Odds and Ends

- Please submit any images files you used along with your program
- Name your screenshot something very obvious – like "screenshot.jpg"
- Do not leave any files scattered in your Dropbox folder. It needs to be in an assignment folder or I won't know which assignment it belongs to!
- Name all your assignment folders well, like assignment01, sketch01, etc

Review

- Variable declarations
- Variable assignments
- Loops
  - Condition
  - index
- Functions
  - Definition
  - Call
  - Parameters

Execution

- Statements are executed one at a time in the order written
- Execution order
  - Globals and initializations
  - **`setup()`** called once
  - **`draw()`** called repeatedly (unless **`noLoop()`** is called in **`setup()`**)
  - If any mouse or keyboard events occur, the corresponding functions are called between calls to **`draw()`** – exact timing can not be guaranteed.

Parameterizing a shape

- Have code that draws something with a bunch of coordinates
- Want to draw the same thing anywhere, in any size and repeat any number of times
- How is a shape defined?
  - a reference point (center, corner)
  - a base size
- To move, scale and repeat
  - put code in a function
  - x and y increments
  - scaling factor

Example: any size and place door

- A door has
  - a plank
  - a handle
  - a window
  - hinges
  - a frame
- How do you move all parts together?
- When size changes:
  - how do you keep parts in same relative locations?
  - What happens when aspect ratio of sketch changes?

Let's design the door

- Function name?
  - parameters
- a plank
  - what is the reference point?
- a handle, etc…
  - what is it's location relative to?
  - what about its size?

Identify Similar Code

```
void drawRandomRect() {
  fill(random(255), random(255), random(255),
       50);
  x = random(width);
  y = random(height);
  w = random(5, 100);
  h = random(5, 100);
  rect(x, y, w, h);
  }
}
```

Similar unit

```
void drawRandomCircle() {
  fill(random(255), 50);
  x = random(width);
  y = random(height);
  w = random(5, 100);
  h = random(5, 100);
  ellipse(x, y, w, h);
}
```

Similar unit

7

manyShapesFunction2

```
float x, y, w, h;
int totalShapeCount = 1000;
int MAX_COL = 255, WHITE = 255;
int TRANSLUCENT = 50;
int BLACK = 0;
int RECT_CHOICE    = 1;
int ELLIPSE_CHOICE = 2;
int MIN_D = 5; int MAX_D = 100);

void setup () {
  int i = 0;
  // other setup code here …
  stroke(WHITE, TRANSLUCENT);
  while (i<totalShapeCount) {
    drawRandomShape(RECT_CHOICE);
    i += 1;
  }
  stroke(BLACK, TRANSLUCENT);
  for (i=0; i<totalShapeCount;
       i++) {
    drawRandomShape(ELLIPSE_CHOICE);
  }
}
```

```
void drawRandomShape(int choice)
{
  x = random(width);
  y = random(height);
  w = random(MIN_D, MAX_D);
  h = random(MIN_D, MAX_D);
  if(choice == ELLIPSE_CHOICE)
  {  // circle
    fill(random(WHITE),
         TRANSLUCENT);
    ellipse(x, y, w, h);
  }
  else { // RECT_CHOICE
    fill(random(MAX_COL),
         random(MAX_COL),
         random(MAX_COL),
         TRANSLUCENT);
    rect(x, y, w, h);
  }
}
```

8

Functions that return values

- The return value of a function is the output of a function.
- A function evaluates to its return value.
- Function must return a value whose type matches the function declaration.

```
return_type function_name(parameter_list) {
    statements;
    return value;
}
```

Example

- What is the value of **`result`** in each line?

```
void setup () {
  int result;
  result = A(2);
  result = B(1, 2);
  result = 10 + A(2);
  result = A(2) + B(1, 2);
  result = B(A(2), B(B(1, 2), A(2)));

}

int A(int x) {
  return x*2;
}

int B(int x, int y) {
  return x+y;
}
```

Variable Lifetime

- – Variables cannot be referenced before they are declared.
- A variable is created and initialized when a program enters the block in which it is declared.
  - Functions
  - Loops
  - Conditionals
  - Function parameters
- A variable is destroyed when a program exists the block in which it was declared.

Variable Scope

- The region of code in which a particular variable is accessible.
- To a first approximation, the scope of a section of your code is demarcated by { and }.
  - Functions
  - Loops
  - Conditionals
- A variable is only accessible/available within the scope in which it is declared.

Global variables

- Variables that are declared outside of any scope are considered globals (versus locals).
- Global variables should be declared at the top of your program.
- Do not sprinkle them between functions!

Shadowing

- When there is a name conflict between variables of different scopes

```
int x = 10;
void setup() {
  int x = 5;
  int y = x;
}
```

- The conflicting variables can not have different types (or it's considered a re-declaration and is not allowed)
- When shadowed, smaller (inner) scopes have precedence over larger (outer) scopes