Crash Course: Functions 2

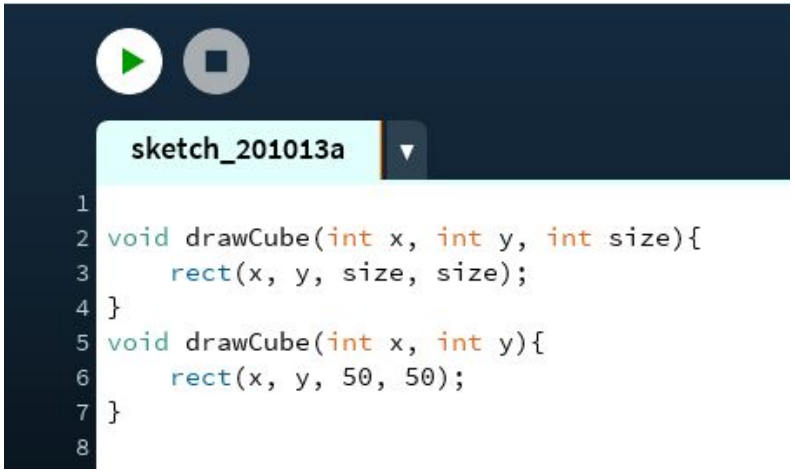
ST. MARY'S HIGH SCHOOL

Overloading Functions

We can make our functions take multiple different types of parameters if we overload it.

We can write another function with the same name and different parameters, so we can use the function many ways.

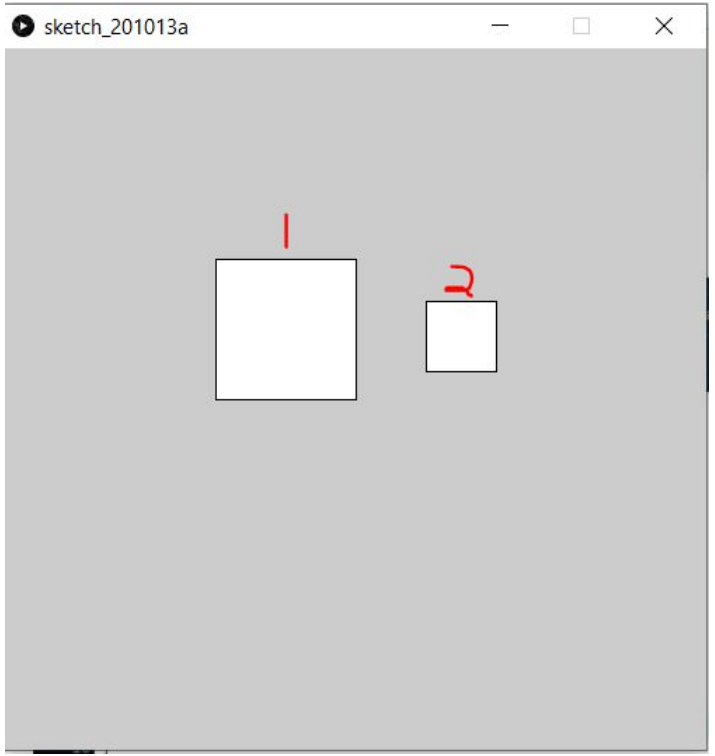
Like giving the function a default value.



The parameters decide which ‘drawCube’ function we use.

```
sketch_201013a

1
2  void drawCube(int x, int y, int size){
3      rect(x, y, size, size);
4  }
5  void drawCube(int x, int y){
6      rect(x, y, 50, 50);
7  }
8
9  void setup(){
10   size(500,500);
11 }
12
13 void draw(){
14   drawCube(150, 150, 100);
15   drawCube(300, 180);
16 }
```



Scope

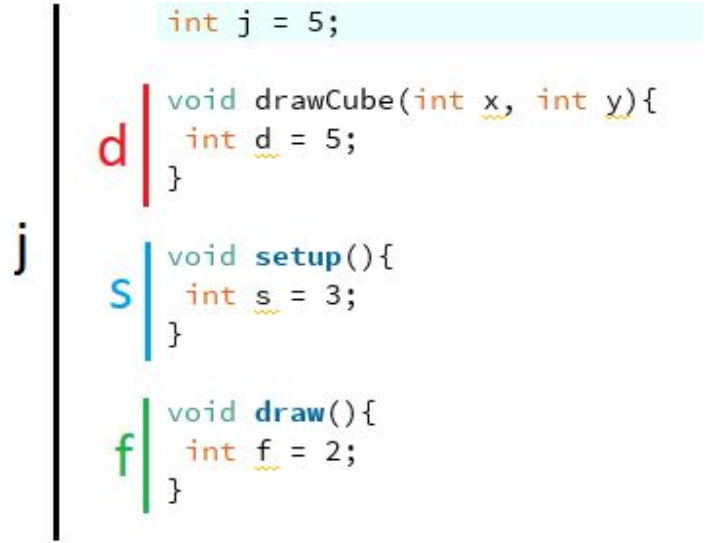
A variable declared inside a function can only be used inside that function:

‘j’ is a global variable and can be used anywhere inside the function.

‘d’ can only be used inside drawCube.

‘s’ can only be used inside setup and

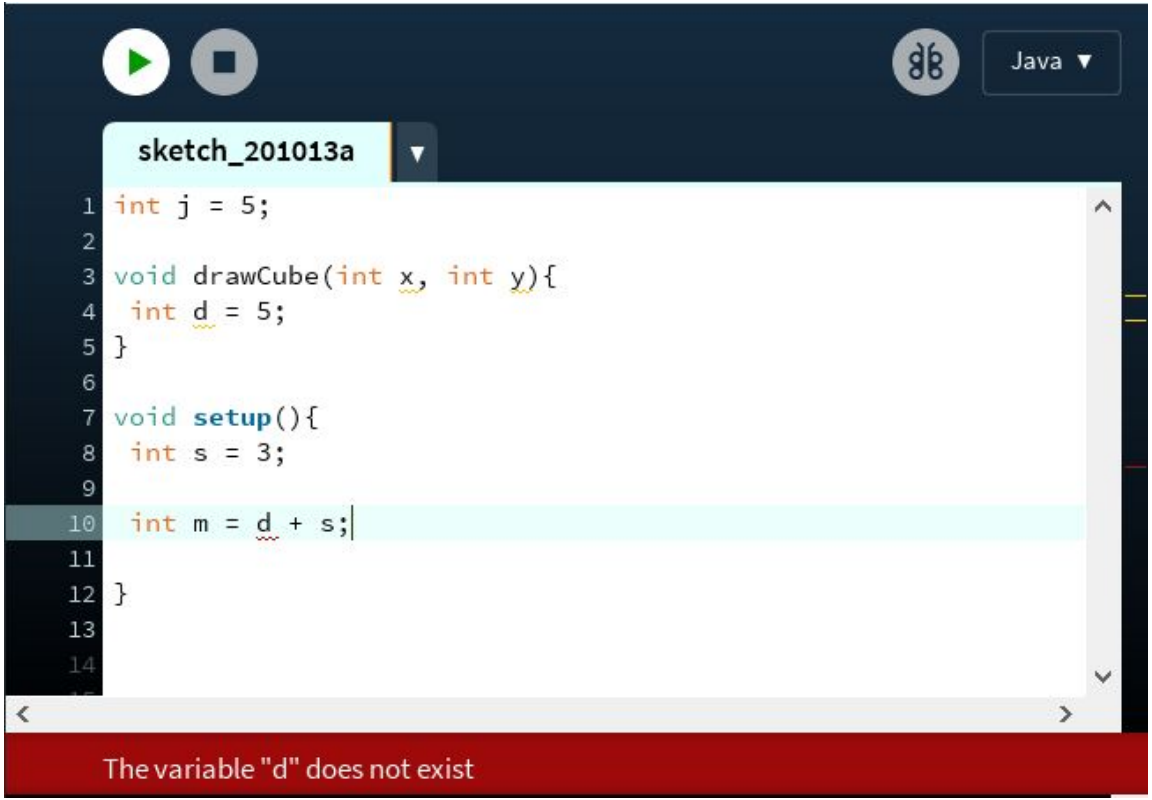
‘f’ can only be used inside draw.



Scope Errors

If we try and use ‘d’ outside of drawCube, we get an error.

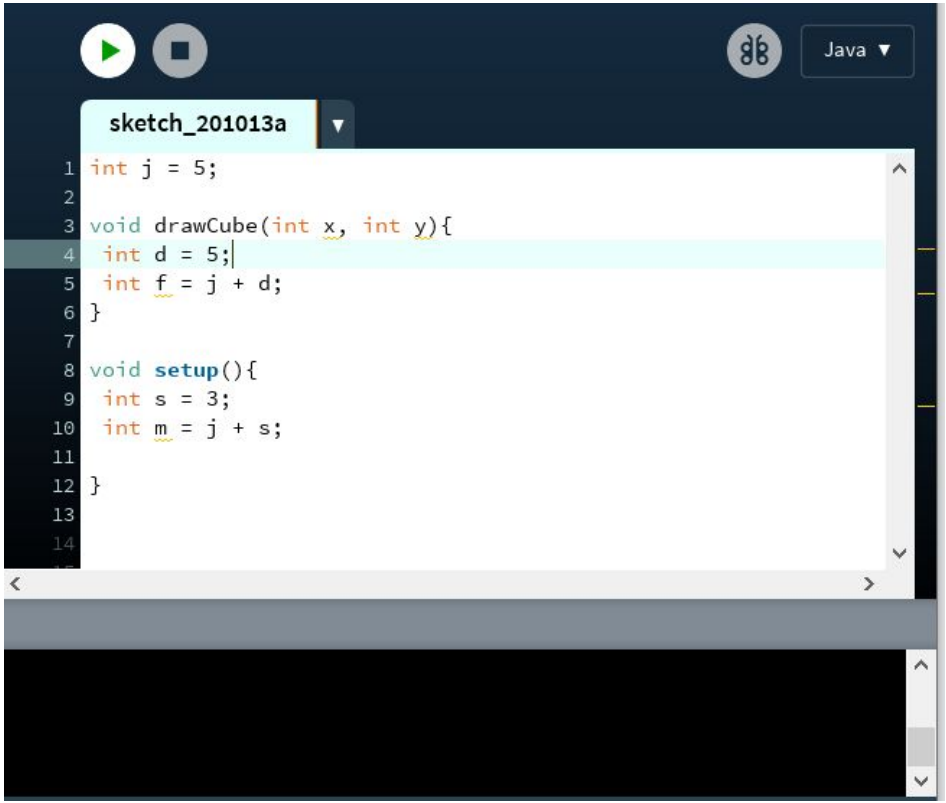
This is because ‘d’ only exists inside drawCube.



Global Variables

Because we declared ‘j’ outside of setup or drawCube, we can use ‘j’ anywhere.

We call these global variables. Use them sparingly to prevent errors.



Why use functions?

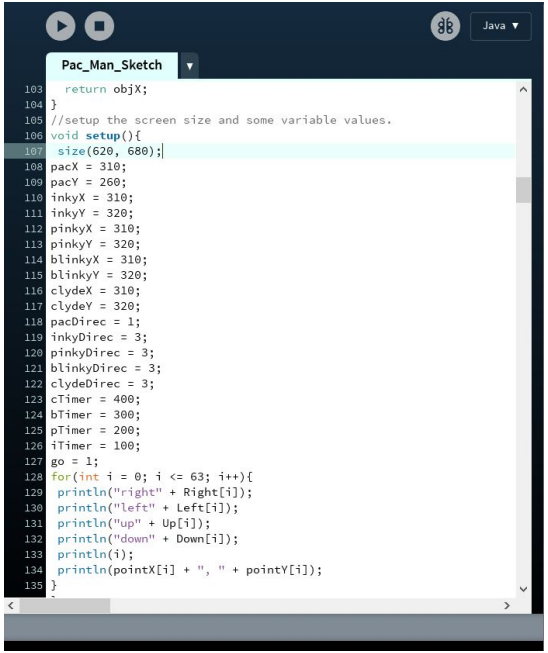
As projects get bigger, code gets more complicated and keeping track of variables becomes very difficult.

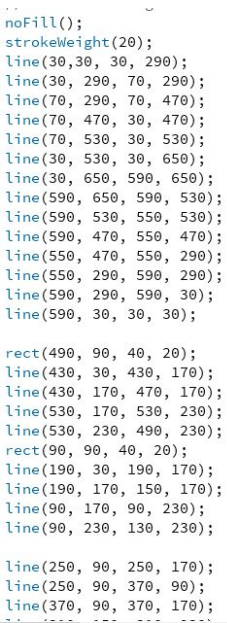
Using many functions is essential to keeping a project running smoothly.

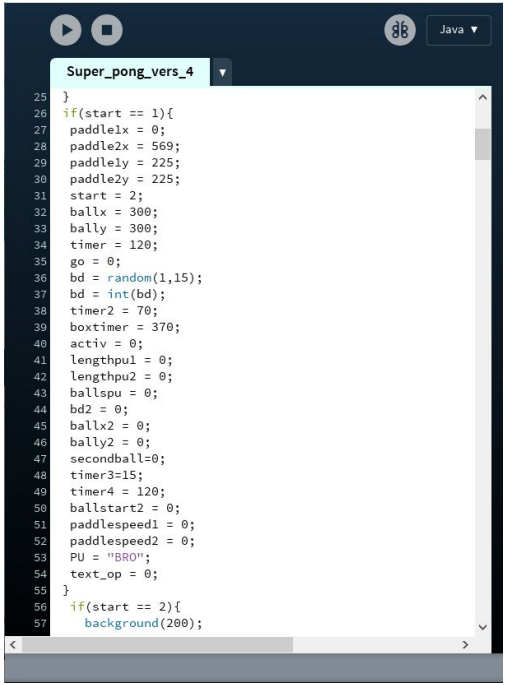
Not only do functions make your code easier to read and fix, but variable scope can help you keep track of variables.

Using functions help with working in teams. Have different members of the team develop different functions, then bring them all together for the project.

Code without many functions is long and difficult to understand







Using functions helps the code make a lot more sense