# Thinking Like a Programmer

## Mistakes are INEVITABLE

> *"Experience is the name everyone gives to their mistakes."* – Oscar Wilde

In programming there is no truer statement. You cannot get better at programming without making a lot of mistakes along the way. In fact, a very common truism in programming is, **"if something works the first time you run it, be very afraid."**

Mistakes come in all types. Misspell a word here, use the wrong character there, forget a semi-colon or curly brace there, and all of a sudden you'll see red on your screen. This should not cause fear. In fact, errors are your friend. They're the greatest teacher and best pathway to learning. In order to understand what you need to do, you often first need to know what *NOT* to do.

See if you can figure out what's wrong with the following sentence:

```
My friend Ron is from The United Kingdom and he loves French cuisine.
```

Did you spot the error? Great! There are, in fact, *more than one* problem with the sentence above. Did you find them, or did you stop once you discovered the first one?

A common issue beginning developers have is that once they encounter an error with their code, they tend to stop paying attention to the rest of it. They can become fixated on the one problem and not realize another problem might also be lurking. After they fix the first bug, they will see another error message due to the second problem but believe the error message is still caused by the first error. It can take days for them to realize they fixed the original issue all along.

Even experienced developers can spend days trying to fix an error in their code. Since a computer interprets the language literally, you must be perfect in how you articulate the instructions — you saw this with the peanut butter and jelly sandwich example. The instructions might be very clear to you as you write them, but when taken literally they don't make sense.

Looking back at the sentence about Ron, despite having two errors, were you still able to understand the meaning? More than likely you were; however, a computer doesn't think the same as you do. It takes a very keen eye and intricate understanding of the language to know that the `T` in the word `the`, when used as an adjective before a proper noun, should be lowercase. That’s just the first error. The second error is that there should be a comma before the conjunction `and` to separate the two independent clauses.

## Logical Errors

There is something else interesting about the sentence above: some grammar-correction software may not detect the errors. As far as the grammar-correction software can tell, `The` might be part of the proper noun `The United Kingdom`. It has no way of knowing your intention is to use `the` as an adjective to `United Kingdom`. A similar issue is happening with the missing comma. Without understanding the intent of the sentence the grammar-correction software will not know that it is two conjoined sentences. It can only interpret the words in their literal form.

Both of these errors are what would be called a `logical error` in programming. The sentence is grammatically correct as far as the grammar-correction software is concerned, but there are logical errors that cannot be determined by the computer since it can only understand the text literally. The rules of capitalization and punctuation often require an understanding of the intention of the phrase trying to be conveyed, and this is not something a computer can determine easily. The same is true with programming. The computer cannot determine the intent of your code, only the syntax you used to write it.

Even if you had a keen eye for both problems, consider how often you make grammatical errors in your own writing. Now imagine that every grammatical error you ever made was taken literally by a computer. Would the computer comprehend your instructions? How often would your intention be misinterpreted by the computer? Could your instructions be performed exactly as you wrote them but have unintended results?

You will make many mistakes along the way. Do not fear them, embrace them. They will help you grow as a programmer. The more mistakes you make, the faster you will grow.

< Previous

Next >