The parts of speech of programming

Remember: programming languages are *languages!*

Different human languages use different vocabularies and syntax structures, but still often share fundamental concepts.

For example:

"My name is Susan."

"Je m'appelle Susan."

Both sentences express the same concept but are structured differently. Both languages have parts of speech like subjects, objects, verbs and modifiers.

nouns => *variables*

In English, a word is a noun if it can be classified as a "person", "place" or "thing". These describe the *types* of items that nouns can be.

In programming, there are five primary *types* of variables: *numbers, strings, objects, arrays,* and *booleans*. These classifications are also referred to as "data types."

You'll know it when you see it

Knowing what qualifies as a "person", "place" and "thing" is actually extremely complicated, but we spend all of our lives both using and testing these concepts.

Computers are much less clever and social than people are, so programming languages use punctuation to clearly indicate each type of variable.

Recognizing datatypes

25

"hello"

["fourteen", 12, [4,5,6], {candy:"yes", type:"chocolate"}]

{name: "Kate Middleton", age: 31, hat:{style:"pillbox", color:"white"}, sibling_names: ["Pippa", "James William"]}

true

Recognizing datatypes

25 => *if it's a number, it's a number*

"hello" => *text surrounded by quotation marks is a string*

["fourteen", 12, [4,5,6], {candy:"yes", type:"chocolate"}]

=> *arrays are lists surrounded by square brackets, with each item separated from the next by a comma*

{name: "Kate Middleton", age: 31, hat:{style:"pillbox", color:"white"}, sibling_names: ["Pippa", "James William"]}

=> *objects are also a kind of list, but each item is labeled. The label (also called a "key") is separated from the value by a colon. Objects are surrounded by curly braces.*

true => *if it's* true *or* false *(and *not* surrounded by quotes) it's a boolean*

By any other name

Real-world nouns are complex and unique, which is why we have names for things. That way we can reference them without having to describe them in detail over and over again.

Variables generally have names, too, for exactly the same reasons.

Defining the relationship

Writing a program is a lot like writing a story: your reader (the computer) knows nothing about the topic to start, and as you go you introduce subjects and sources, building and shaping the reader's experience.

The first time you mention to a source, you usually describe them in some meaningful way. Later on, you can simply refer to them by name, and your reader knows who you're talking about.

Computing follows a similar process in the way that programming languages *declare* variables and *assign* them values.

I do declare!

English:

The instructor is Susan McGregor.

After reading this sentence, you will associate the name "Susan McGregor" with the label instructor. *If someone asks you who is running this course, you'll remember this and say "Susan McGregor."*

code:

instructor = "Susan McGregor"

Above is the equivalent sentence in code. The var *keyword tells the computer that it needs to remember the label that immediately follows. The equals sign (=) then tells it to associate that label with whatever is on the right hand side - which could be any of our data types. Notice that programming sentences end with a semicolon, not a period.*

Programming languages: mostly your own

In the previous example, we used the label (or name) *instructor* for our variable, but there's nothing magic about that.

Variable names can be almost anything you like, but since their values can be so abstract, it's a good idea to name them descriptively (though lots of programmers don't follow this advice).

Things to avoid when creating variable names? Punctuation (other than underscores) or so-called "keywords" that describe data types (e.g. number, array, string) or computer functions (e.g. save, delete &c).