**Say "Hello, World!" With C**

Problem Statement:

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 1**

Hello, World!



**Playing with Characters**

**Problem Statement:**

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

chars [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:** After inputting the character and the string, inputting the sentence by the above

mentioned statement won't work. This is because, at the end of each line, a new line

character(\n) is present. So, the statement: scanf("%[^\n] %\*c", s); will not work because

the last statement will read a newline character from the previous line. This can be

handled in a variety of ways and one of them being: 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

**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 1**

C

program

Programming using C

**Sample Output 1**

C

program

Programming using C



**Sum and Difference of Two Numbers**

**Problem Statement:**

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 ≤ integer variables ≤ 104, 1 ≤ float variables ≤ 104

**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



**Average Marks**

**Problem Statement**

Write a program to input a name (as a single character) and marks of three tests as m1,

m2, and m3 of a student considering all the three marks have been given in integer format.

Now, you need to calculate the average of the given marks and print it along with the name

as mentioned in the output format section.

All the test marks are in integers and hence calculate the average in integer as well. That

is, you need to print the integer part of the average only and neglect the decimal part.

**Input Format :**

Line 1 : Name(Single character)

Line 2 : Marks scored in the 3 tests separated by single space.

**Output Format:**

First line of output prints the name of the student. Second line of the output prints the

average mark.

**Constraints**

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

**Sample Input 1 :**

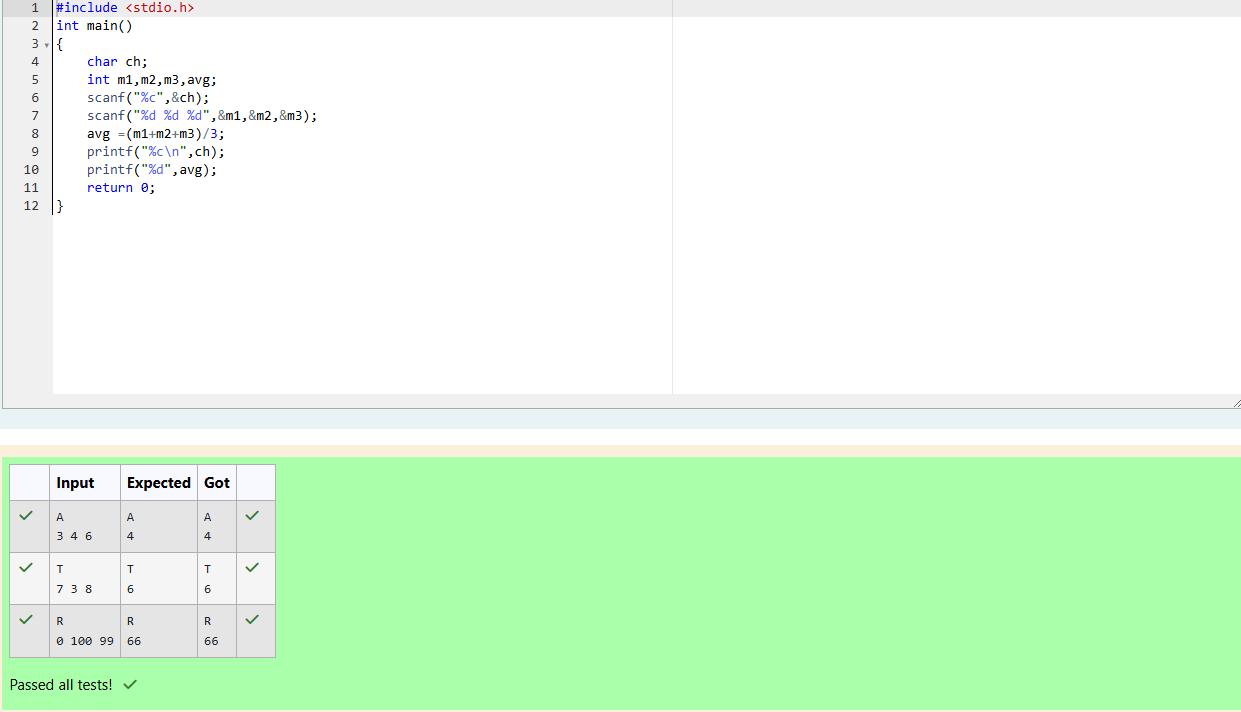
A

3 4 6

**Sample Output 1 :**

A

4



**Basic Data Types**

**Problem Statement:**

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**

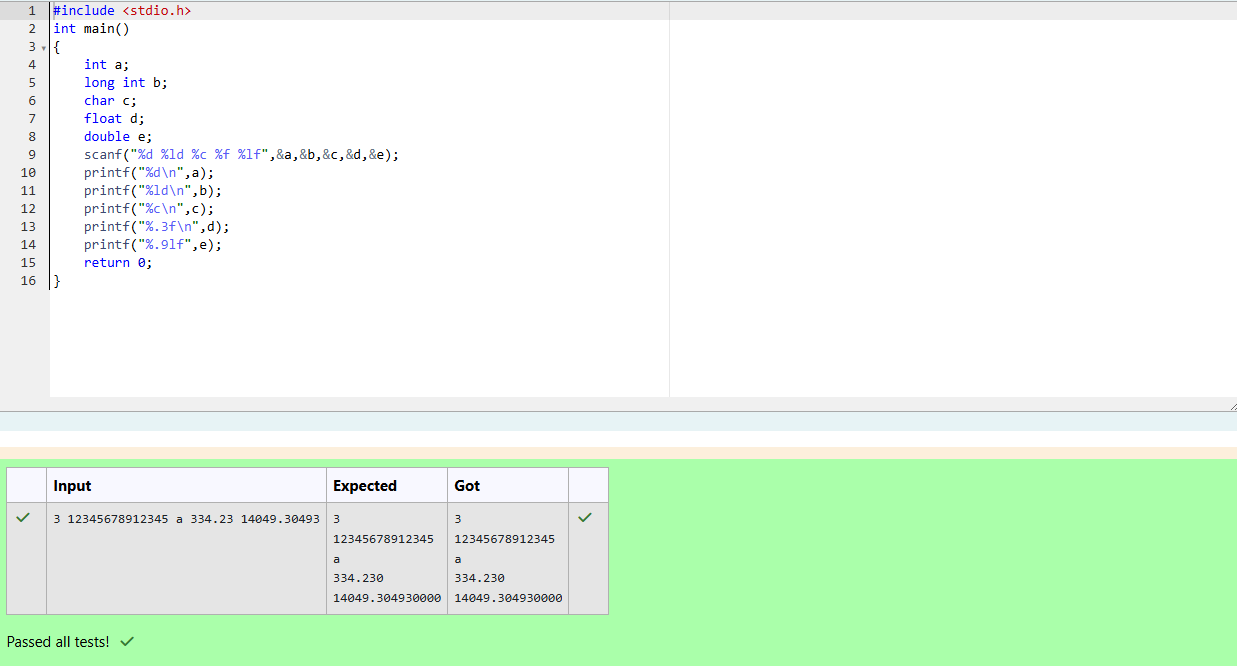
3

12345678912345

a

334.230

14049.304930000



**ASCII Value and Adjacent Characters**

**Problem Statement:**

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

character.

**Input Format:** Reads the character

**Output Format:** First line prints the ascii value, second line prints the previous character

and next character of the input character

**Sample Input 1:**

E

**Sample Output 1:**

69

D F

