Question **1**

Correct

Marked out of 3.00

Flag question

Objective

This is a simple challenge to help you practice printing to stdout.

We're starting out by printing the most famous computing phrase of all time! In the editor below, use either printf or cout to print the string ***Hello, World!*** to stdout.

Input Format

You do not need to read any input in this challenge.

Output Format

Print ***Hello, World!*** to stdout.

Sample Output

```
Hello, World!
```

Sample Output

Hello, World!

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    printf("Hello, World!");
    return 0;
}
```

	Expected	Got	
✓	`Hello, World!`	`Hello, World!`	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

Flag question

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);
```

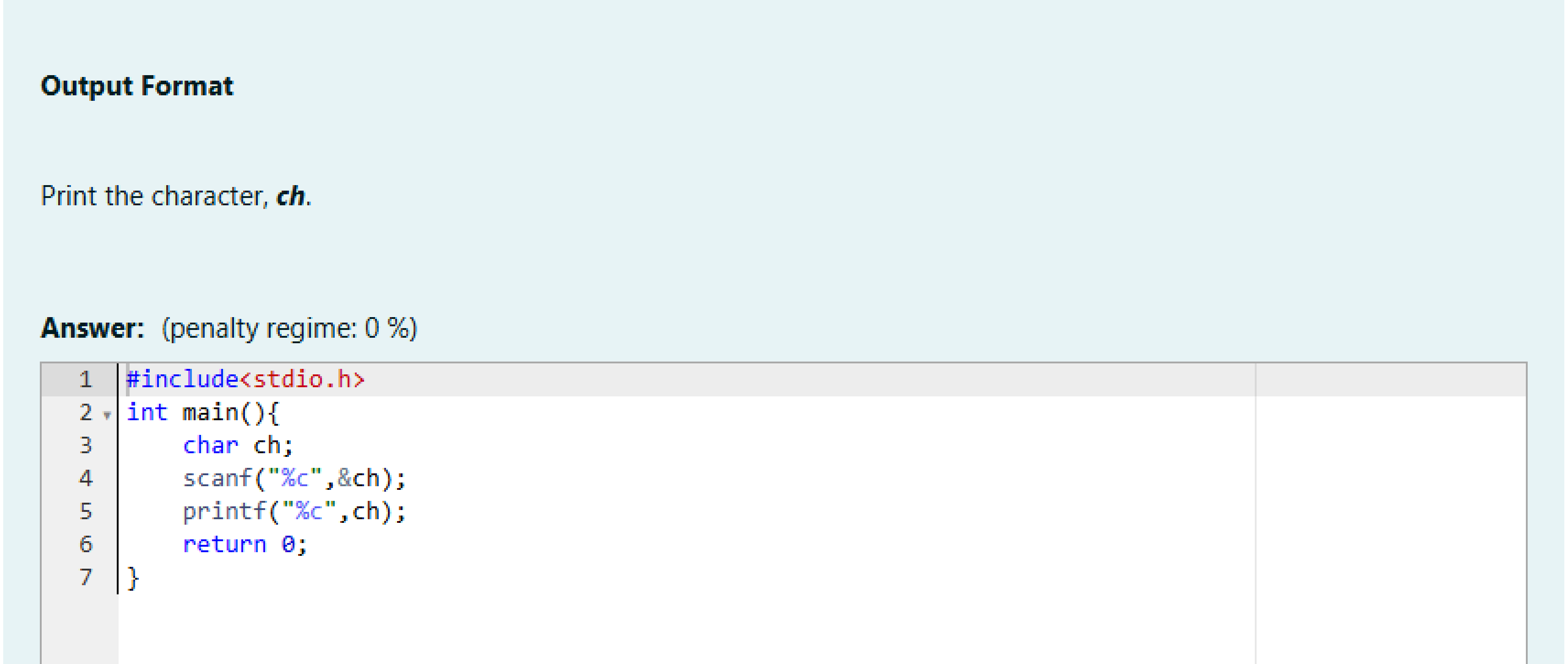
This piece of code prints the character ***ch***.

Task

You have to print the character, ***ch***.

Input Format

Take a character, ***ch*** as input.



Output Format

Print the character, ***ch***.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    char ch;
    scanf("%c",&ch);
    printf("%c",ch);
    return 0;
}
```

	Input	Expected	Got	
✓	c	c	c	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 7.00

Flag question

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:

- o Print the sum and difference of two int variable on a new line.
- o 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

- · $1 \leq \textit{integer variables} \leq 10^4$
- · $1 \leq \textit{float variables} \leq 10^4$

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
```

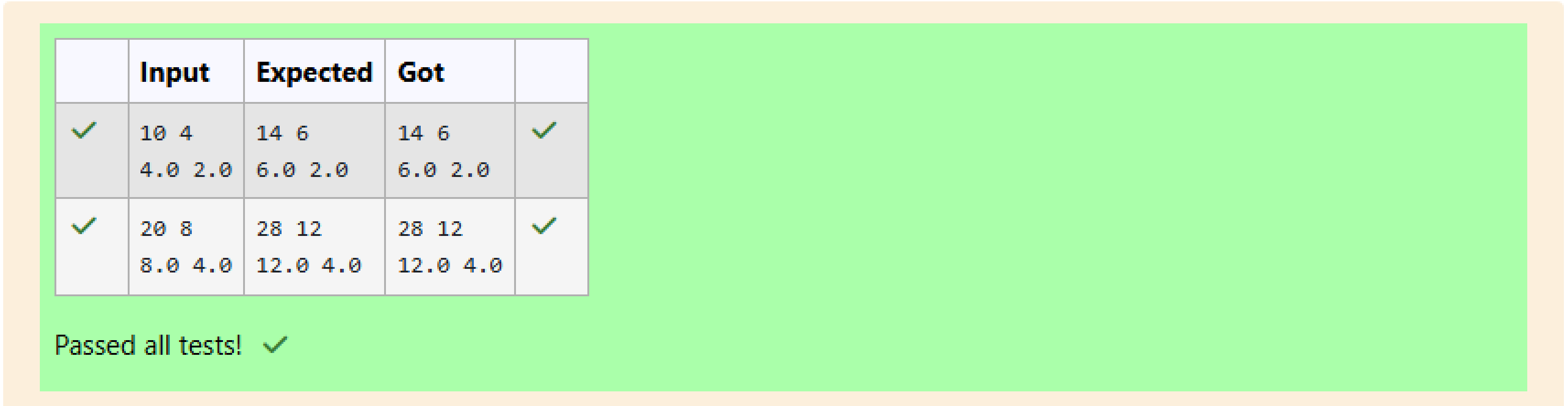
Explanation

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.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int a,b;
    float c,d;
    scanf("%d" "%d",&a,&b);
    scanf("%f" "%f",&c,&d);
    printf("%d %d\n",a+b,a-b);
    printf("%0.1f %0.1f\n",c+d,c-d);
    return 0;
}
```



	Input	Expected	Got	
✓	10 4 4.0 2.0	14 6 6.0 2.0	14 6 6.0 2.0	✓
✓	20 8 8.0 4.0	28 12 12.0 4.0	28 12 12.0 4.0	✓

Passed all tests! ✓

Constraints

Marks for each student lie in the range 0 to 100 (both inclusive)

Sample Input 1 :

```
A
3 4 6
```

Sample Output 1 :

```
A
4
```

Sample Input 2 :

```
T
7 3 8
```

Constraints

Marks for each student lie in the range 0 to 100 (both inclusive)

Sample Input 1 :

```
A
3 4 6
```

Sample Output 1 :

```
A
4
```

Sample Input 2 :

```
T
7 3 8
```

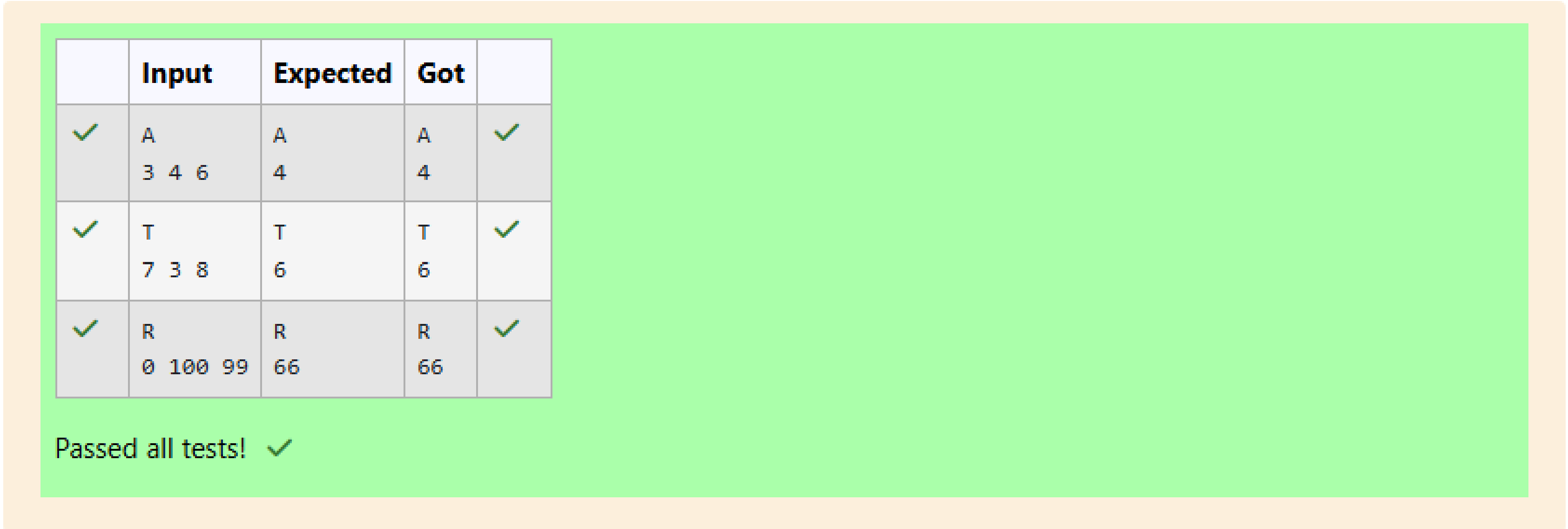
Sample Output 2 :

T

6

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    char name;
    int m1,m2,m3,avg;
    scanf("%c",&name);
    scanf("%d %d %d",&m1,&m2,&m3);
    avg=(m1+m2+m3)/3;
    printf("%c\n%d",name,avg);
    return 0;
}
```



	Input	Expected	Got	
✓	A 3 4 6	A 4	A 4	✓
✓	T 7 3 8	T 6	T 6	✓
✓	R 0 100 99	R 66	R 66	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

Flag question

Some *C* data types, their format specifiers, and their most common bit widths are as follows:

· *Int ("%d"):* 32 Bit integer

· *Long ("%ld"):* 64 bit integer

· *Char ("%c"):* Character type

· *Float ("%f"):* 32 bit real value

· *Double ("%lf"):* 64 bit real value

Reading

To read a data type, use the following syntax:

scanf("`format_specifier`", &val)

For example, to read a *character* followed by a *double*:

char ch;

double d;

scanf("%c %lf", &ch, &d);

For the moment, we can ignore the spacing between format specifiers.

Printing

To print a data type, use the following syntax:

printf("`format_specifier`", val)

For example, to print a *character* followed by a *double*:

```
char ch = 'd';

double d = 234.432;

printf("%c %lf", ch, d);
```

Note: You can also use *cin* and *cout* instead of *scanf* and *printf*; however, if you are taking a million numbers as input and printing a million lines, it is faster to use *scanf* and *printf*.

Input Format

Input consists of the following space-separated values: *int*, *long*, *char*, *float*, and *double*, respectively.

Output Format

Print each element on a new line in the same order it was received as input. Note that the floating point value should be correct up to 3 decimal places and the double to 9 decimal places.

Sample Input

```
3 12345678912345 a 334.23 14049.30493
```

Sample Output

3

12345678912345

a

334.230

14049.304930000

Explanation

Print *int* ***3***,

followed by *long* ***12345678912345***,

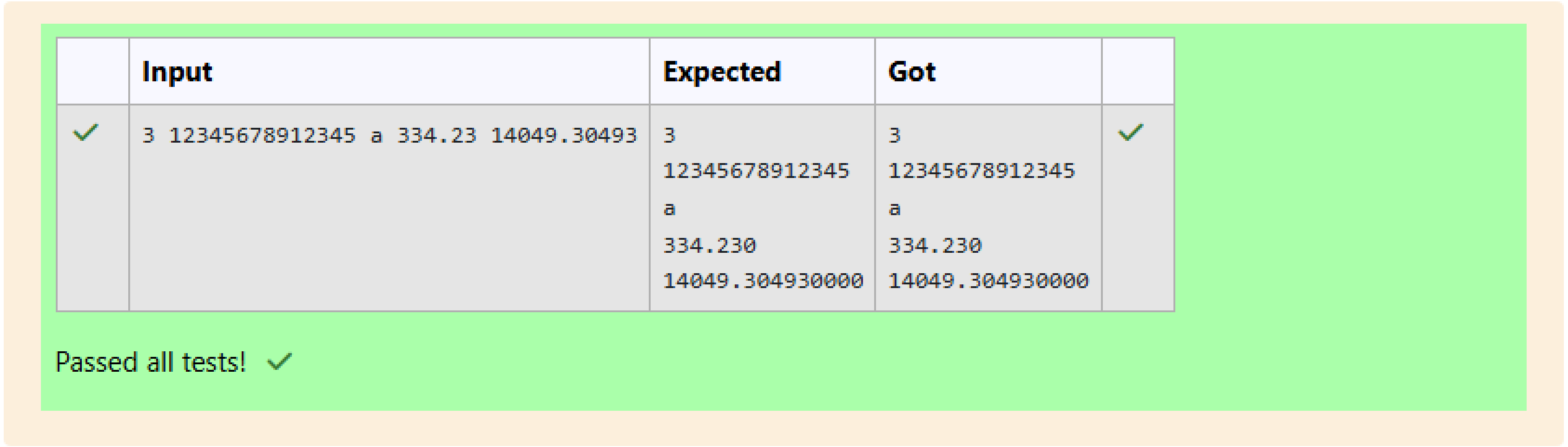
followed by *char* ***a***,

followed by *float* ***334.23***,

followed by *double* ***14049.30493***.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int a;
    long b;
    char c;
    float d;
    double e;
    scanf("%d\n%ld\n%c\n%f\n%lf\n",&a,&b,&c,&d,&e);
    printf("%d\n%ld\n%c\n%0.3f\n%0.9lf\n",a,b,c,d,e);
    return 0;
}
```



	Input	Expected	Got	
✓	`3 12345678912345 a 334.23 14049.30493`	3 12345678912345 a 334.230 14049.304930000	3 12345678912345 a 334.230 14049.304930000	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 7.00

Flag question

Write a program to print the ASCII value and the two adjacent characters of the given character.

Input

E

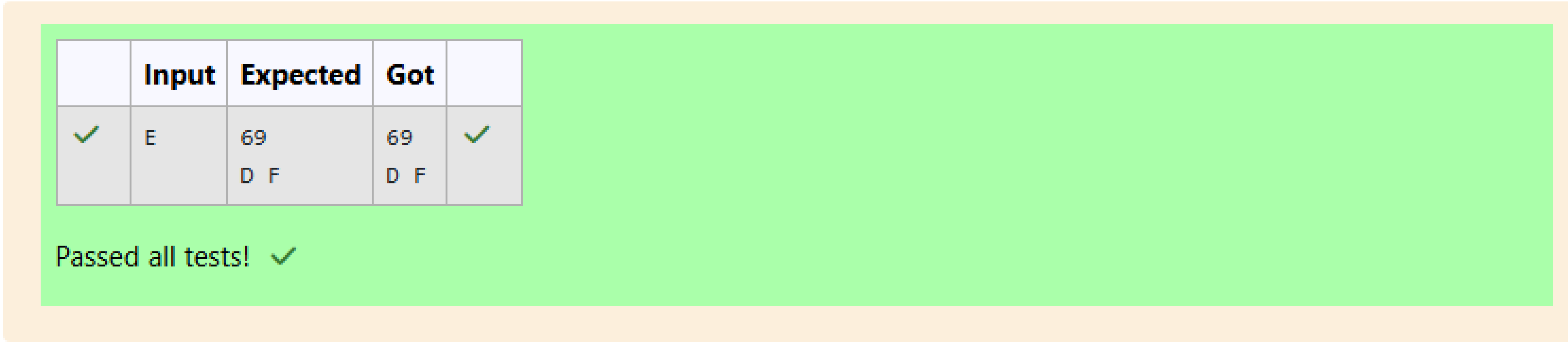
Output

69

D F

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    char ch;
    scanf("%c",&ch);
    printf("%d\n",ch);
    printf("%c %c",ch-1,ch+1);
    return 0;
}
```



	Input	Expected	Got	
✓	E	69 D F	69 D F	✓

Passed all tests! ✓