**Variables** – Well, a variable is a placeholder. In code, it basically allows you to store a value and then reference it. Imagine you have a lot of math problems:

10 + 10 = 20

10 – 7 = 3

10 / 2 = 5

10 \* 3 = 30

What if I wanted to use 20 instead of 10 in all of those problems? I’d have to write them all out again. With a variable, I could tell you that ‘a’ = 10, and then use ‘a’ in those problems:

a + 10 = 20

a - 7 = 3

a / 2 = 5

a \* 3 = 30

Now, when I want to use 20, or 30, or 40 instead of 10, I can just redefine the variable. Take the same problems, but a = 20. The answers are now 30, 13, 10, and 60. If I say a = 5, the answers are 15, -2, 2.5, and 15. I can now change the entire set of problems by just changing 1 number, 1 time. It may not seem like a big deal with 4 problems, but what if there were 400? 4000?

**Strings** – Pretend strings are like speaking. If you were writing dialogue in a book, you’d have to use quotation marks, right? The same concept applies to strings. In programming, filter and ‘filter’ are different. ‘filter’ is treated like a word, where filter is treated like a variable (see above).

**Functions** – Functions are where you write your programs. It’s a set of instructions you can use over and over again, with different variables. Using math as an example again, a function could be ‘a + 10’. Now, every time you need to add 10 to something, you could call that function and replace the ‘a’ variable.

**If statements** – if statements determine whether something is true, and then does something based on the result. For example, if I said “If your name is Josh, run to the store and claim your free candy bar. If not, you can have an apple.” That’s an if statement.

**Boolean values** – Boolean values are true of false. It’s pretty self-explanatory.