Variables & Datatypes

CSE 120 Winter 2020

Instructor: Sam Wolfson

Teaching Assistants: Yae Kubota, Eunia Lee, Erika Wolfe

Tesla hacking competition offers $1 million and free car if someone can hijack Model 3

San Francisco: Electric automaker Tesla has once again challenged hackers to find bugs in its connected cars.

The Elon Musk-run company is returning to the annual hackers' competition "Pwn20wn" to be held in Vancouver in March, reports electrek.

Some Model 3 cars and $1 million in award money will be up for grabs.

In March last year, a group of hackers won a Tesla Model 3 and $35,000 for hacking into its systems.

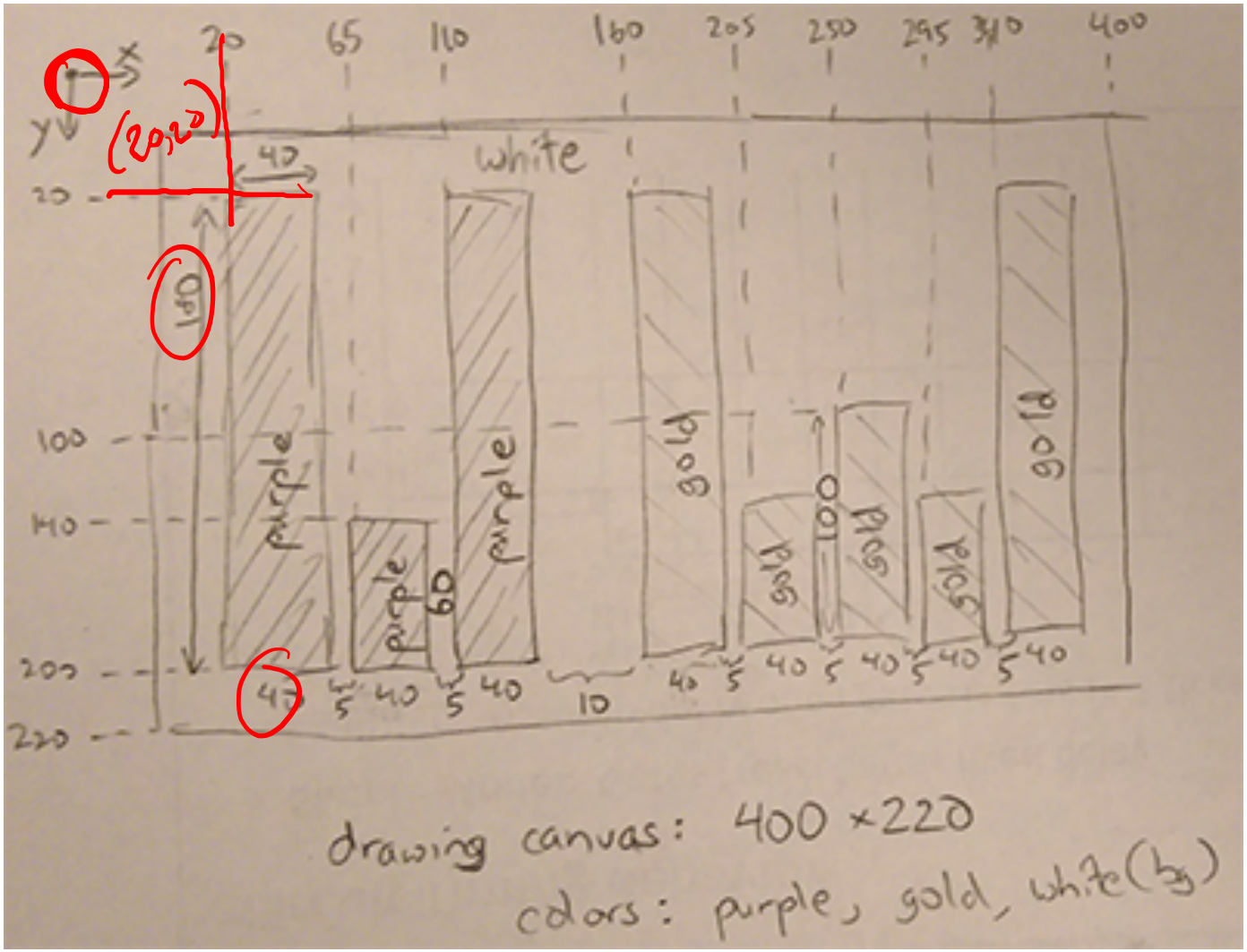
https://www.livemint.com/auto-news/tesla-hacking-competition-offers-1-million-and-free-car-if-someone-can-hijack-model-3-11578889743038.html

Administrivia

- Assignments
  - Taijitu [checkoff] due Thursday (1/16)
  - Reading Check 2 due Thursday by 3:30 pm (1/16)
  - Logo Design due **Monday** (1/20)

- Quiz Friday
  - Review packet posted on website

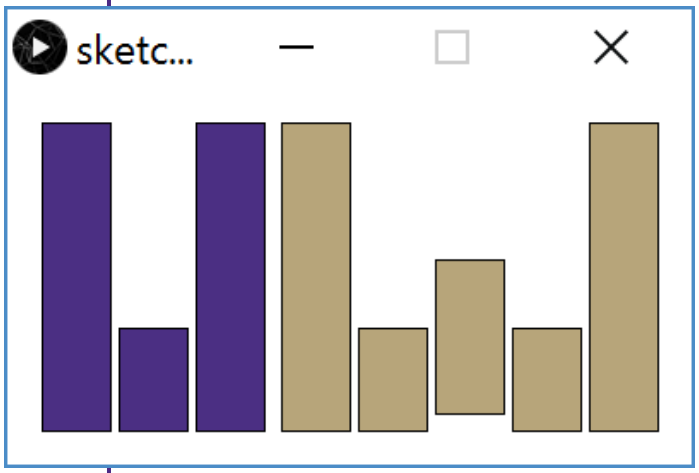
Homework: Logo Design



Homework: Logo Design

uw_logo

```
/* uw_logo.pde
   Created by Justin Hsia

   UW logo made out of rectangles in school colors.
*/

size(400,220);     // drawing canvas of 400x220
background(255);   // white background

// The letter 'U' in purple
fill( 75,  47, 131);       // purple fill
rect( 20,  20, 40, 180);   // left side of U
rect( 65, 140, 40,  60);   // middle base of U
rect(110,  20, 40, 180);   // right side of U

// The letter 'W' in gold
fill(183, 165, 122);       // gold fill
rect(160,  20, 40, 180);   // left segment of W
rect(205, 140, 40,  60);   // left base of W
rect(250, 100, 40,  90);   // middle segment of W
rect(295, 140, 40,  60);   // right base of W
rect(340,  20, 40, 180);   // right segment of W
```

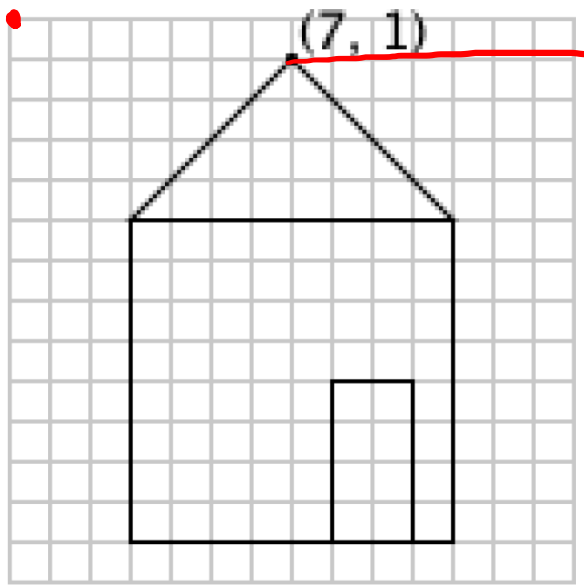


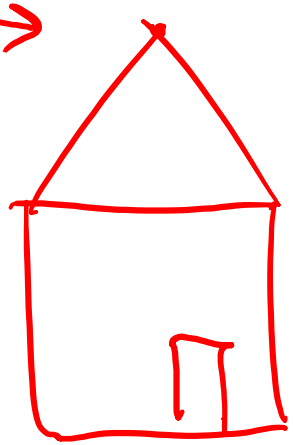
Drawing a House

- One solution from worksheet:

```
triangle(7, 1, 3, 5, 11, 5);
rect(3, 5, 8, 8);
rect(8, 9, 2, 4);
```

  - What if we wanted to move the house?





Variables

- Piece of your program that holds the value of something
  - Every variable must be given a *name* and a *data type*

identifier - "which box?"

"what's in the box?"

- The values of these variables can change (*i.e.* vary) during the execution of your program
  - Warning: Not like a variable in Algebra (*i.e.* an unknown)

int x;

- Assignment/Write: give a variable a specific value
  - *e.g.* `x ← 12;`

x 12

Variables

- Piece of your program that holds the value of something
  - Every variable must be given a *name* and a *data type*

- The values of these variables can change (*i.e.* vary) during the execution of your program
  - Warning: Not like a variable in Algebra (*i.e.* an unknown)

- Read: use the current value of a variable
  - *e.g.* `ellipse(x+1, 50, 20, 20);`

int x;

x=12;

x 12

13

ellipse centered at (13, 50)

Datatypes

- `int` integers (*e.g.* `12` or `-3`)
- `float` decimal/real numbers (*e.g.* `3.14`)
- `color` RGB (*e.g.* `color(0)`)
- `char` characters (*e.g.* `'J'`)
- `boolean` true or false (*e.g.* `true`)

- Many more exist and can be found in the Processing Reference:

Primitive
boolean
byte
char
color
double
float
int
long

Declarations

- We declare a variable by telling Processing the variable's datatype, followed by the variable's name:

```
int x;
float half;
color yellow;
```

x: 3 int

half: float

x = 3;

- You can also give a variable a starting value (initialization) in the same line as the declaration:

```
int x = 4;
float half = 0.5;
color yellow = color(255, 255, 0);
```

int x;
x = 4;

Variable Manipulation

- Executed sequentially, just like other statements
- For variable assignments, compute right-hand side *first*, then store result in variable
- <u>Example</u>:

```
int x = 4;
x = x + 1;
```

x 45

4 + 1

1) Read the current value of `x` (4) for the right-hand side
2) Add 1 to the current value of `x`
3) Store the result (5) back into `x`

Drawing a House with Variables

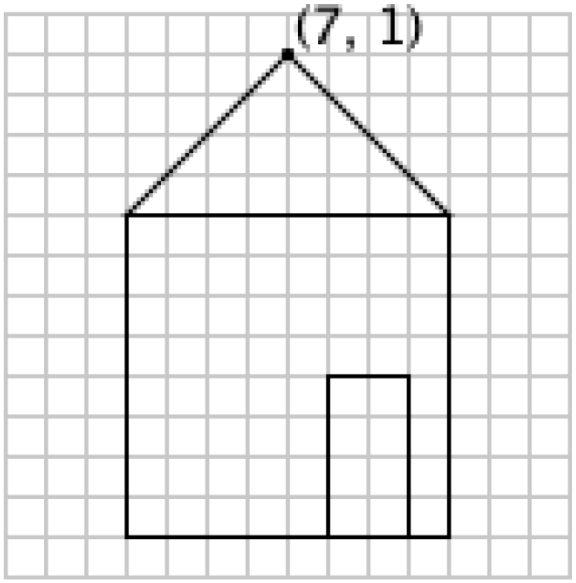
- Initial solution:

int x = 7;

```
triangle(7, 1, 3, 5, 11, 5);
rect(3, 5, 8, 8);
rect(8, 9, 2, 4);
```

x x-4 x+4 x-4 x+1

  - What properties might be useful to store in variables?



houseX
houseColor
houseY

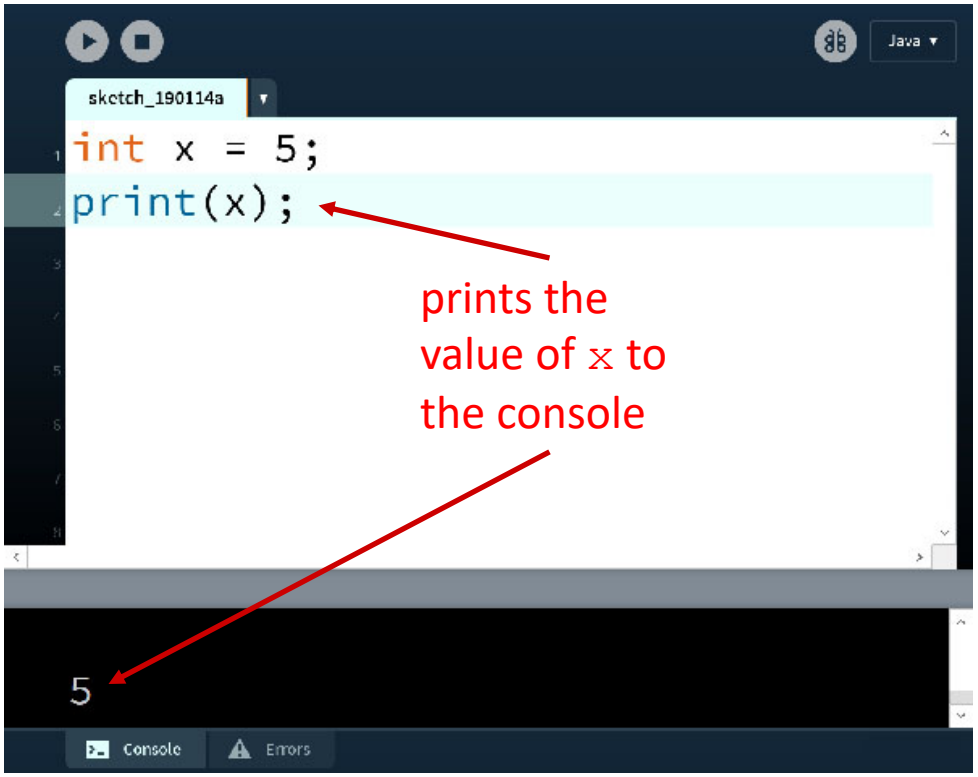
Variable Rules & Guidelines

- Variable naming rules:
  - Variables are case-sensitive (e.g. `myx` vs. `myX`)
  - Numbers allowed, but not at beginning (e.g. `k9` vs. `9k`)
  - Generally avoid symbols
- Variable names are meaningless to computers, but meaningful to humans
  - Choosing informative names improves readability and reduces confusion

    houseColor, houseY, etc.
  - In Processing, variables written in "`camelCase`"
- In this class, most of our variables will be declared and initialized at the very top of our programs

Variable Worksheet

- New functions: `print()`, `println()`



System Variables

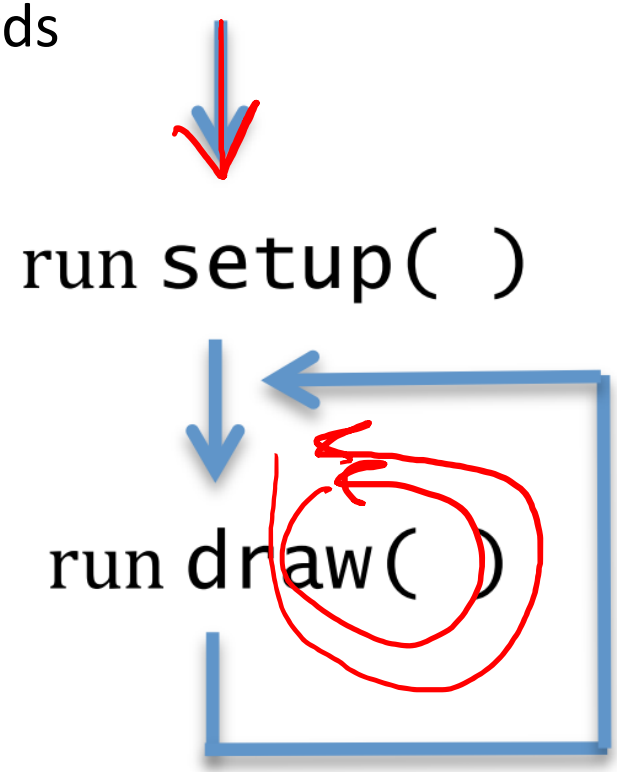
- Special variables that hold values related to the state of the program, often related to user input
  - You don't need to declare these variables
  - These variables will update automatically as the program runs
  - Colored **pink/magenta-ish** in the Processing environment

- <u>Examples</u>: `width` and `height` hold the value of the width and height of the drawing canvas, respectively

Active Mode in Processing

- We enter active mode by creating the functions `setup()` and `draw()` in our program
  - `setup()` automatically executes once at the start
  - `draw()` executes infinitely afterwards

- Each time `draw()` executes, it is called a new *frame*



Drawing and Frames

- System variable `frameCount` returns the number of frames since the start of the program
  - Starts at 0 in **`setup`**()

Setup

- `frameRate`() changes the desired number of frame updates there are ~~per~~ *second*

  Frame Count: 1 2 3

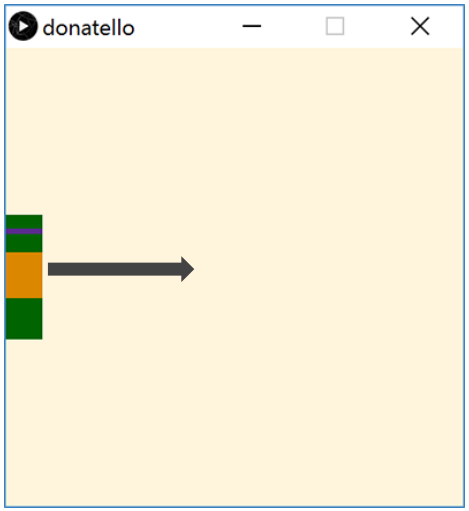
  - Larger argument is faster
  - Default is `frameRate`(60)

draw

- `noLoop`() stops **`draw`**() from being continuously executed
  - Can restart using `loop`()

Motion with Variables

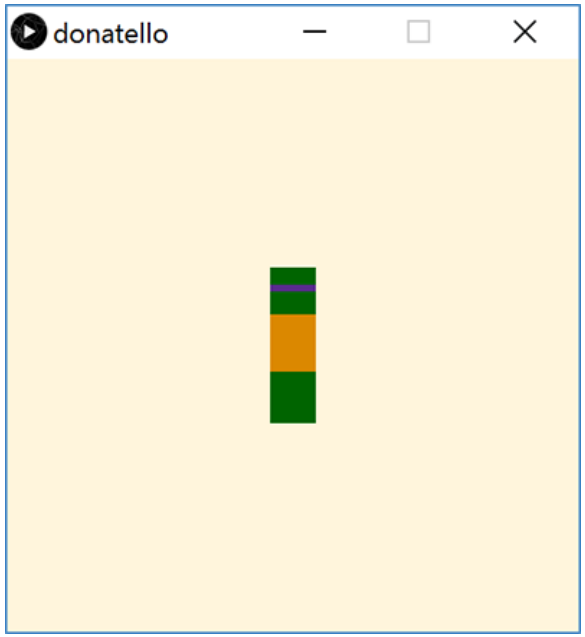
1) Create your drawing
2) Introduce a variable
3) Substitute values in your drawing with expressions that use the new variable
4) Change the variable value in-between frames
   - Use `background()` to cover old frames



TMNT: Donatello

donatello

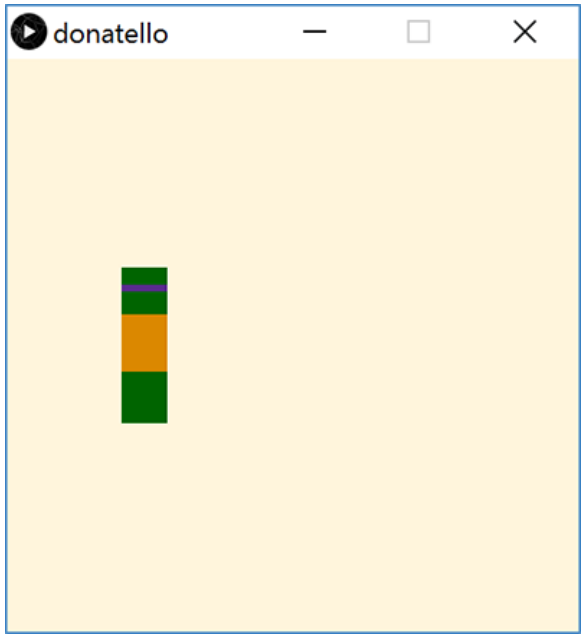
```
size(500,500);
noStroke();
background(255,245,220);

// Donatello
fill(0,100,0);          // dark green
rect(230,182,40,15);    // top of head

fill(88,44,141);        // purple
rect(230,197,40,6);     // bandana mask

fill(0,100,0);          // dark green
rect(230,203,40,20);    // bottom of head

fill(219,136,0);        // dark yellow
rect(230,223,40,50);    // shell

fill(0,100,0);          // dark green
rect(230,273,40,45);    // lower body
```



Donatello with a Variable

donatello

```
int xPos = 100;     // x-position

size(500, 500);
noStroke();
background(255, 245, 220);

// Donatello
fill(0, 100, 0);            // dark green
rect(xPos, 182, 40, 15);    // top of head

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

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

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

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



Stopping Motion

- Stop Donatello from running off the *right* side of the screen:

```
xPos = xPos + 1;
xPos = min(xPos + 1, width-40);
```

- Stop Donatello from running off the *left* side of the screen:

```
xPos = xPos - 1;
xPos = max(xPos - 1, 0);
```

Falling Into Place

- Introduce variables for each body segment:

```
int   head_pos = 0;     // head position
float mask_pos = 15;    // mask position
int   face_pos = 21;    // face position
float body_pos = 41;    // body position
int   leg_pos  = 91;    // leg  position
```

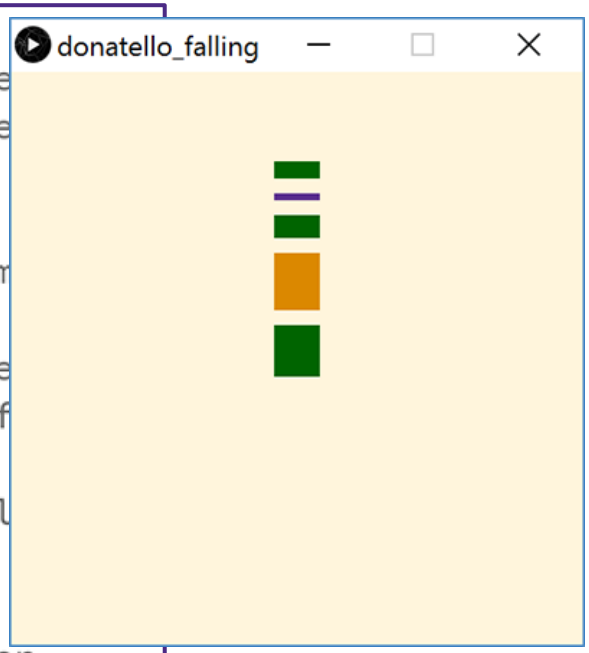
- Update each variable at different speeds:

```
  head_pos = min(head_pos + 3,   364);
  mask_pos = min(mask_pos + 3.5, 379);
  face_pos = min(face_pos + 4,   385);
  body_pos = min(body_pos + 4.5, 405);
  leg_pos  = min(leg_pos  + 5,   455);
```

Falling Into Place

- Update y-positions of drawing based on new variables:

```
// Donatello
fill(0,100,0);                  // dark gree
rect(x_pos,head_pos,40,15);     // top of he

fill(88,44,141);                // purple
rect(x_pos,mask_pos,40,6);      // bandana m

fill(0,100,0);                  // dark gree
rect(x_pos,face_pos,40,20);     // bottom of

fill(219,136,0);                // dark yell
rect(x_pos,body_pos,40,50);     // shell

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



Summary

- Variables are named quantities that can vary during the execution of a program
  - Datatypes specific different forms of data
    - *e.g.* `int`, `float`, `color`, `boolean`
  - Variable *declarations* specify a variable datatype and name to the program
    - Generally occur at top of program
- Active mode uses **`setup`**`()` and **`draw`**`()`
  - Motion can be introduced by changing the values of variables used in drawing commands in-between frames
- `min()` and `max()` functions can be used to limit or stop change in a variable value