**Resistor Color Trio**

In Practice Mode

Introduction

If you want to build something using a Raspberry Pi, you'll probably use *resistors*. For this exercise, you need to know only three things about them:

* Each resistor has a resistance value.
* Resistors are small - so small in fact that if you printed the resistance value on them, it would be hard to read. To get around this problem, manufacturers print color-coded bands onto the resistors to denote their resistance values.
* Each band acts as a digit of a number. For example, if they printed a brown band (value 1) followed by a green band (value 5), it would translate to the number 15. In this exercise, you are going to create a helpful program so that you don't have to remember the values of the bands. The program will take 3 colors as input, and outputs the correct value, in ohms. The color bands are encoded as follows:
* Black: 0
* Brown: 1
* Red: 2
* Orange: 3
* Yellow: 4
* Green: 5
* Blue: 6
* Violet: 7
* Grey: 8
* White: 9

In resistor-color duo you decoded the first two colors. For instance: orange-orange got the main value 33. The third color stands for how many zeros need to be added to the main value. The main value plus the zeros gives us a value in ohms. For the exercise it doesn't matter what ohms really are. For example:

* orange-orange-black would be 33 and no zeros, which becomes 33 ohms.
* orange-orange-red would be 33 and 2 zeros, which becomes 3300 ohms.
* orange-orange-orange would be 33 and 3 zeros, which becomes 33000 ohms.

(If Math is your thing, you may want to think of the zeros as exponents of 10. If Math is not your thing, go with the zeros. It really is the same thing, just in plain English instead of Math lingo.)

This exercise is about translating the colors into a label:

"... ohms"

So an input of "orange", "orange", "black" should return:

"33 ohms"

When we get more than a thousand ohms, we say "kiloohms". That's similar to saying "kilometer" for 1000 meters, and "kilograms" for 1000 grams. So an input of "orange", "orange", "orange" should return:

"33 kiloohms"

Getting Started

Make sure you have read the "Guides" section of the [C track](https://exercism.io/my/tracks/c) on the Exercism site. This covers the basic information on setting up the development environment expected by the exercises.

Passing the Tests

Get the first test compiling, linking and passing by following the [three rules of test-driven development](http://butunclebob.com/ArticleS.UncleBob.TheThreeRulesOfTdd).

The included makefile can be used to create and run the tests using the test task.

make test

Create just the functions you need to satisfy any compiler errors and get the test to fail. Then write just enough code to get the test to pass. Once you've done that, move onto the next test.

As you progress through the tests, take the time to refactor your implementation for readability and expressiveness and then go on to the next test.

Try to use standard C99 facilities in preference to writing your own low-level algorithms or facilities by hand.