Functions in Processing

CSE 120 Spring 2017

Instructor: Justin Hsia

Teaching Assistants: Anupam Gupta, Braydon Hall, Eugene Oh, Savanna Yee

Administrivia

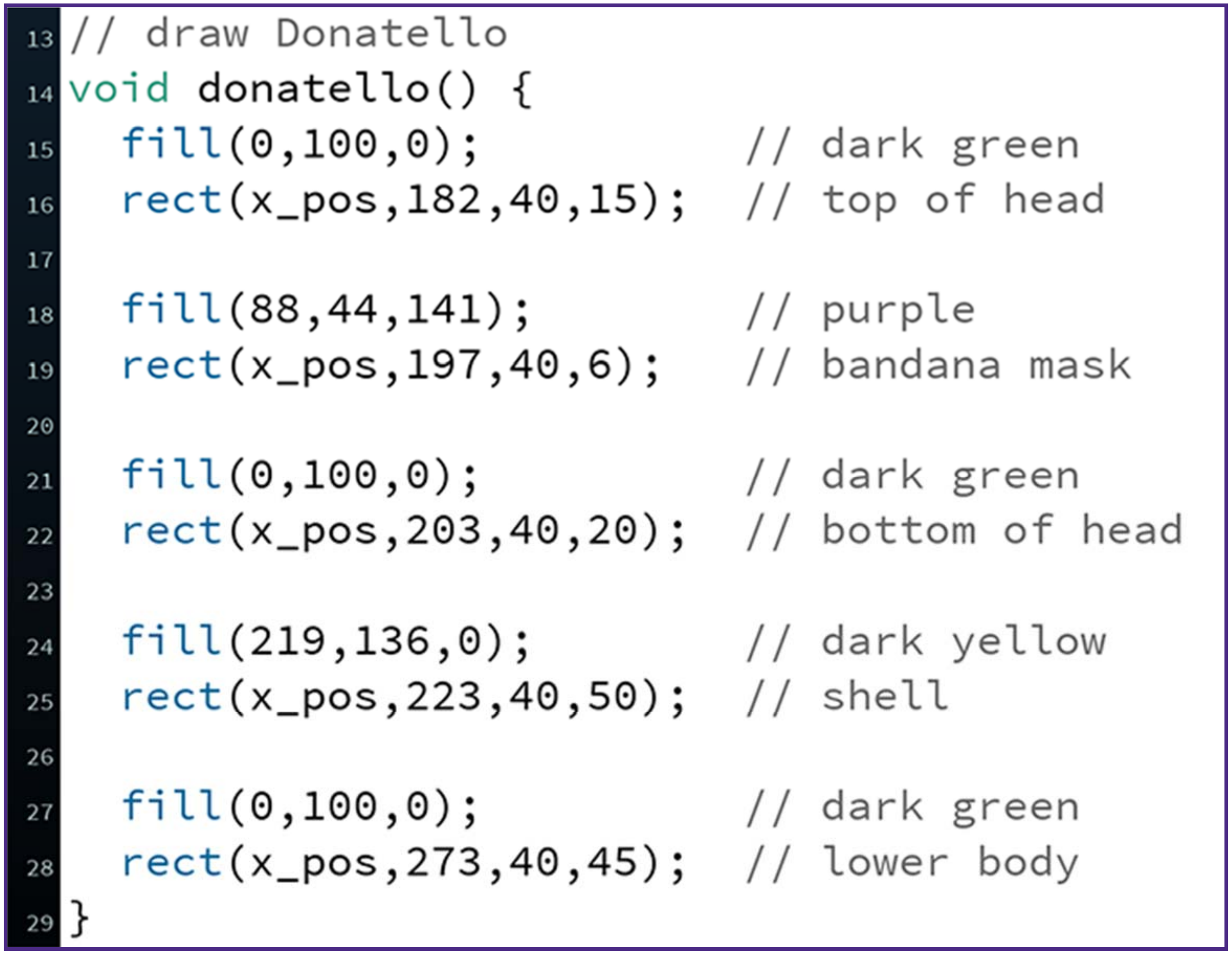
- Assignments:
  - Custom Logo due today (4/7)
  - Lego Family due Sunday (4/9)

- Make sure to take advantage of office hours and Piazza!

Drawing a Square with Functions

- [See Demo on Panopto]

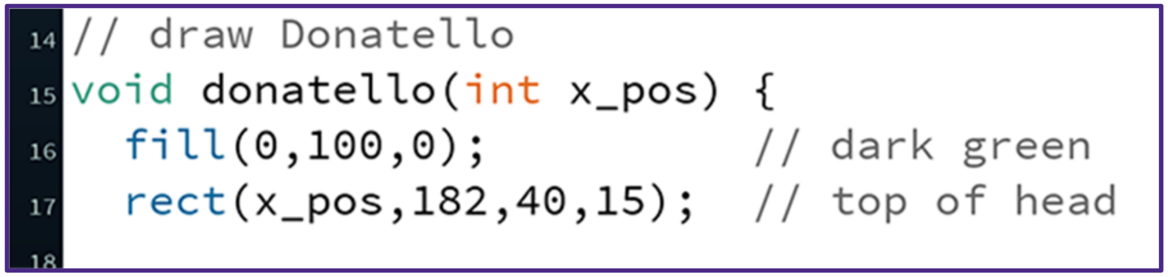
Donatello as a Function



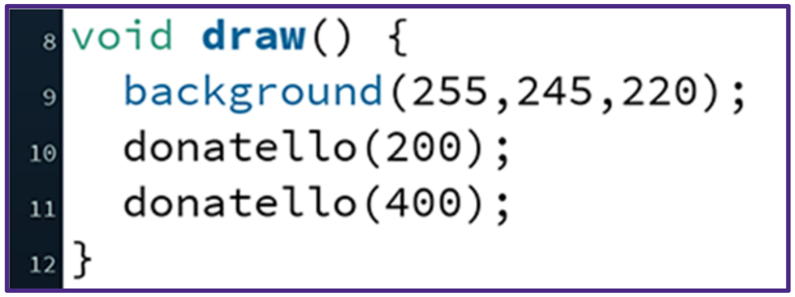
```
// draw Donatello
void donatello() {
  fill(0,100,0);              // dark green
  rect(x_pos,182,40,15);      // top of head

  fill(88,44,141);            // purple
  rect(x_pos,197,40,6);       // bandana mask

  fill(0,100,0);              // dark green
  rect(x_pos,203,40,20);      // bottom of head

  fill(219,136,0);            // dark yellow
  rect(x_pos,223,40,50);      // shell

  fill(0,100,0);              // dark green
  rect(x_pos,273,40,45);      // lower body
}
```

Donatello Function *Parameterized*

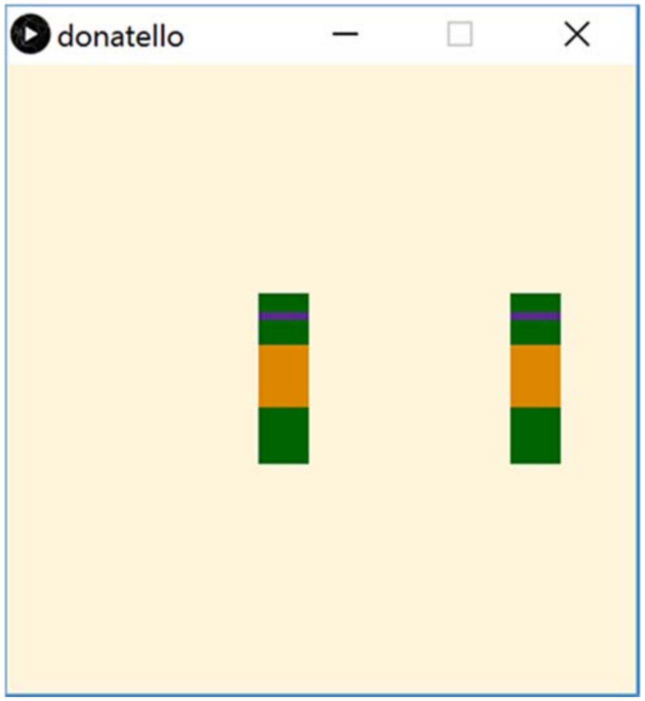
- Can now call `donatello()` function with different `x_pos`



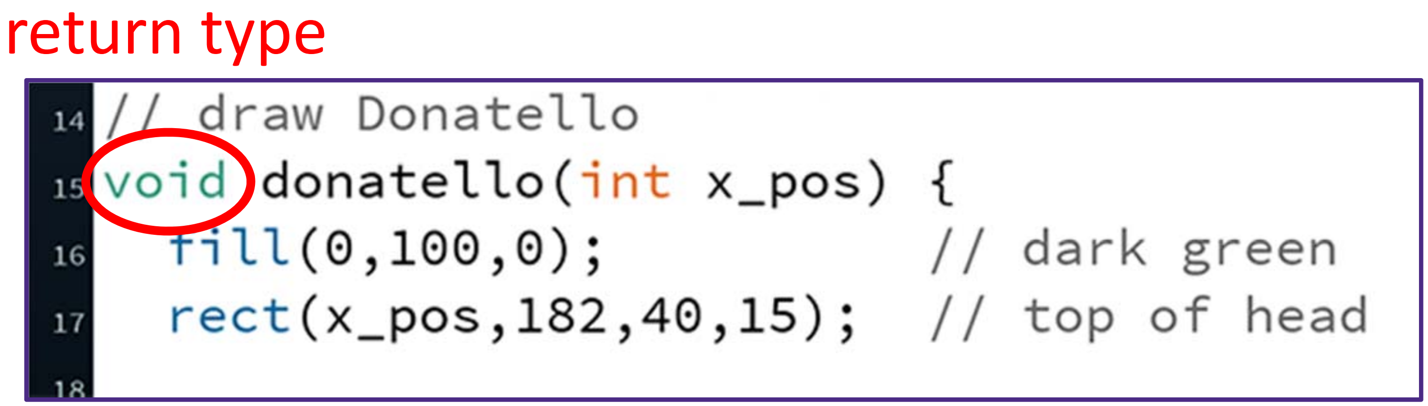
```
// draw Donatello
void donatello(int x_pos) {
  fill(0,100,0);            // dark green
  rect(x_pos,182,40,15);    // top of head

```



```
void draw() {
  background(255,245,220);
  donatello(200);
  donatello(400);
}
```



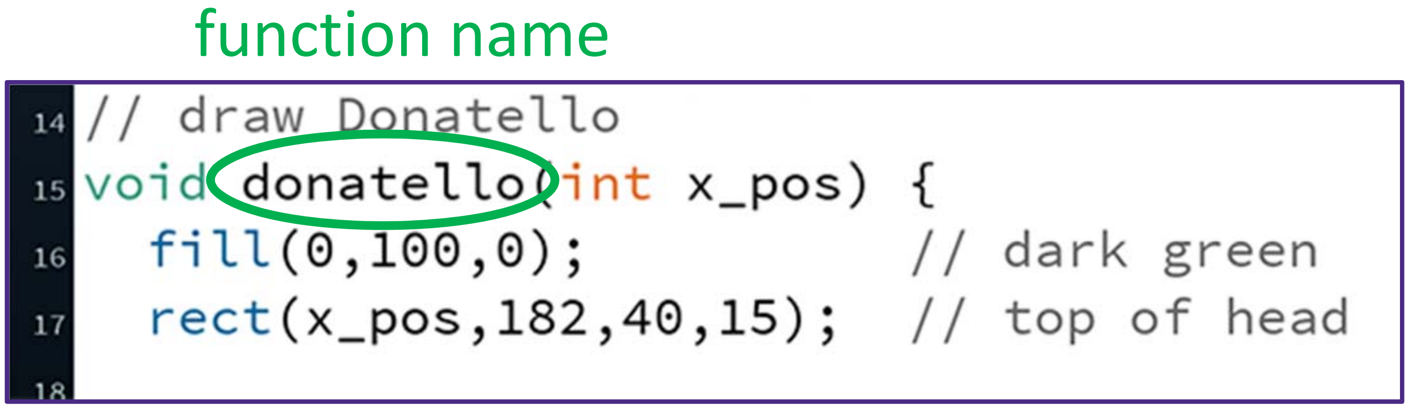
Return Type



```
// draw Donatello
void donatello(int x_pos) {
  fill(0,100,0);              // dark green
  rect(x_pos,182,40,15);     // top of head
```

- What the function sends back to whoever called it
  - Can be any of the datatypes: `int`, `float`, `color`, etc.
  - If not returning anything, then we use `void`

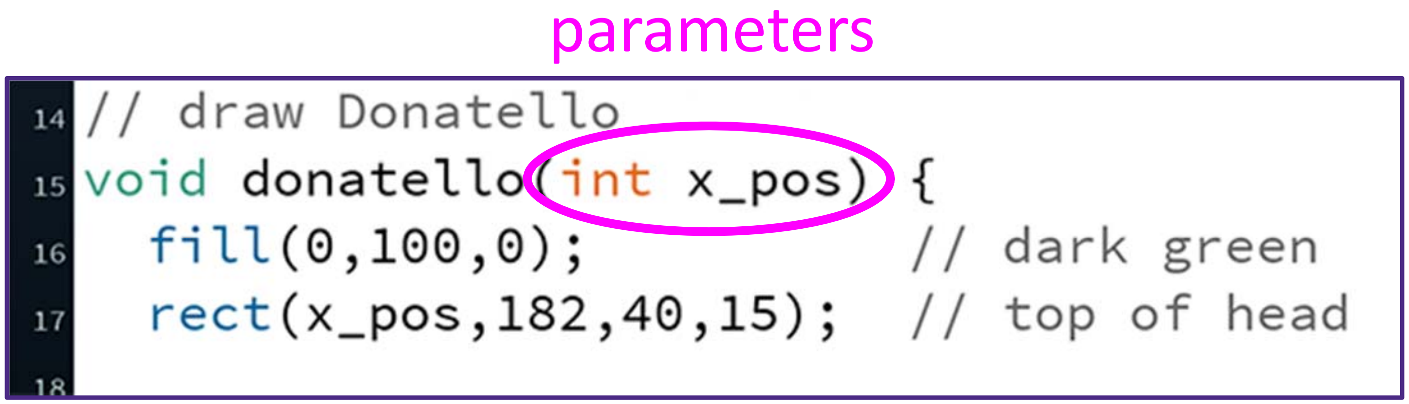
Function Name



```
// draw Donatello
void donatello(int x_pos) {
  fill(0,100,0);              // dark green
  rect(x_pos,182,40,15);  // top of head
```

- Does not matter to computer, but does to humans
  - Should describe what the function does
- *Must* start with a letter, but can contain numbers and underscores
  - Why not hyphen?
- No two functions (or variables) can have the same name

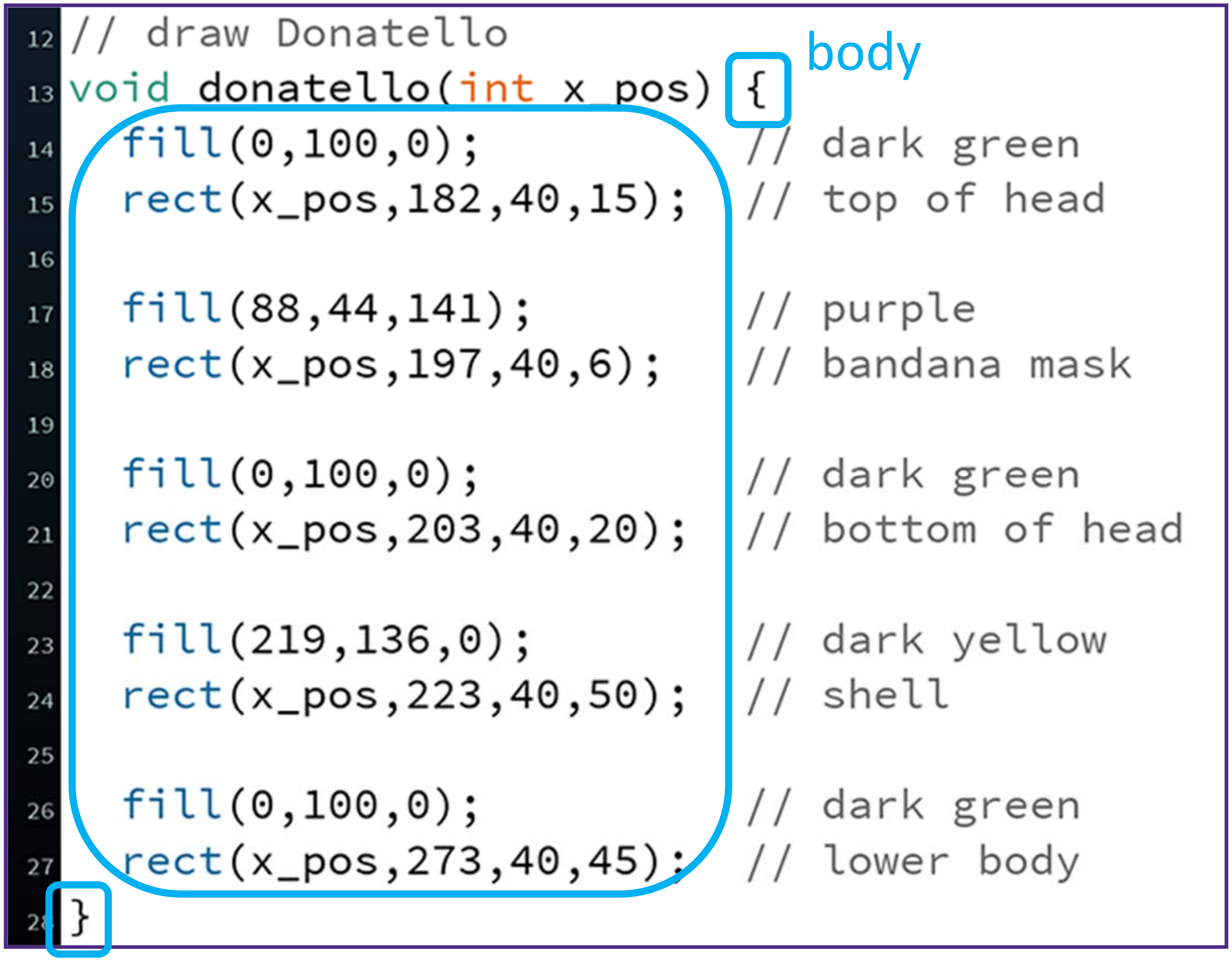
Parameters



```
// draw Donatello
void donatello(int x_pos) {
  fill(0,100,0);              // dark green
  rect(x_pos,182,40,15);      // top of head
```

- Required part of every function definition
  - Must be surrounded by parentheses
  - If no parameters, parentheses are left empty
- Datatype and name for every parameter must be specified
  - Separate parameters with commas

Function Body



```
// draw Donatello
void donatello(int x_pos) {
  fill(0,100,0);              // dark green
  rect(x_pos,182,40,15);      // top of head

  fill(88,44,141);            // purple
  rect(x_pos,197,40,6);       // bandana mask

  fill(0,100,0);              // dark green
  rect(x_pos,203,40,20);      // bottom of head

  fill(219,136,0);            // dark yellow
  rect(x_pos,223,40,50);      // shell

  fill(0,100,0);              // dark green
  rect(x_pos,273,40,45);      // lower body
}
```

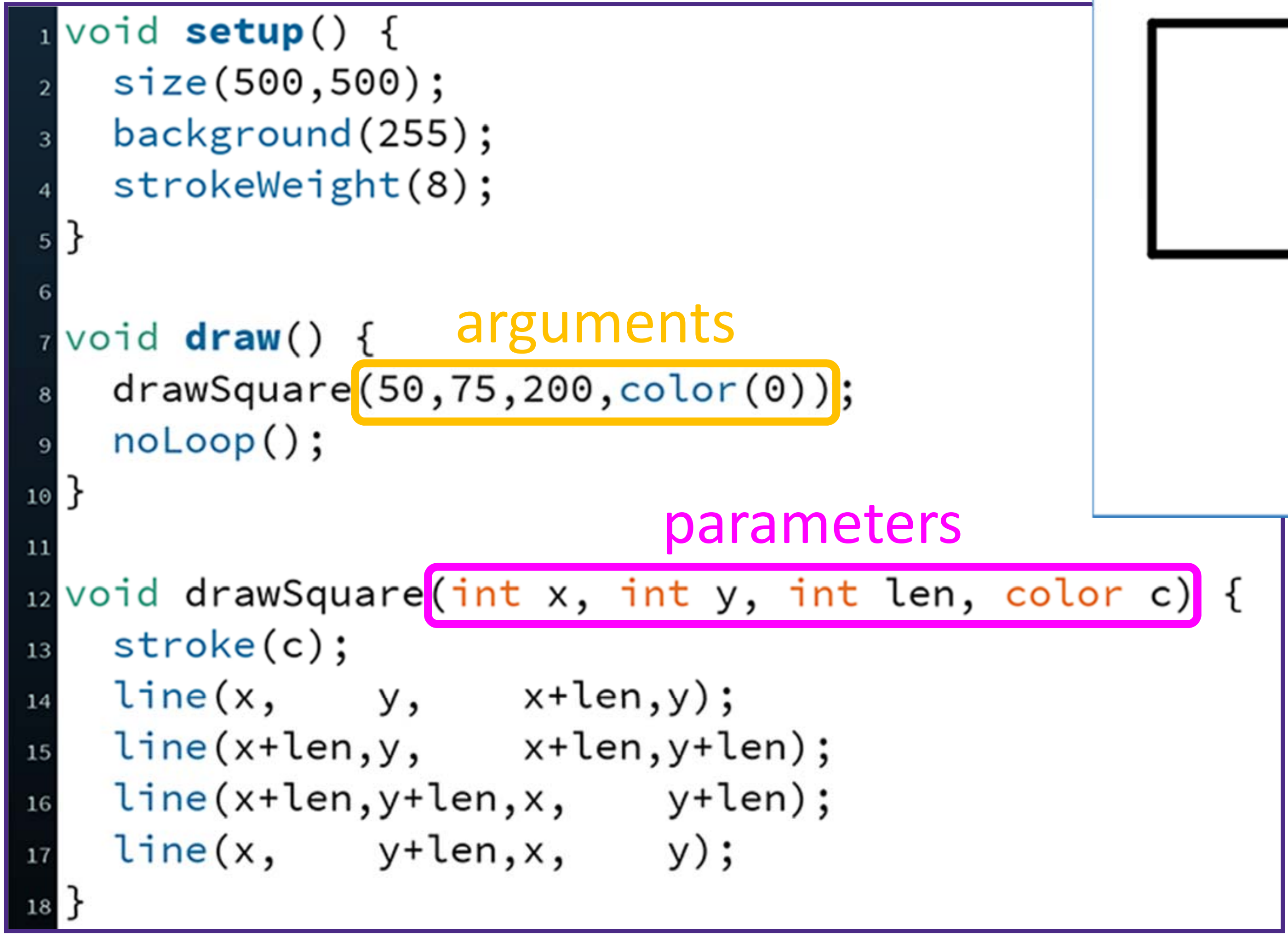
Lightbot Functions

- Lightbot functions had a different syntax, but similar parts:

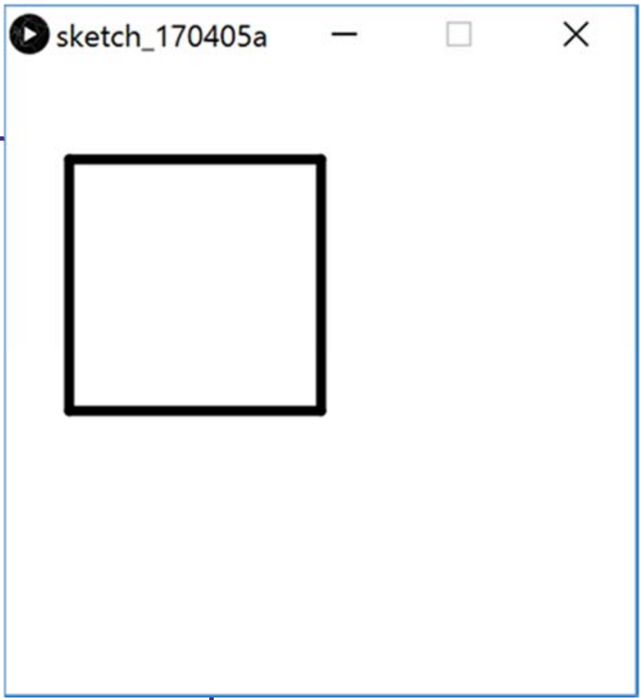
function name parameters body

```
F.turn_around() Right, Right.
```

Parameters vs. Arguments



```
void setup() {
  size(500,500);
  background(255);
  strokeWeight(8);
}

void draw() {
  drawSquare(50,75,200,color(0));
  noLoop();
}

void drawSquare(int x, int y, int len, color c) {
  stroke(c);
  line(x,      y,      x+len,y);
  line(x+len,y,      x+len,y+len);
  line(x+len,y+len,x,      y+len);
  line(x,      y+len,x,      y);
}
```



Parameters vs. Arguments

- When you define a function, you specify the parameters
  - Use parameters for values that you want to be different on different calls to this function
- When you call a function, you pass arguments
  - The order of the arguments must match the order of the parameters
- We define a function once, but can call it as many times as we want!

Variable Scope

- When an argument is passed to a function, what does the function actually get?
  - Internal variables (*i.e.* parameters) get a *copy* of the argument value

- Internal variables only exist within the function they are declared
  - The variables "cease to exist" when the function finishes
  - "Scope" of a variable is the part(s) of code where that variable name binding is valid (*i.e.* where it exists)

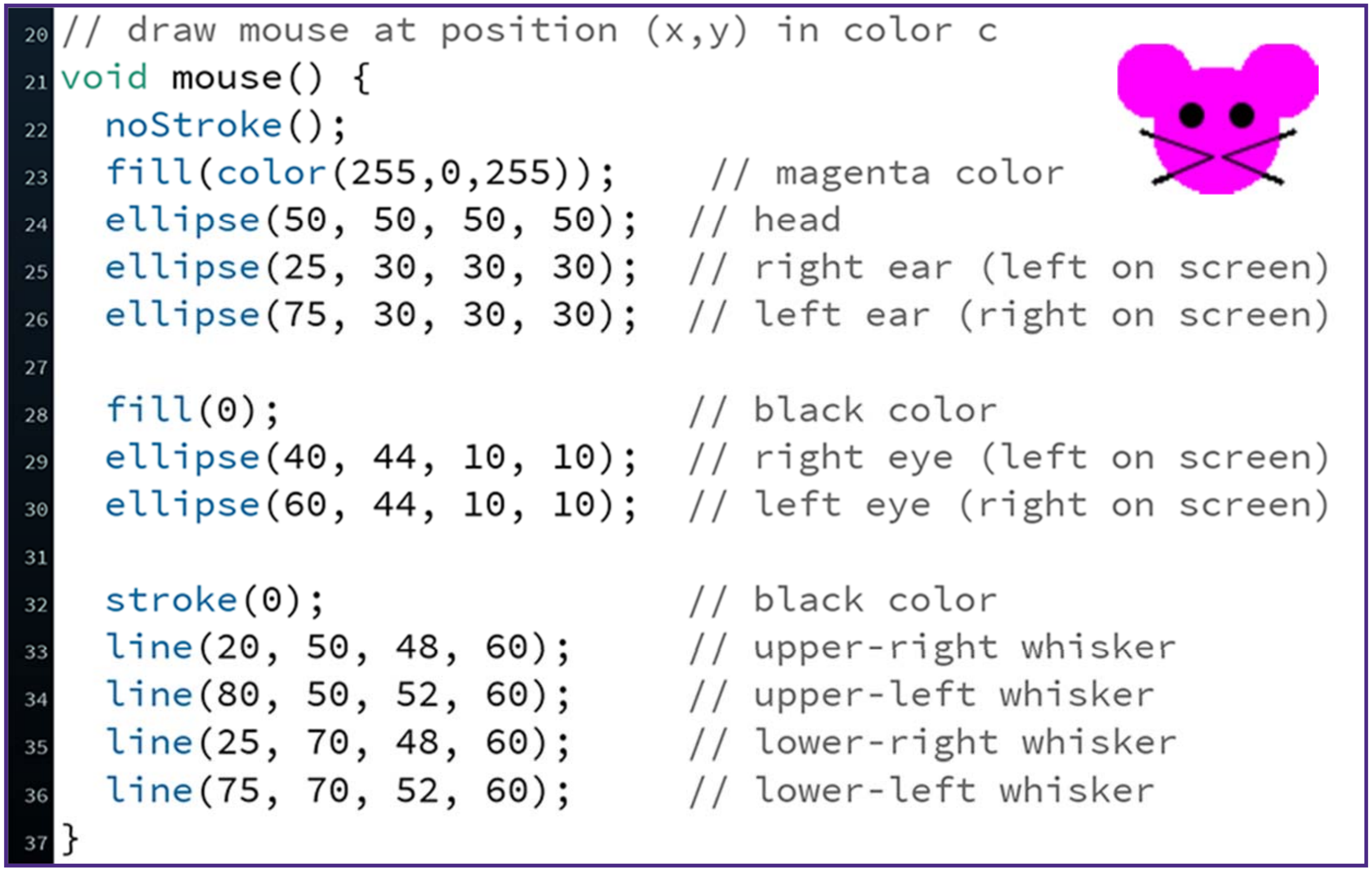
Question

- If you're dreaming and someone in your dream hands you a turnip, do you wake up with a turnip in your bed?

  A. Yes
  B. No
  C. I will report back on Monday

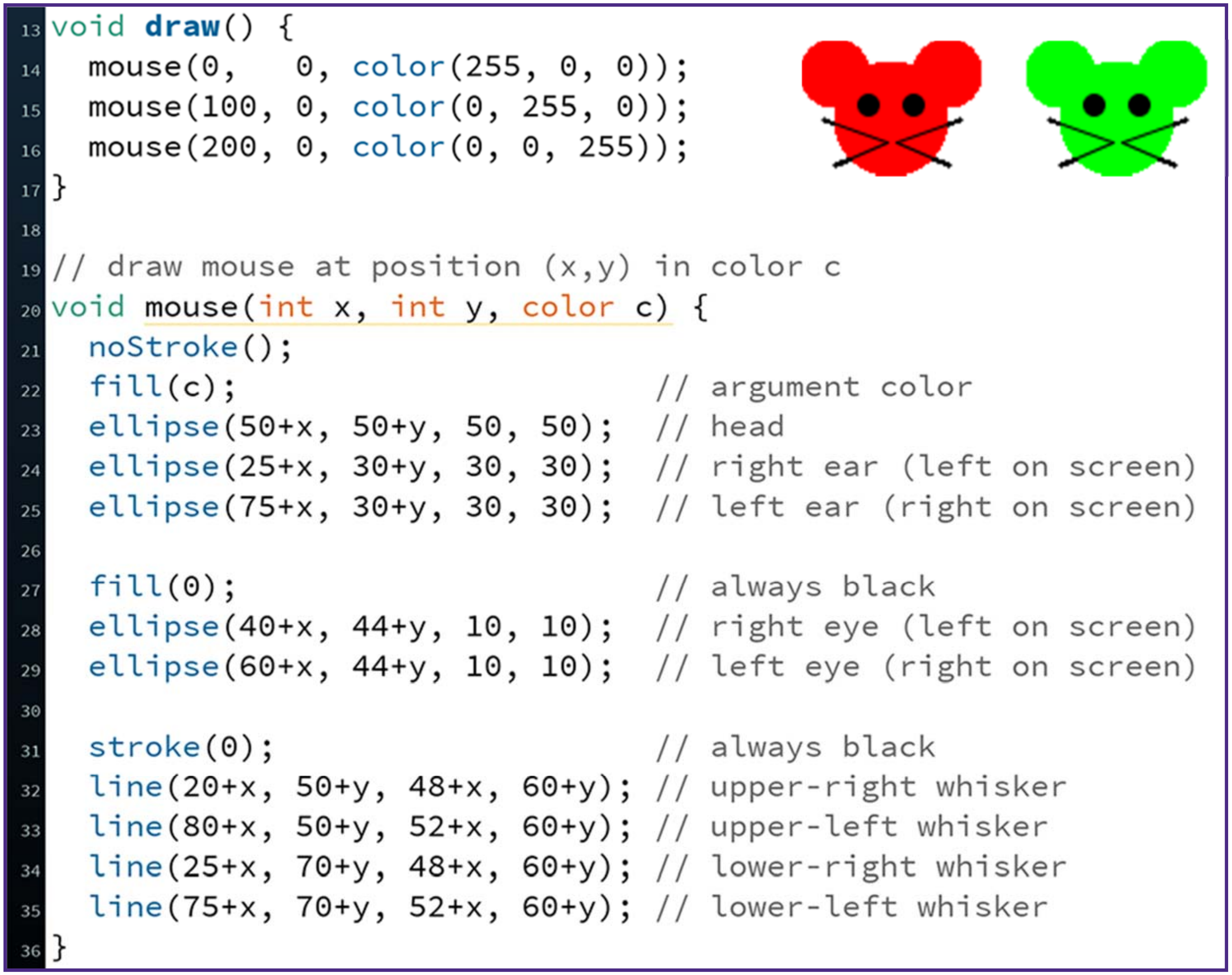
- Variable scope demo in Processing: [see Panopto]

Parameter Example

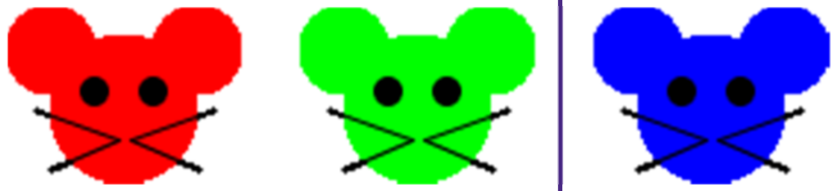


```
// draw mouse at position (x,y) in color c
void mouse() {
  noStroke();
  fill(color(255,0,255));    // magenta color
  ellipse(50, 50, 50, 50);  // head
  ellipse(25, 30, 30, 30);  // right ear (left on screen)
  ellipse(75, 30, 30, 30);  // left ear (right on screen)

  fill(0);                  // black color
  ellipse(40, 44, 10, 10);  // right eye (left on screen)
  ellipse(60, 44, 10, 10);  // left eye (right on screen)

  stroke(0);                // black color
  line(20, 50, 48, 60);     // upper-right whisker
  line(80, 50, 52, 60);     // upper-left whisker
  line(25, 70, 48, 60);     // lower-right whisker
  line(75, 70, 52, 60);     // lower-left whisker
}
```

Parameter Example



```
void draw() {
  mouse(0,   0, color(255, 0, 0));
  mouse(100, 0, color(0, 255, 0));
  mouse(200, 0, color(0, 0, 255));
}

// draw mouse at position (x,y) in color c
void mouse(int x, int y, color c) {
  noStroke();
  fill(c);                         // argument color
  ellipse(50+x, 50+y, 50, 50);    // head
  ellipse(25+x, 30+y, 30, 30);    // right ear (left on screen)
  ellipse(75+x, 30+y, 30, 30);    // left ear (right on screen)

  fill(0);                         // always black
  ellipse(40+x, 44+y, 10, 10);    // right eye (left on screen)
  ellipse(60+x, 44+y, 10, 10);    // left eye (right on screen)

  stroke(0);                       // always black
  line(20+x, 50+y, 48+x, 60+y);   // upper-right whisker
  line(80+x, 50+y, 52+x, 60+y);   // upper-left whisker
  line(25+x, 70+y, 48+x, 60+y);   // lower-right whisker
  line(75+x, 70+y, 52+x, 60+y);   // lower-left whisker
}
```



Solving Problems

- Understand the problem
  - What is the problem description?
  - What is specified and what is *un*specified?
  - What has been given to you (*e.g.* starter code)?
- Break the task down into less complex subtasks

- <u>Example</u>: Make a function that draws a row of five mice with their ears touching/overlapping. The mice should all be the same color except for the middle one, which should be red.

Looking Forward

- Lego Family
  - Design an abstracted family
  - Create functions for drawing each family member, including variables for position/movement
  - Have family members start at corners, then move into place
- Events
  - Introduce user interactions! Due Tuesday (4/11)
- Animal Functions
  - Start in lab on Tuesday, due Wednesday (4/12)
  - Design your own animal (like the mouse shown here)