2.14. 🧑‍💻 Hard-Coding

As you begin programming, you'll see that there are many ways to solve problems. You'll also find that one of the thrills of programming is how easily you can do things correctly that humans could easily make errors on. For example, you'll learn how to write just a very small bit of code to find the 1047th character in a sentence that might be thousands of letters long, and you'll learn how to do the exact same computation on many different pieces of data.

We'll often tell you in this textbook *not to hard-code* answers. What does that mean?

Some things in programming you can only do by typing them out. As you've seen, when you have to assign a value to a variable, you simply type something like `xyz = 6`. No other way.

But in most cases, it's best not to do computation in your head or write complete answers to programming problems out by hand. That's where **hard-coding** comes in. "Don't hard code" basically means, you should rely on your code, your logic, your program, and you should *not* write things out by hand or do computation in your head – even if you can do so easily.

When you are writing code or working on the answer to a programming exericse, you should ask yourself: *Would my answer be correct even if the provided variables had different values?* If the answer to that question is no, you're probably hard- coding, which you should avoid – and there's probably at least a slightly more concise way to construct your answer!

For example, in this following code, if you're asked in an exercise to create a variable `zx` and assign it the value of the sum of the value of `y` and the value of `x`, writing `zx = 55` is *hard-coding*.

Save & Run Show Feedback Show Code Show CodeLens

Activity: 1 -- ActiveCode (hard_coding_example)

The operation `20 + 35` may be easy math to do in your head or with a calculator, but when you learn to program, you want to train yourself to notice useful patterns of how to solve problems, which will make your life easier (perhaps beyond programming, even).

The correct way to answer that sort of exercise would be to write: `zx = y + x` (or `zx = x + y`, as you were just reminded of the order of operations). That is not hard-coding, and it will be correct no matter what the values of `x` and `y` are.

In the code above, if the value of `x` were `40`, `55` would not be the correct value for `zx` to have. But `zx = y + x` would still be absolutely correct.

Try as much as you can not to rely on your brilliant but fallible human brain to do *computation* when you program – use your brain to determine how to write the correct code to solve the problem for you! That's why we require you to avoid hard -coding for most exercises in this book.

You have attempted 2 of 1 activities on this page

✔ Completed. Well Done!

2.13. Updating Variables">

odating Variables">

2.15. Input">

© Copyright 2017 bradleymiller. Created using Runestone 4.1.17.

| Back to top