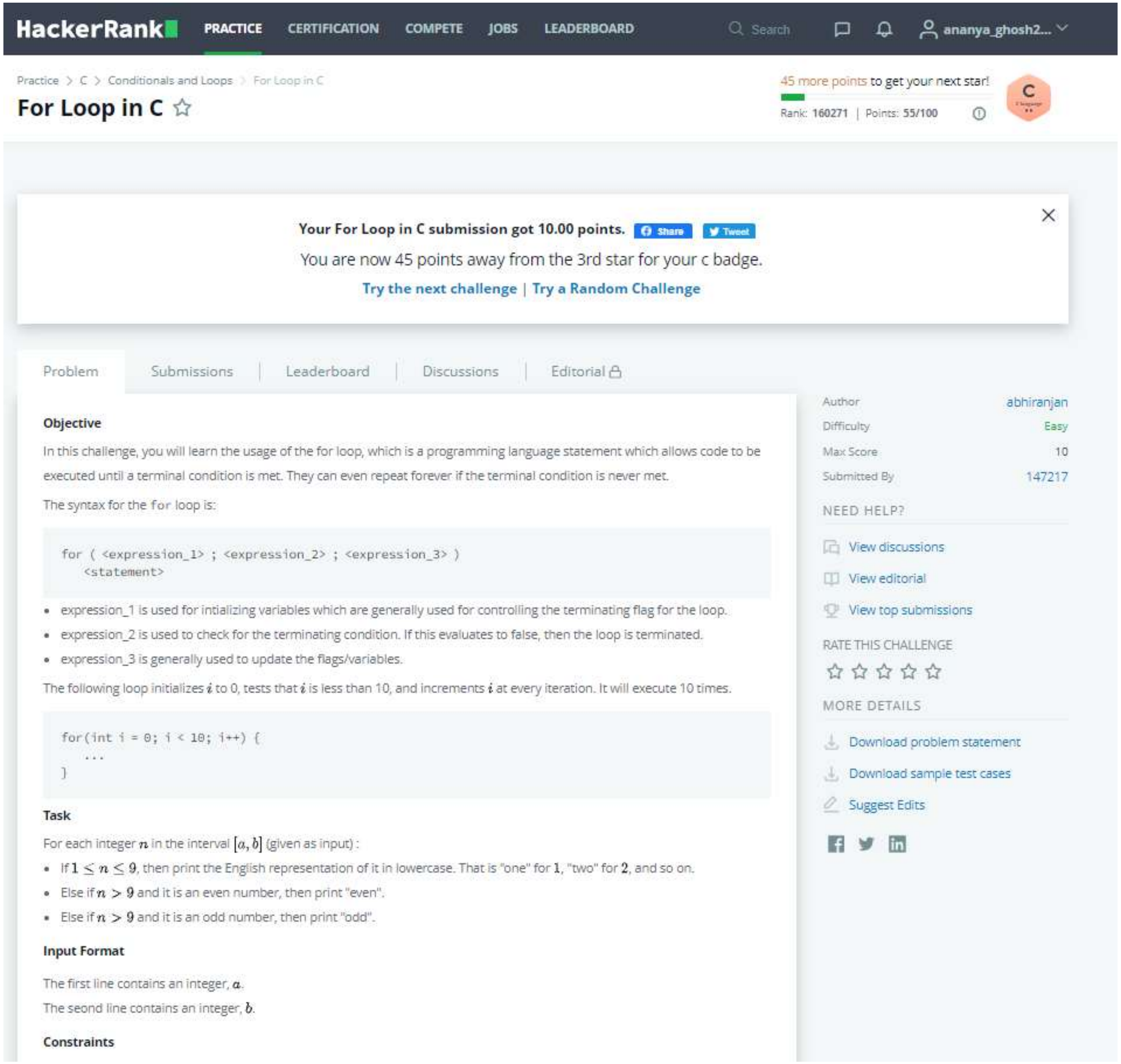
Input (stdin)

```
5
```

Expected Output

```
five
```

7.



HackerRank PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search ananya_ghosh2...

Practice > C > Conditionals and Loops > For Loop in C

For Loop in C

45 more points to get your next star!
Rank: 160271 | Points: 55/100

Your For Loop in C submission got 10.00 points. Share Tweet

You are now 45 points away from the 3rd star for your c badge.

Try the next challenge | Try a Random Challenge

Problem | Submissions | Leaderboard | Discussions | Editorial

Objective

In this challenge, you will learn the usage of the for loop, which is a programming language statement which allows code to be executed until a terminal condition is met. They can even repeat forever if the terminal condition is never met.

The syntax for the `for` loop is:

```
for ( <expression_1> ; <expression_2> ; <expression_3> )
    <statement>
```

- expression_1 is used for intializing variables which are generally used for controlling the terminating flag for the loop.
- expression_2 is used to check for the terminating condition. If this evaluates to false, then the loop is terminated.
- expression_3 is generally used to update the flags/variables.

The following loop initializes i to 0, tests that i is less than 10, and increments i at every iteration. It will execute 10 times.

```
for(int i = 0; i < 10; i++) {
    ...
}
```

Task

For each integer n in the interval $[a, b]$ (given as input) :

- If $1 \leq n \leq 9$, then print the English representation of it in lowercase. That is "one" for 1, "two" for 2, and so on.
- Else if $n > 9$ and it is an even number, then print "even".
- Else if $n > 9$ and it is an odd number, then print "odd".

Input Format

The first line contains an integer, a.

The seond line contains an integer, b.

Constraints

Author	abhiranjan
Difficulty	Easy
Max Score	10
Submitted By	147217

NEED HELP?

View discussions

View editorial

View top submissions

RATE THIS CHALLENGE

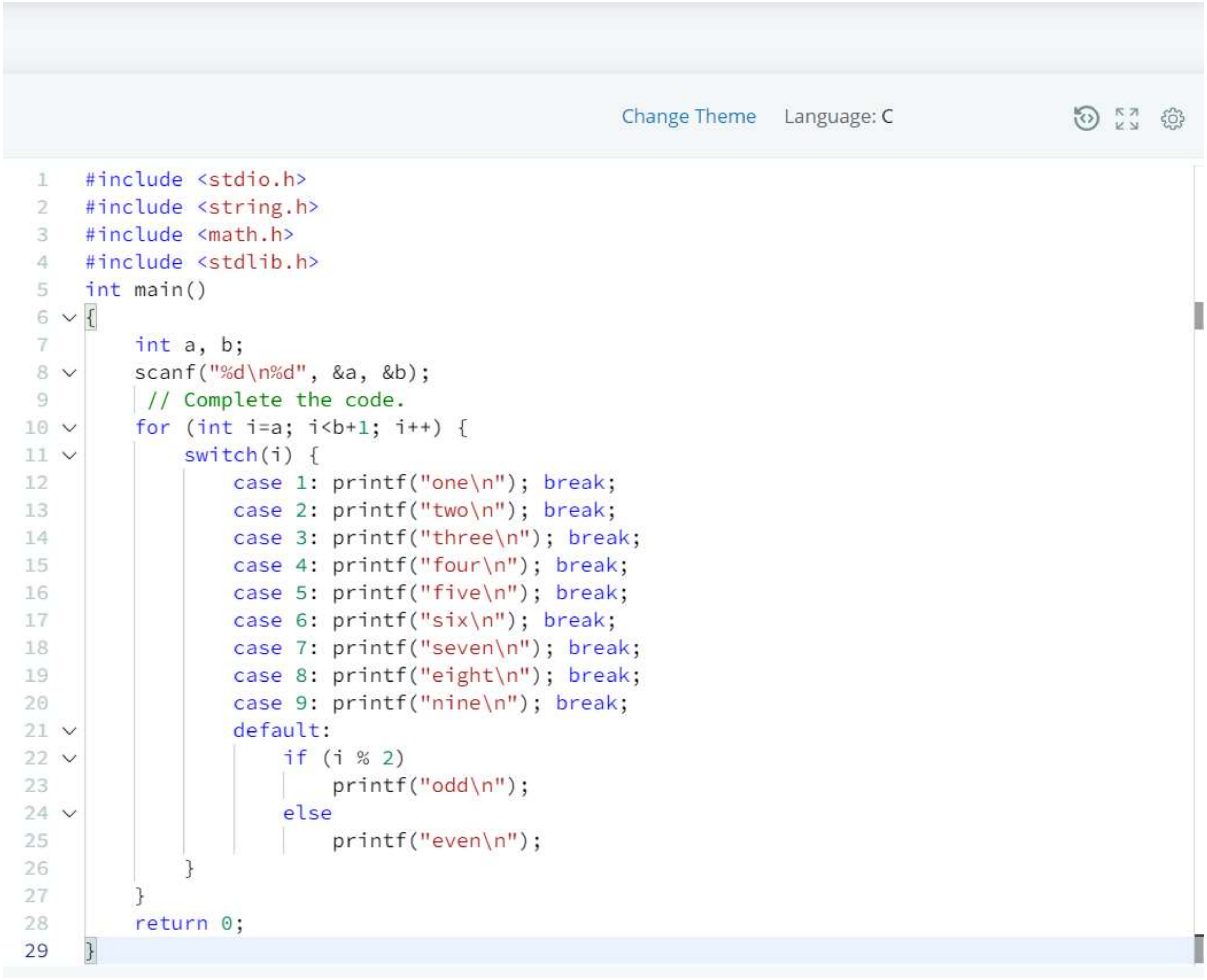
☆ ☆ ☆ ☆ ☆

MORE DETAILS

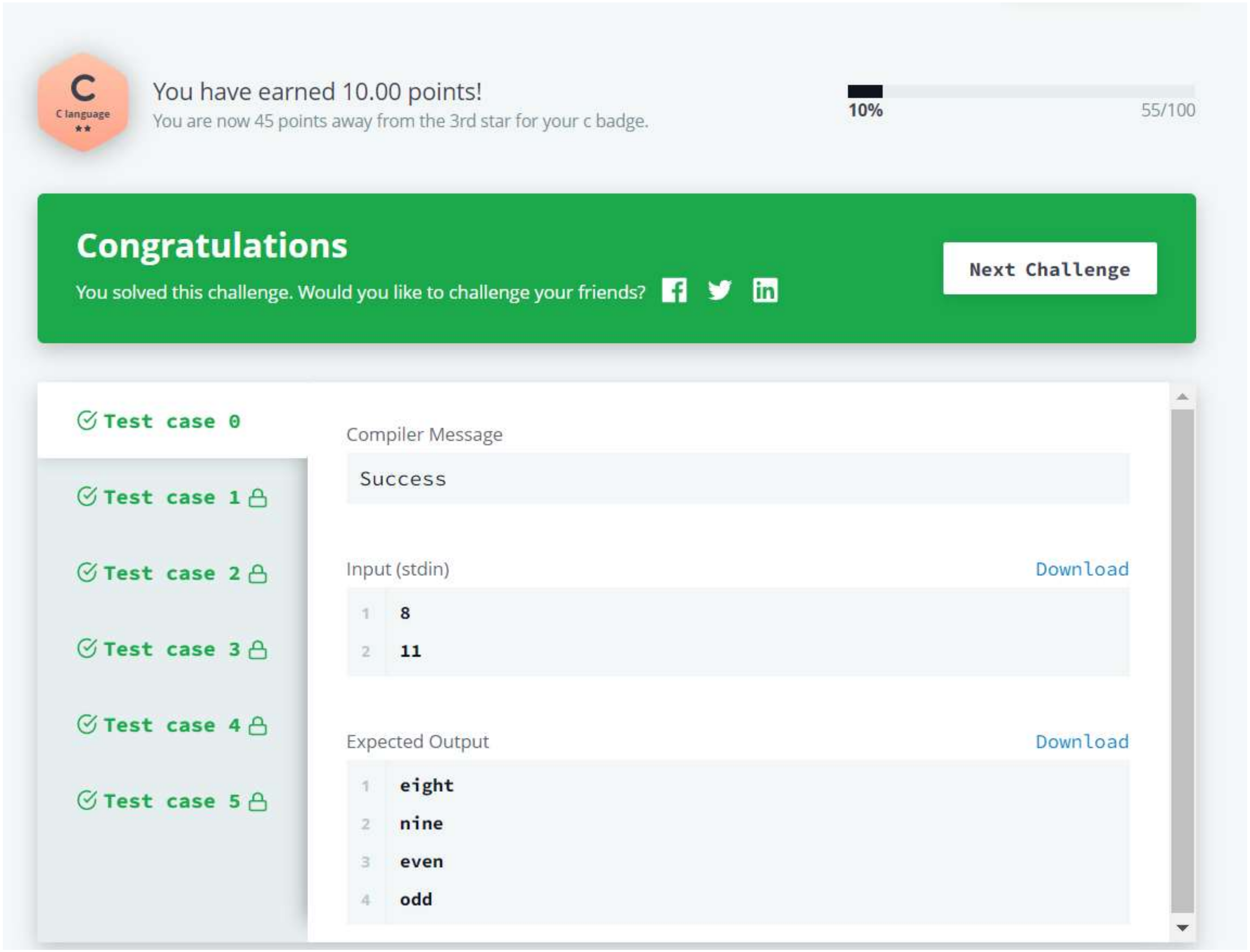
Download problem statement

Download sample test cases

Suggest Edits



```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
int main()
{
    int a, b;
    scanf("%d\n%d", &a, &b);
     // Complete the code.
    for (int i=a; i<b+1; i++) {
        switch(i) {
            case 1: printf("one\n"); break;
            case 2: printf("two\n"); break;
            case 3: printf("three\n"); break;
            case 4: printf("four\n"); break;
            case 5: printf("five\n"); break;
            case 6: printf("six\n"); break;
            case 7: printf("seven\n"); break;
            case 8: printf("eight\n"); break;
            case 9: printf("nine\n"); break;
            default:
                if (i % 2)
                    printf("odd\n");
                else
                    printf("even\n");
        }
    }
    return 0;
}
```



You have earned 10.00 points!

You are now 45 points away from the 3rd star for your c badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

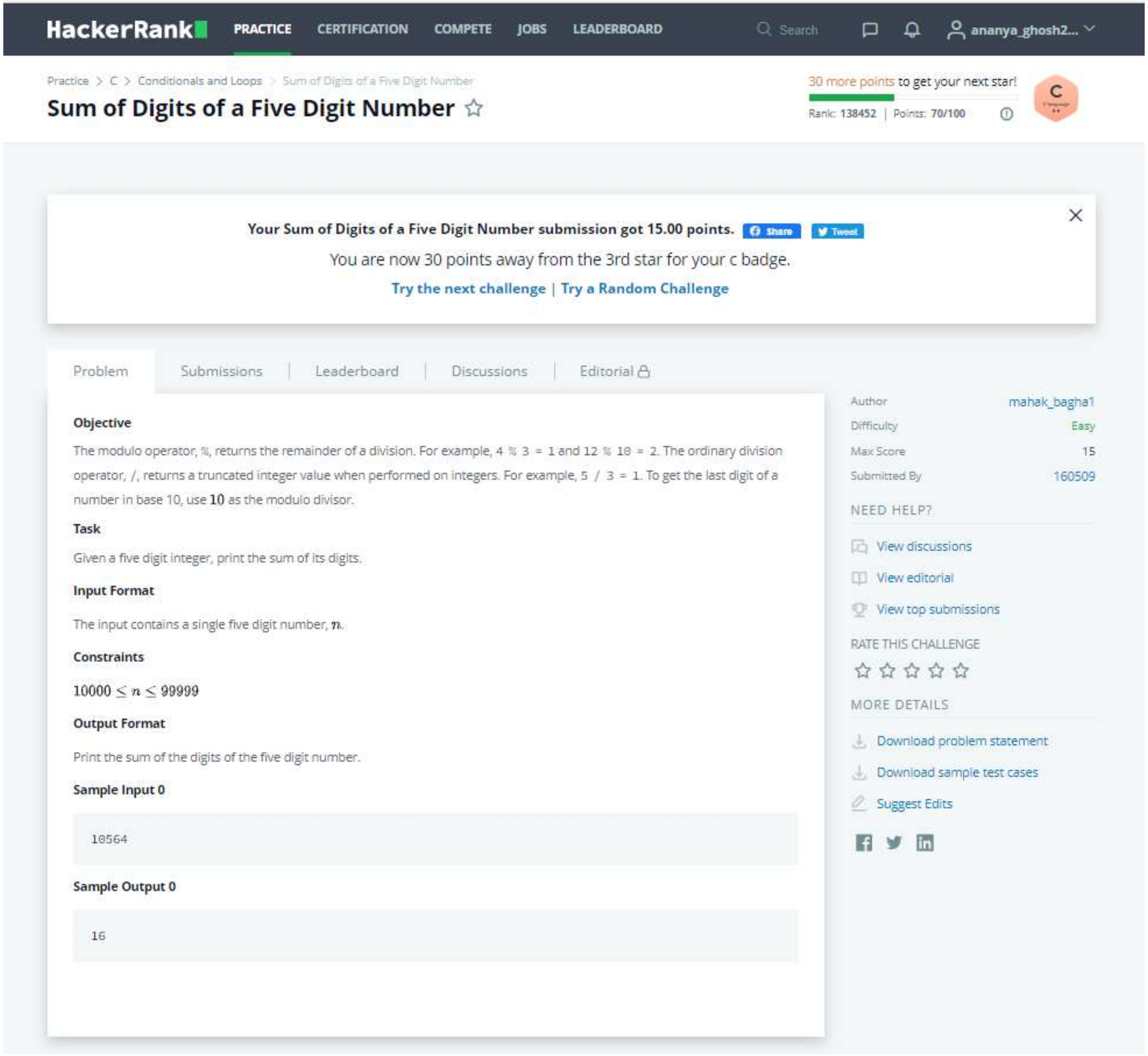
Input (stdin)

```
8
11
```

Expected Output

```
eight
nine
even
odd
```

8.



HackerRank PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search ananya_ghosh2...

Practice > C > Conditionals and Loops > Sum of Digits of a Five Digit Number

Sum of Digits of a Five Digit Number

30 more points to get your next star!
Rank: 138452 | Points: 70/100

Your Sum of Digits of a Five Digit Number submission got 15.00 points. Share Tweet

You are now 30 points away from the 3rd star for your c badge.

Try the next challenge | Try a Random Challenge

Problem | Submissions | Leaderboard | Discussions | Editorial

Objective

The modulo operator, `%`, returns the remainder of a division. For example, `4 % 3 = 1` and `12 % 10 = 2`. The ordinary division operator, `/`, returns a truncated integer value when performed on integers. For example, `5 / 3 = 1`. To get the last digit of a number in base 10, use 10 as the modulo divisor.

Task

Given a five digit integer, print the sum of its digits.

Input Format

The input contains a single five digit number, n.

Constraints

$10000 \le n \le 99999$

Output Format

Print the sum of the digits of the five digit number.

Sample Input 0

```
10564
```

Sample Output 0

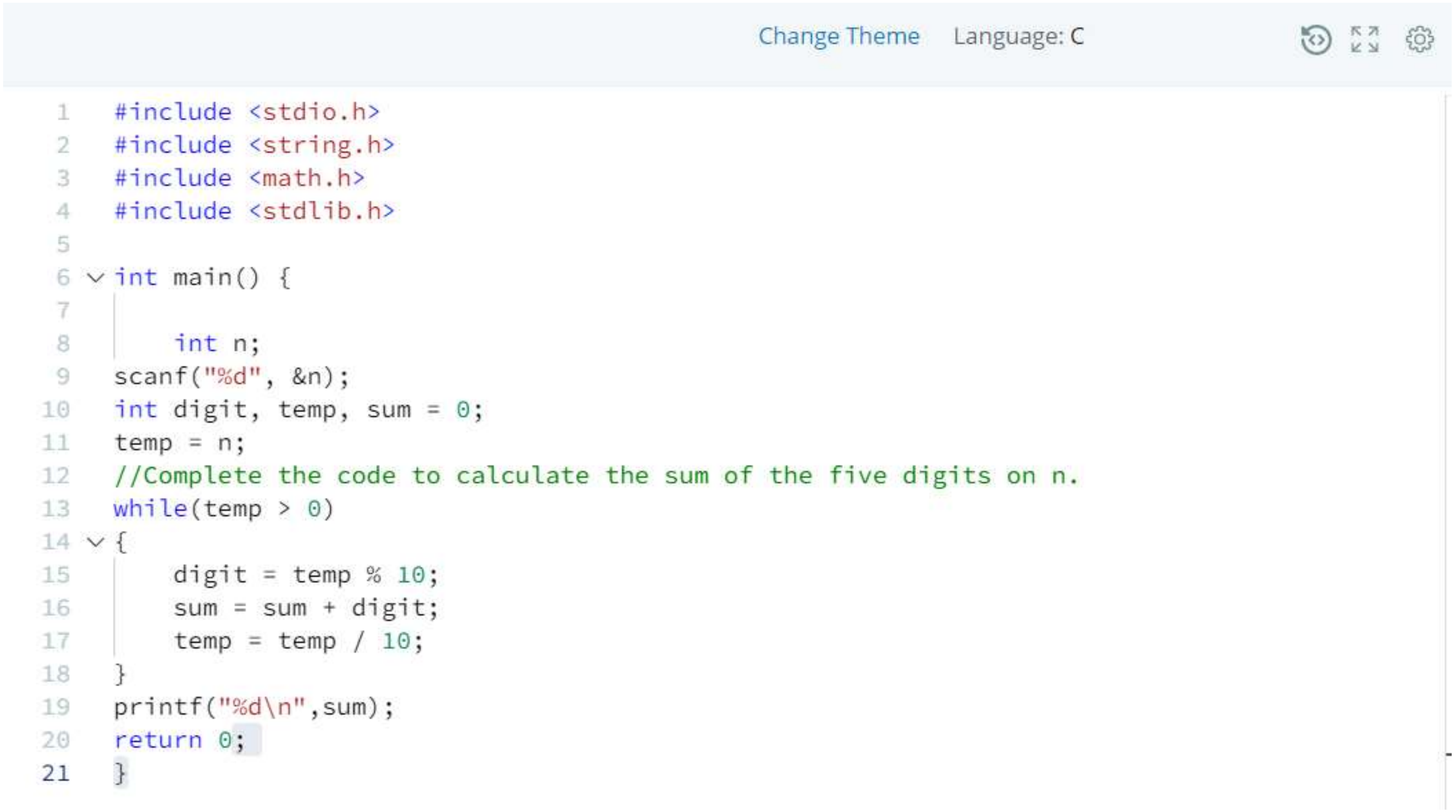
```
16
```

Author	mahak_bagha1
Difficulty	Easy
Max Score	15
Submitted By	160509

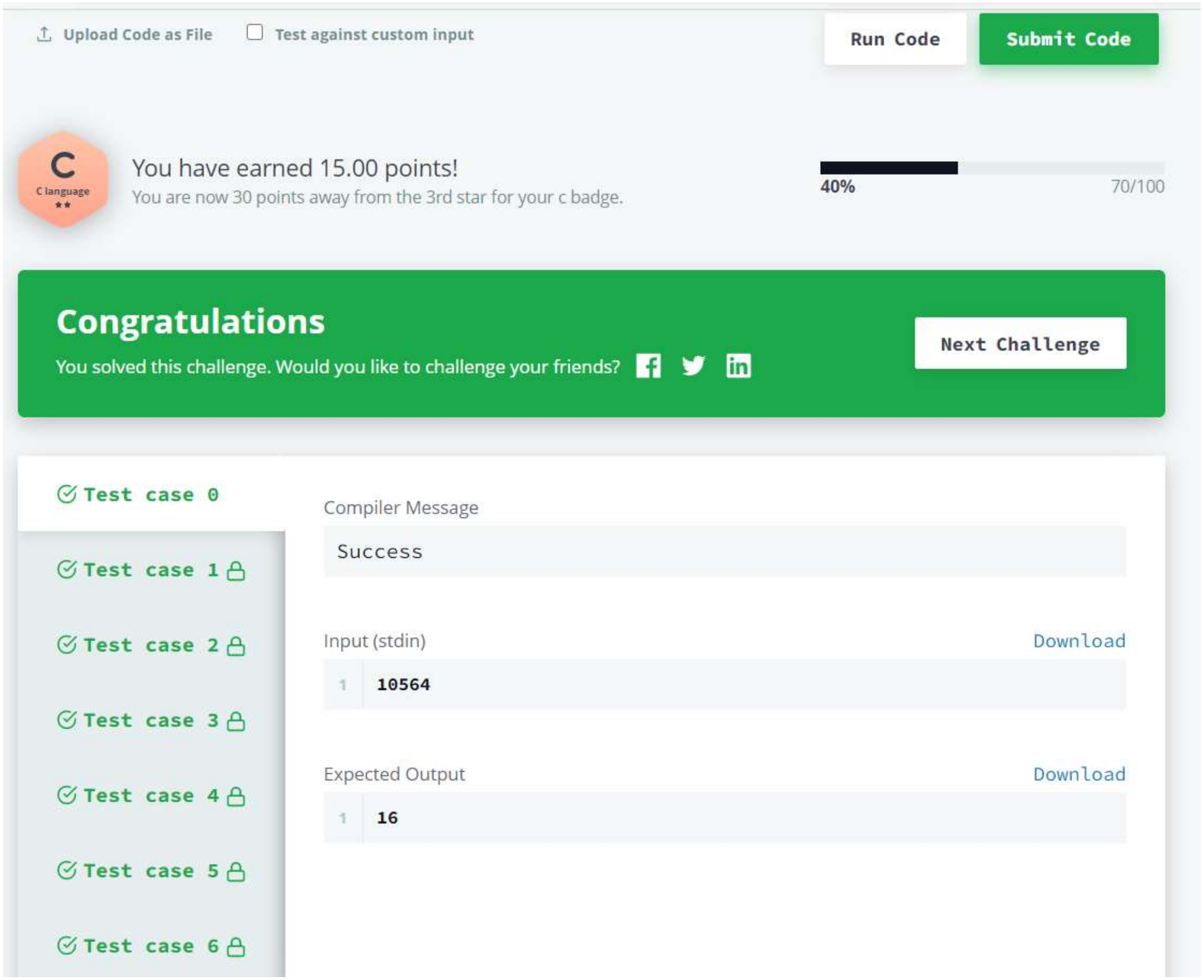
NEED HELP?
View discussions
View editorial
View top submissions

RATE THIS CHALLENGE

MORE DETAILS
Download problem statement
Download sample test cases
Suggest Edits



```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

int main() {

    int n;
scanf("%d", &n);
int digit, temp, sum = 0;
temp = n;
//Complete the code to calculate the sum of the five digits on n.
while(temp > 0)
{
    digit = temp % 10;
    sum = sum + digit;
    temp = temp / 10;
}
printf("%d\n",sum);
return 0;
}
```



You have earned 15.00 points!

You are now 30 points away from the 3rd star for your c badge.

40% 70/100

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

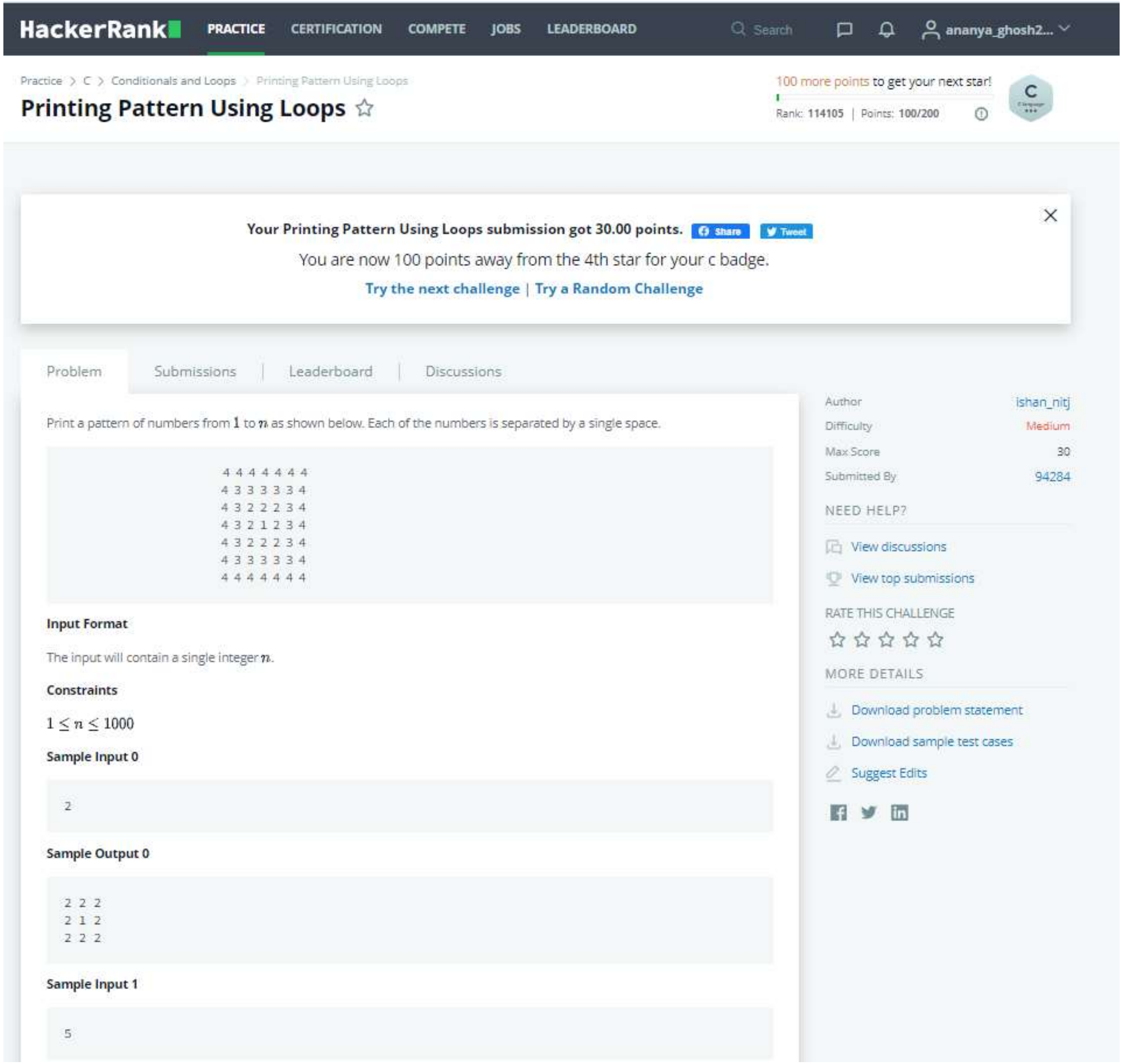
Input (stdin)

```
10564
```

Expected Output

```
16
```

9.



HackerRank PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search ananya_ghosh2...

Practice > C > Conditionals and Loops > Printing Pattern Using Loops

Printing Pattern Using Loops

100 more points to get your next star!
Rank: 114105 | Points: 100/200

Your Printing Pattern Using Loops submission got 30.00 points. Share Tweet

You are now 100 points away from the 4th star for your c badge.

Try the next challenge | Try a Random Challenge

Problem | Submissions | Leaderboard | Discussions

Print a pattern of numbers from 1 to n as shown below. Each of the numbers is separated by a single space.

```
4 4 4 4 4 4 4
4 3 3 3 3 3 4
4 3 2 2 2 3 4
4 3 2 1 2 3 4
4 3 2 2 2 3 4
4 3 3 3 3 3 4
4 4 4 4 4 4 4
```

Input Format

The input will contain a single integer n.

Constraints

$1 \leq n \leq 1000$

Sample Input 0

```
2
```

Sample Output 0

```
2 2 2
2 1 2
2 2 2
```

Sample Input 1

```
5
```

Author	ishan_nitj
Difficulty	Medium
Max Score	30
Submitted By	94284

NEED HELP?

View discussions

View top submissions

RATE THIS CHALLENGE

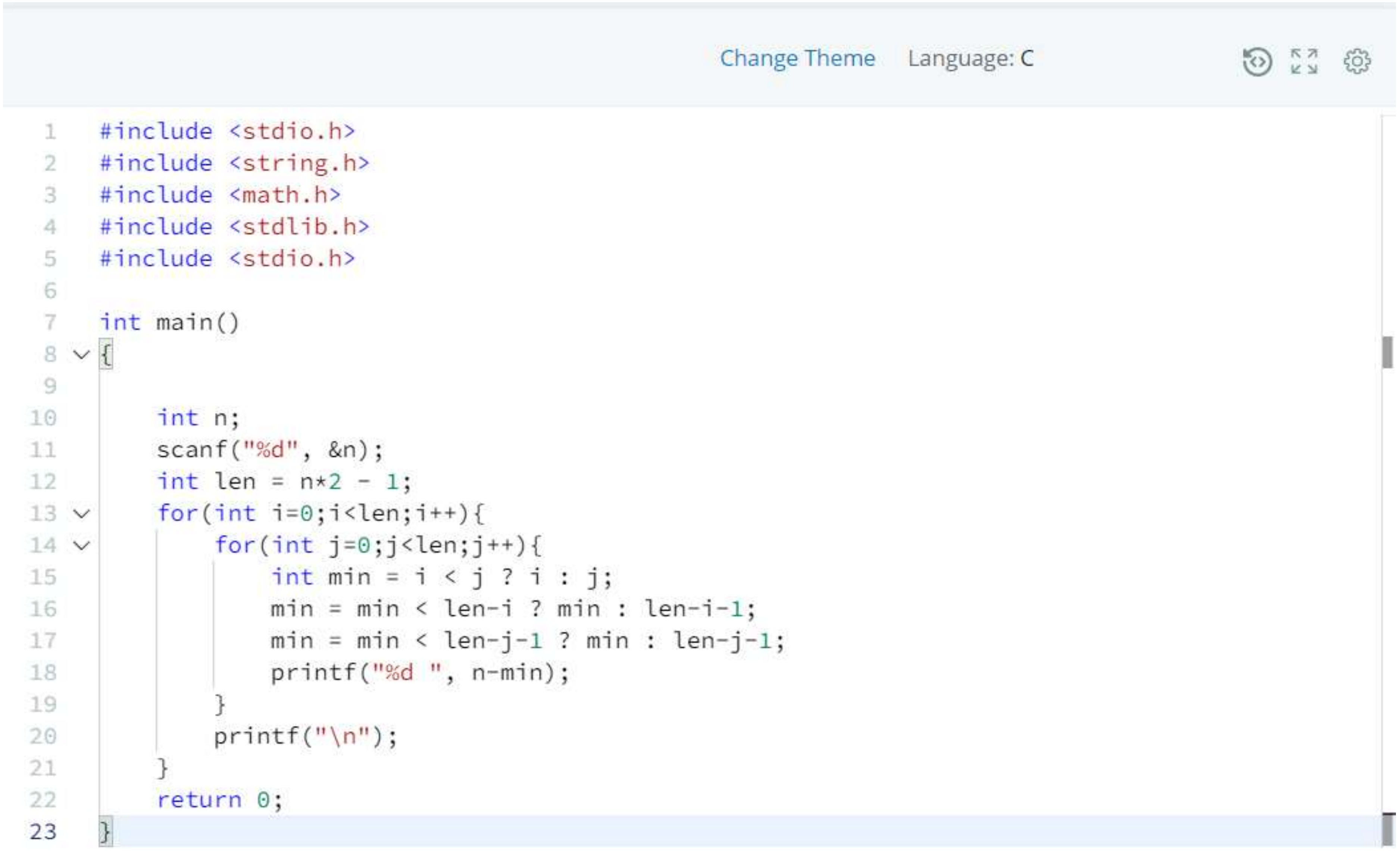
☆ ☆ ☆ ☆ ☆

MORE DETAILS

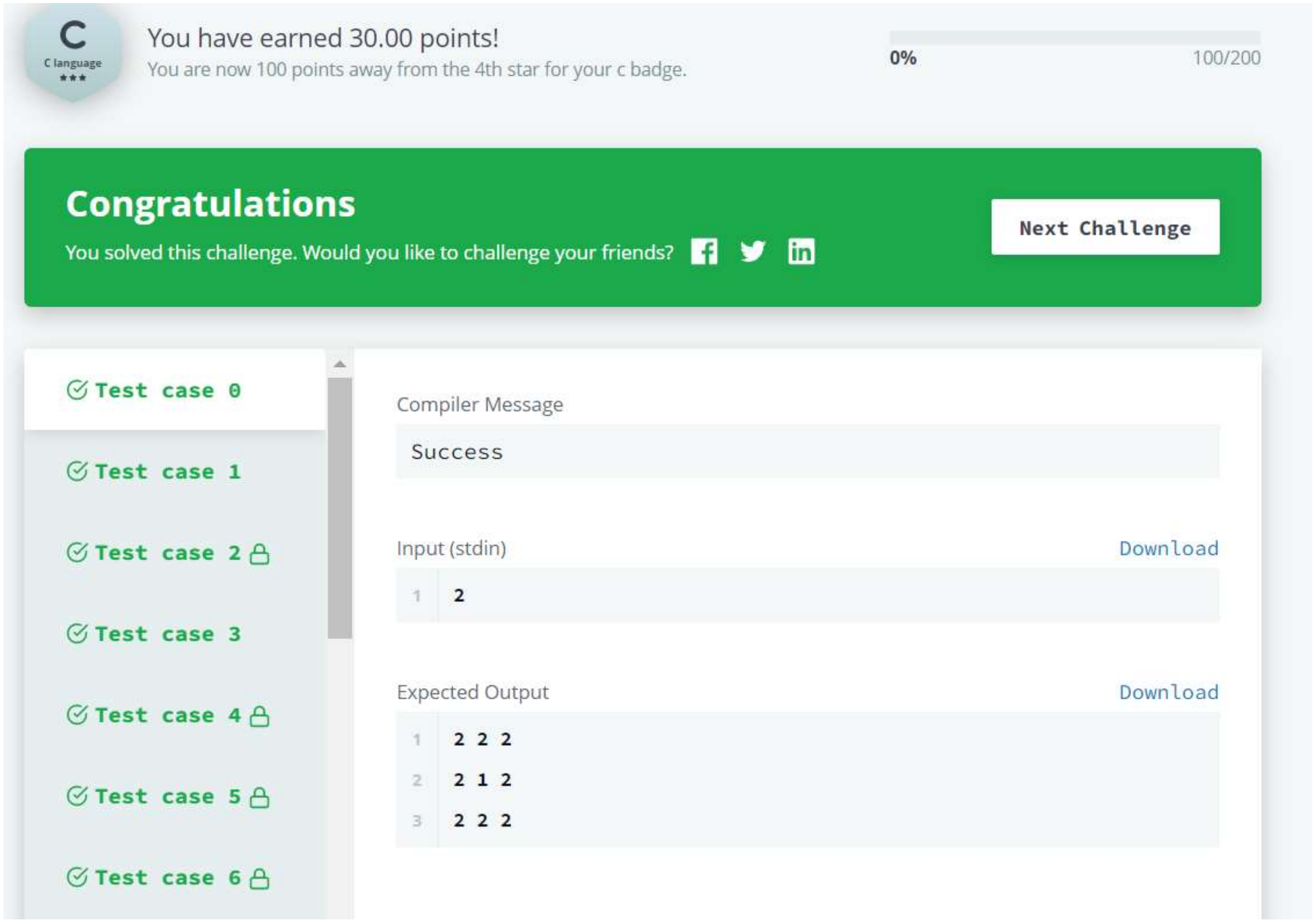
Download problem statement

Download sample test cases

Suggest Edits



```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
#include <stdio.h>

int main()
{


    int n;
    scanf("%d", &n);
    int len = n*2 - 1;
    for(int i=0;i<len;i++){
        for(int j=0;j<len;j++){
            int min = i < j ? i : j;
            min = min < len-i ? min : len-i-1;
            min = min < len-j-1 ? min : len-j-1;
            printf("%d ", n-min);
        }
        printf("\n");
    }
    return 0;
}
```



You have earned 30.00 points!

You are now 100 points away from the 4th star for your c badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

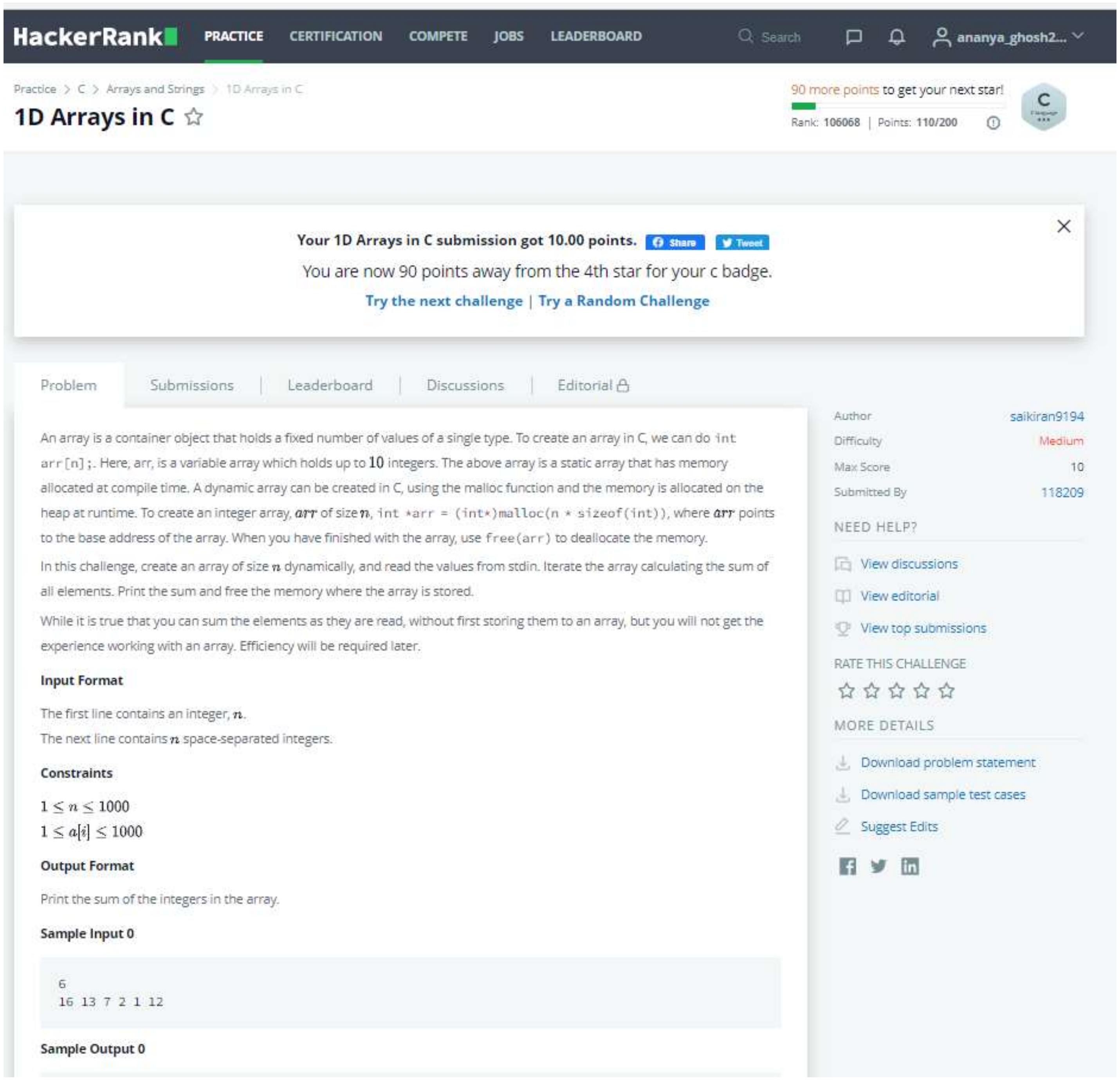
Input (stdin)

```
2
```

Expected Output

```
2 2 2
2 1 2
2 2 2
```

10.



Practice > C > Arrays and Strings > 1D Arrays in C

1D Arrays in C

90 more points to get your next star!

Rank: 106068 | Points: 110/200

Your 1D Arrays in C submission got 10.00 points. Share Tweet

You are now 90 points away from the 4th star for your c badge.

Try the next challenge | Try a Random Challenge

An array is a container object that holds a fixed number of values of a single type. To create an array in C, we can do `int arr[n];`. Here, arr, is a variable array which holds up to 10 integers. The above array is a static array that has memory allocated at compile time. A dynamic array can be created in C, using the malloc function and the memory is allocated on the heap at runtime. To create an integer array, arr of size n, `int *arr = (int*)malloc(n * sizeof(int))`, where arr points to the base address of the array. When you have finished with the array, use `free(arr)` to deallocate the memory.

In this challenge, create an array of size n dynamically, and read the values from stdin. Iterate the array calculating the sum of all elements. Print the sum and free the memory where the array is stored.

While it is true that you can sum the elements as they are read, without first storing them to an array, but you will not get the experience working with an array. Efficiency will be required later.

Input Format

The first line contains an integer, n.
The next line contains n space-separated integers.

Constraints

$1 \leq n \leq 1000$
$1 \leq a[i] \leq 1000$

Output Format

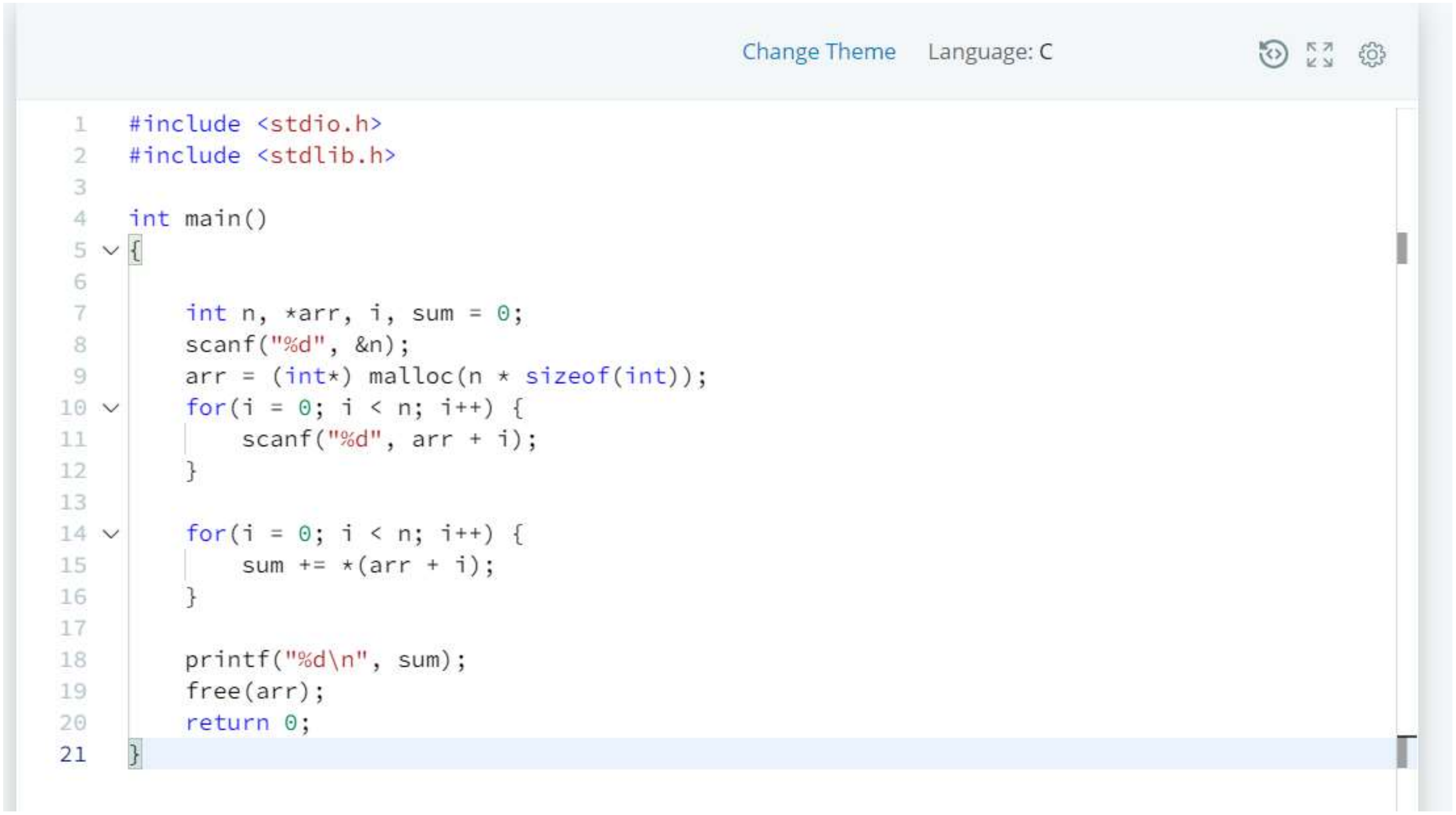
Print the sum of the integers in the array.

Sample Input 0

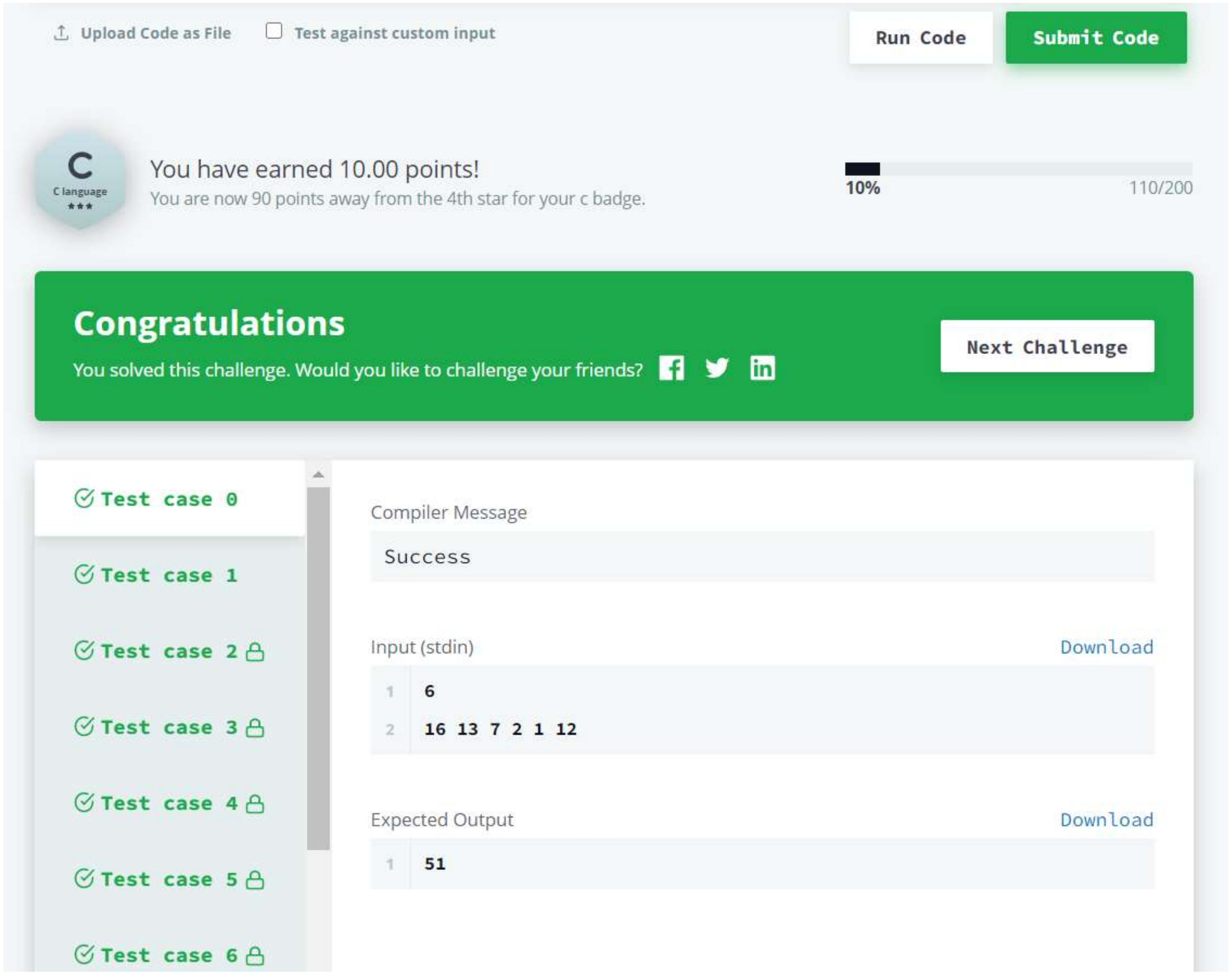
```
6
16 13 7 2 1 12
```

Sample Output 0

Author	saikiran9194
Difficulty	Medium
Max Score	10
Submitted By	118209



```c
#include <stdio.h>
#include <stdlib.h>

int main()
{

    int n, *arr, i, sum = 0;
    scanf("%d", &n);
    arr = (int*) malloc(n * sizeof(int));
    for(i = 0; i < n; i++) {
        scanf("%d", arr + i);
    }

    for(i = 0; i < n; i++) {
        sum += *(arr + i);
    }

    printf("%d\n", sum);
    free(arr);
    return 0;
}
```



You have earned 10.00 points!

You are now 90 points away from the 4th star for your c badge.

10%

110/200

Congratulations

You solved this challenge. Would you like to challenge your friends?

Compiler Message

```
Success
```

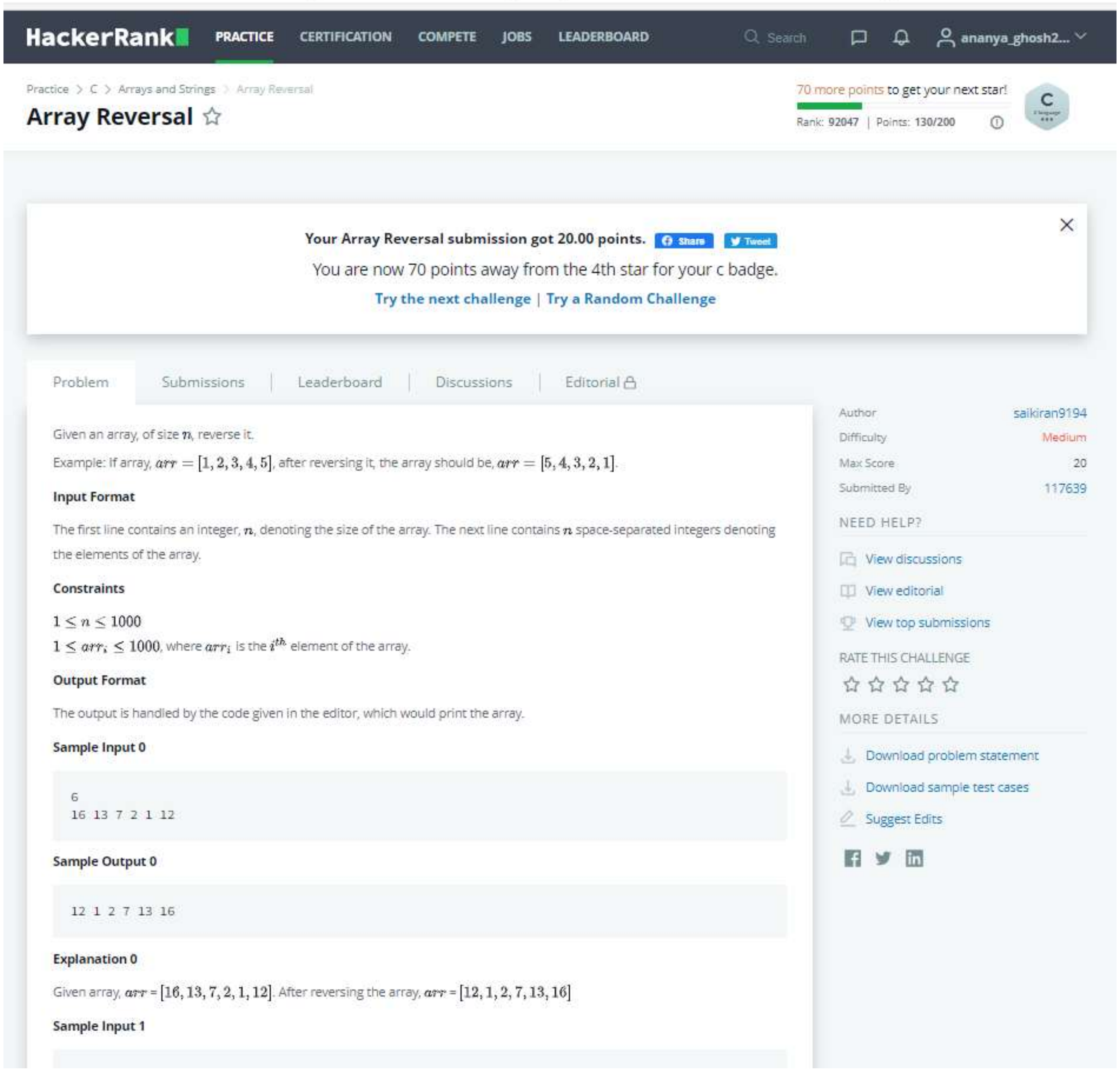
Input (stdin)

```
6
16 13 7 2 1 12
```

Expected Output

```
51
```

11.



Practice > C > Arrays and Strings > Array Reversal

Array Reversal

70 more points to get your next star!

Rank: 92047 | Points: 130/200

Your Array Reversal submission got 20.00 points. Share Tweet

You are now 70 points away from the 4th star for your c badge.

Try the next challenge | Try a Random Challenge

Problem | Submissions | Leaderboard | Discussions | Editorial

Given an array, of size n, reverse it.

Example: If array, $arr = [1, 2, 3, 4, 5]$, after reversing it, the array should be, $arr = [5, 4, 3, 2, 1]$.

Input Format

The first line contains an integer, n, denoting the size of the array. The next line contains n space-separated integers denoting the elements of the array.

Constraints

$1 \leq n \leq 1000$

$1 \leq arr_i \leq 1000$, where arr_i is the i^{th} element of the array.

Output Format

The output is handled by the code given in the editor, which would print the array.

Sample Input 0

```
6
16 13 7 2 1 12
```

Sample Output 0

```
12 1 2 7 13 16
```

Explanation 0

Given array, $arr = [16, 13, 7, 2, 1, 12]$. After reversing the array, $arr = [12, 1, 2, 7, 13, 16]$

Sample Input 1

Author	saikiran9194
Difficulty	Medium
Max Score	20
Submitted By	117639

NEED HELP?

View discussions

View editorial

View top submissions

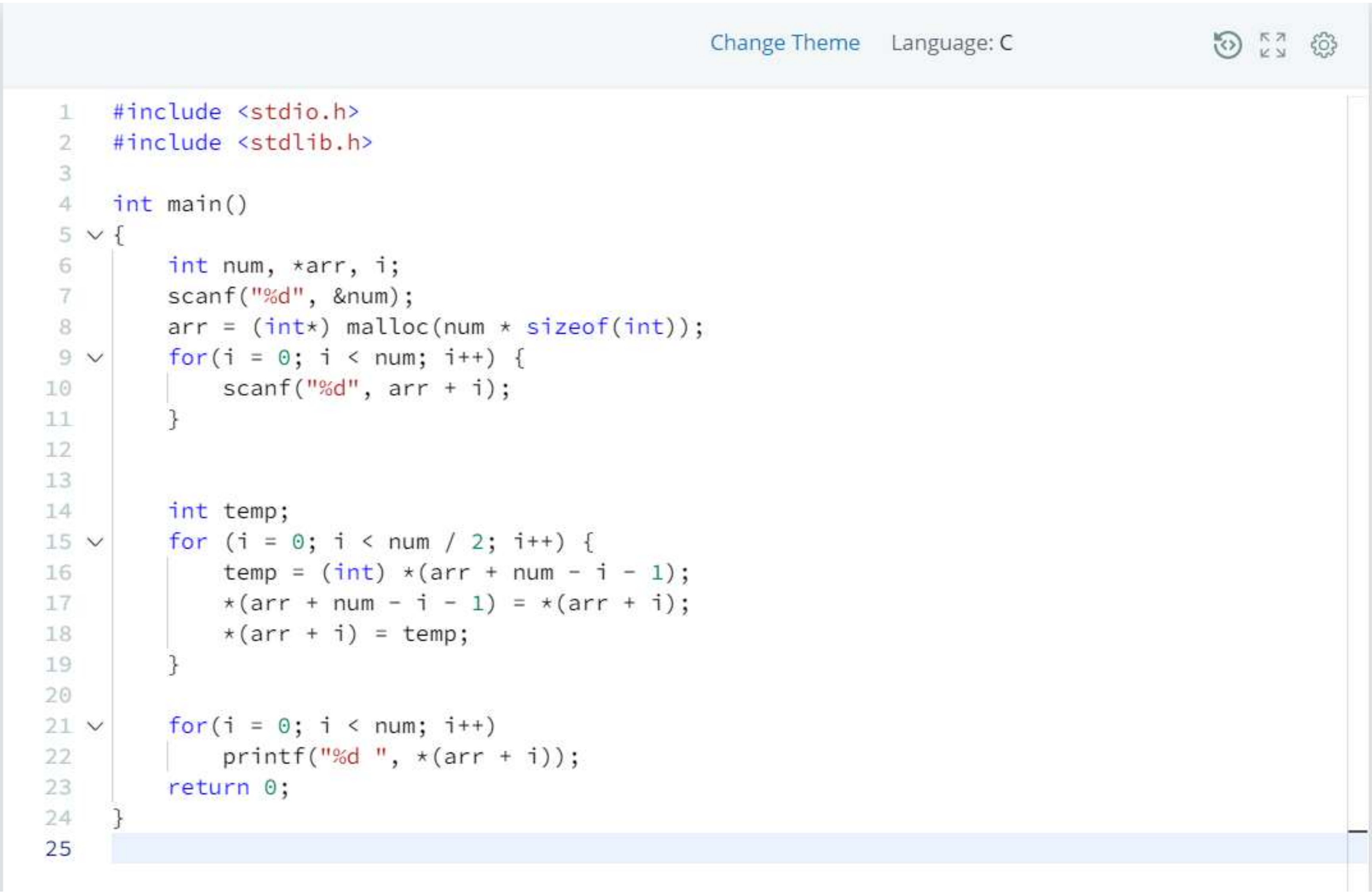
RATE THIS CHALLENGE

MORE DETAILS

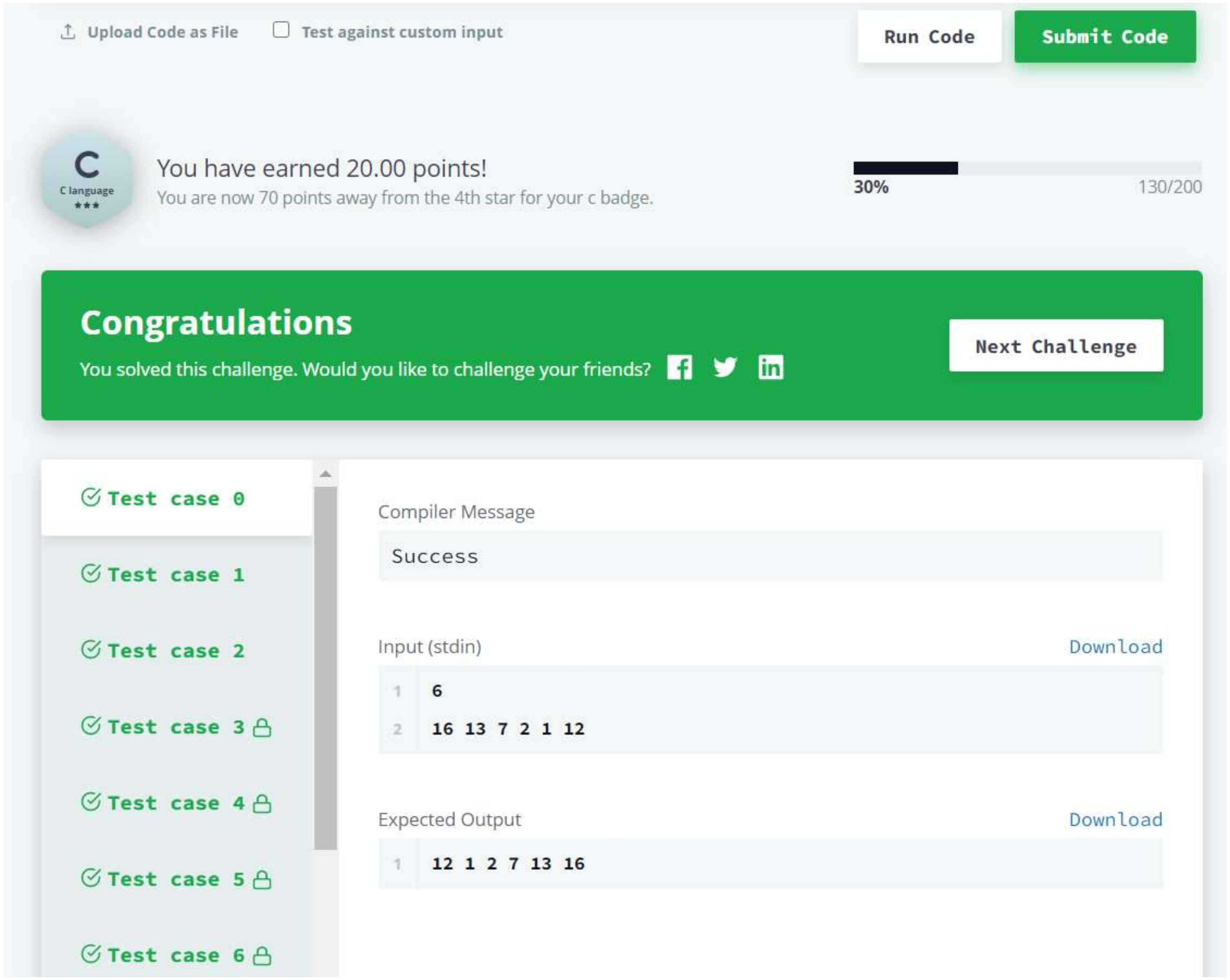
Download problem statement

Download sample test cases

Suggest Edits



```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int num, *arr, i;
    scanf("%d", &num);
    arr = (int*) malloc(num * sizeof(int));
    for(i = 0; i < num; i++) {
        scanf("%d", arr + i);
    }


    int temp;
    for (i = 0; i < num / 2; i++) {
        temp = (int) *(arr + num - i - 1);
        *(arr + num - i - 1) = *(arr + i);
        *(arr + i) = temp;
    }

    for(i = 0; i < num; i++)
        printf("%d ", *(arr + i));
    return 0;
}
```



You have earned 20.00 points!

You are now 70 points away from the 4th star for your c badge.

30% 130/200

Congratulations

You solved this challenge. Would you like to challenge your friends?

✓ Test case 0
✓ Test case 1
✓ Test case 2
✓ Test case 3
✓ Test case 4
✓ Test case 5
✓ Test case 6

Compiler Message

```
Success
```

Input (stdin)

```
6
16 13 7 2 1 12
```

Expected Output

```
12 1 2 7 13 16
```