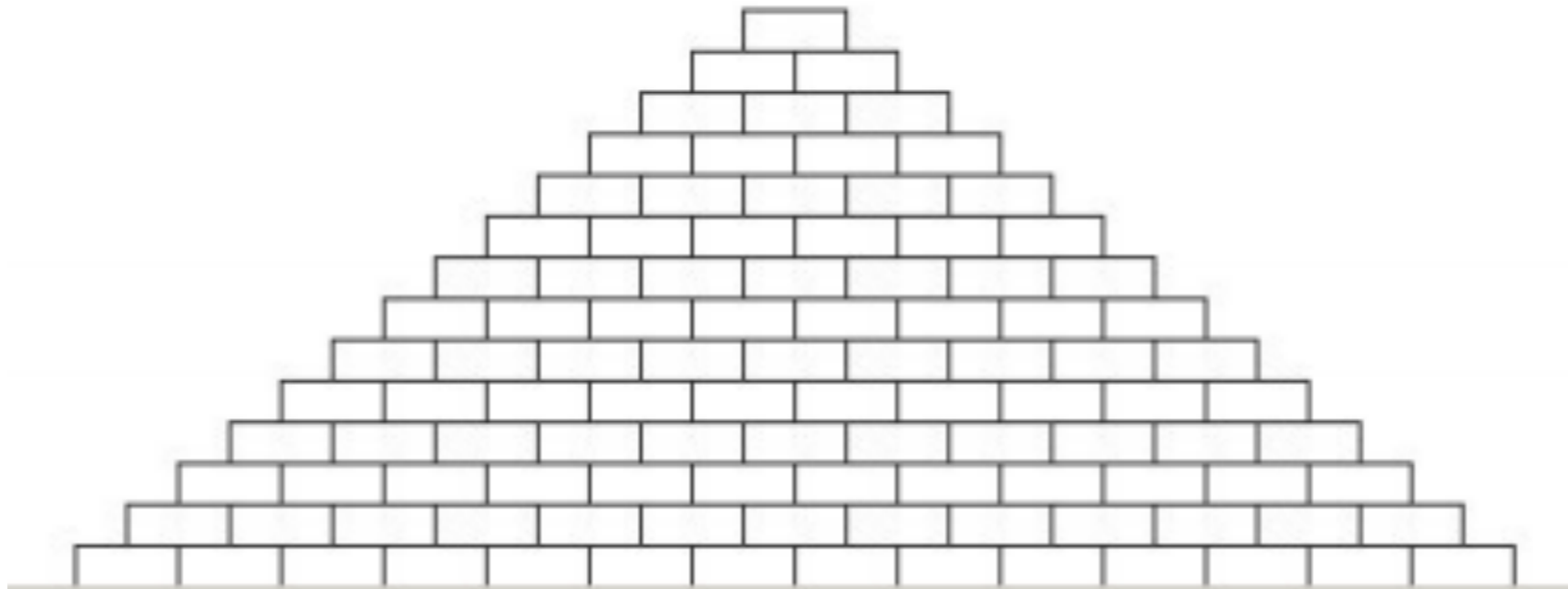
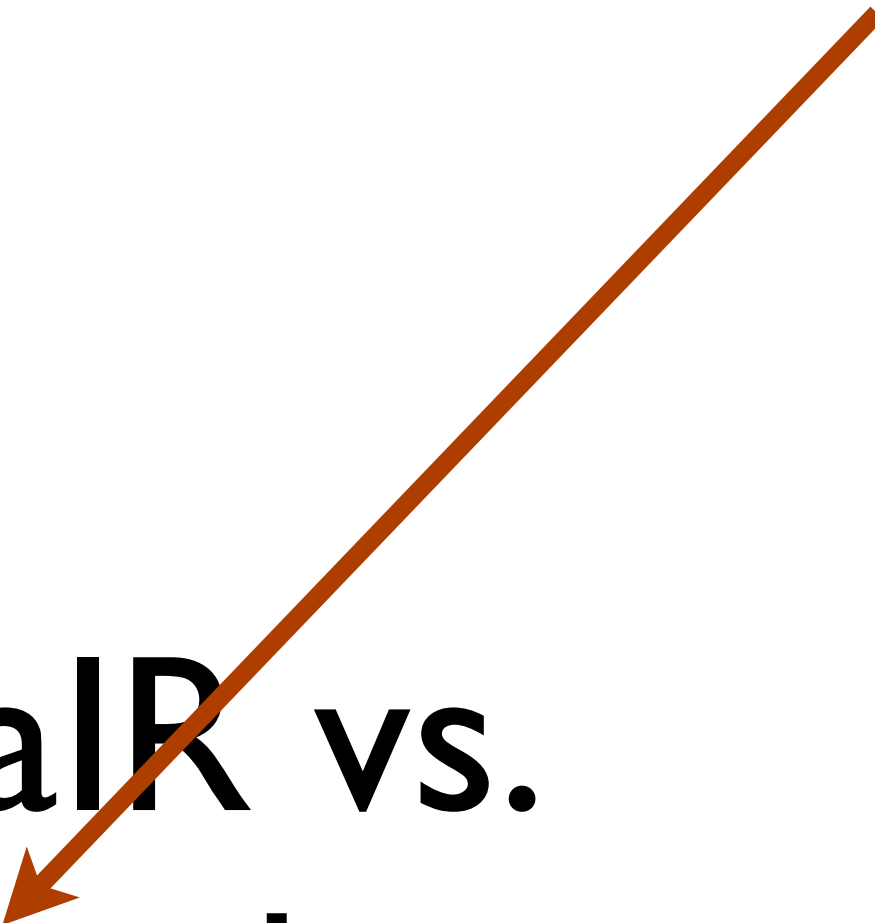


# YEAH session #2



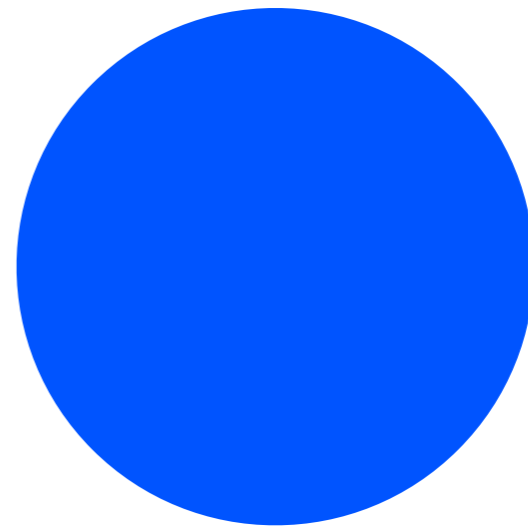
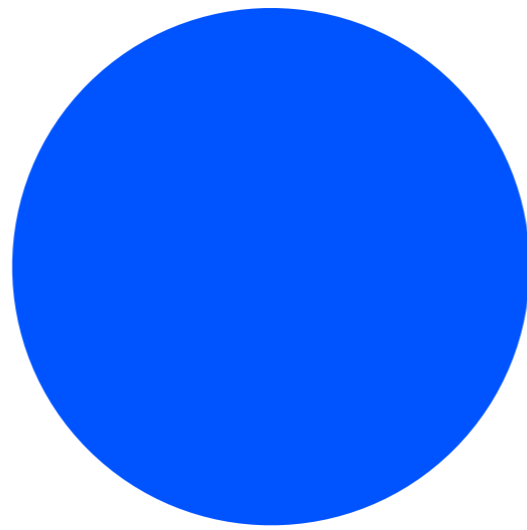
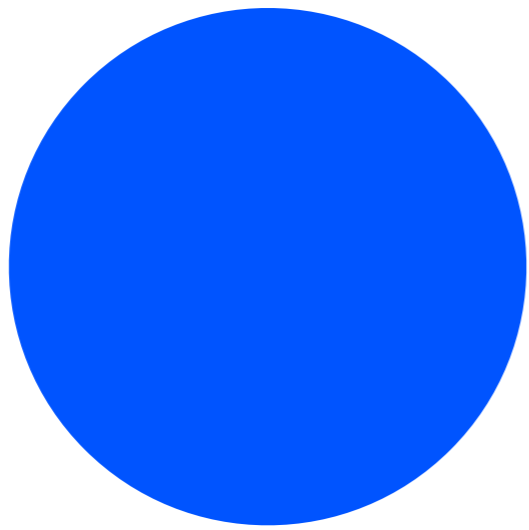
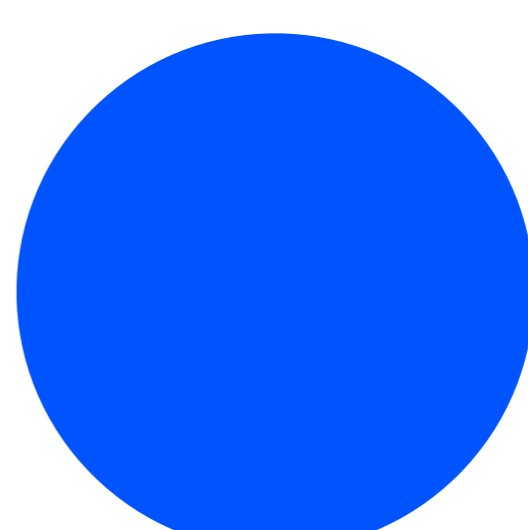
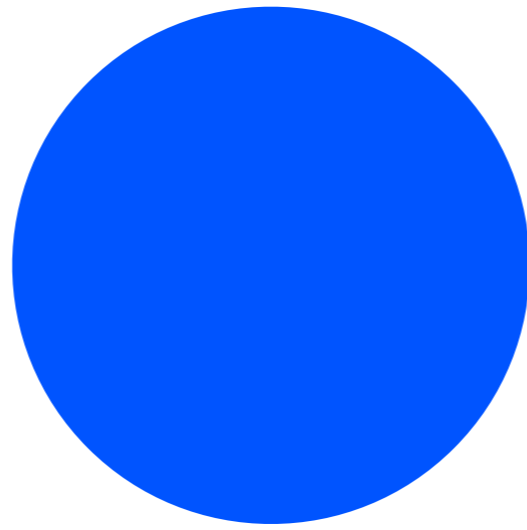
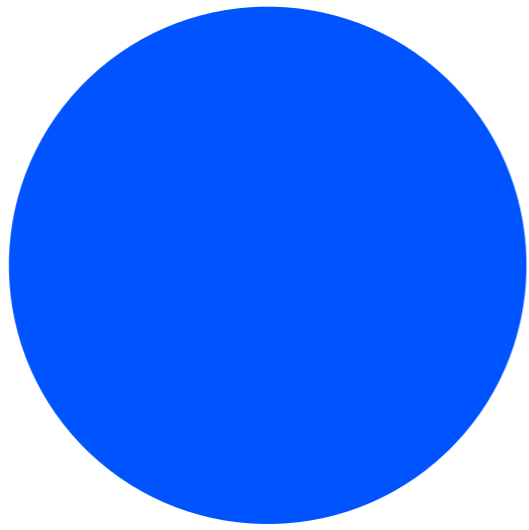
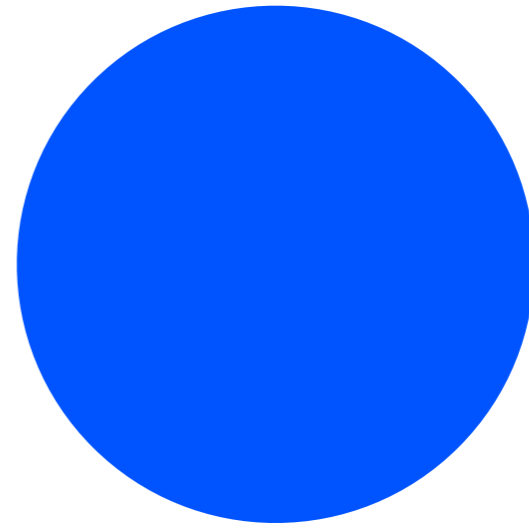
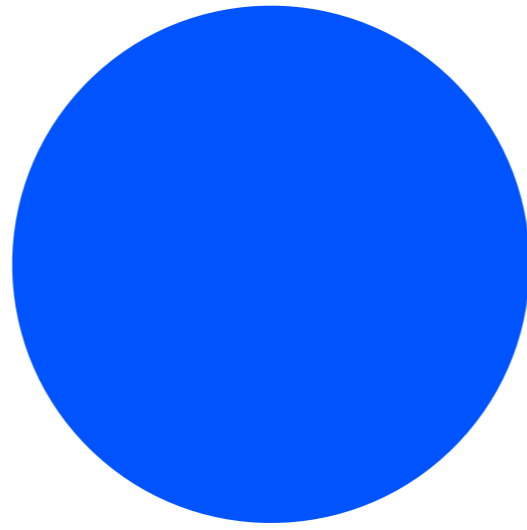
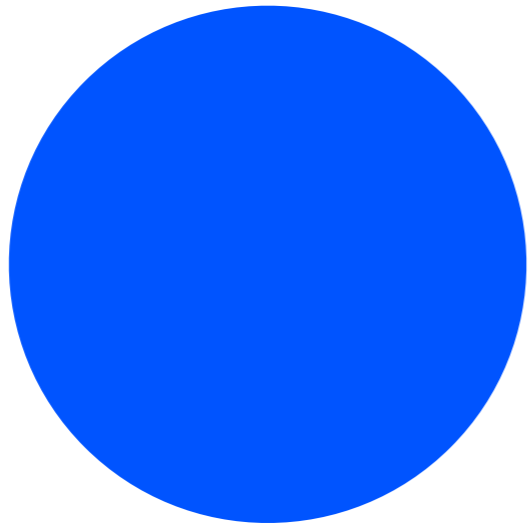
26 January 2014, 5p-6p  
Miles Seiver

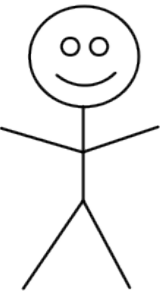
**YEAH vs. LaIR vs.  
section vs. office hours**

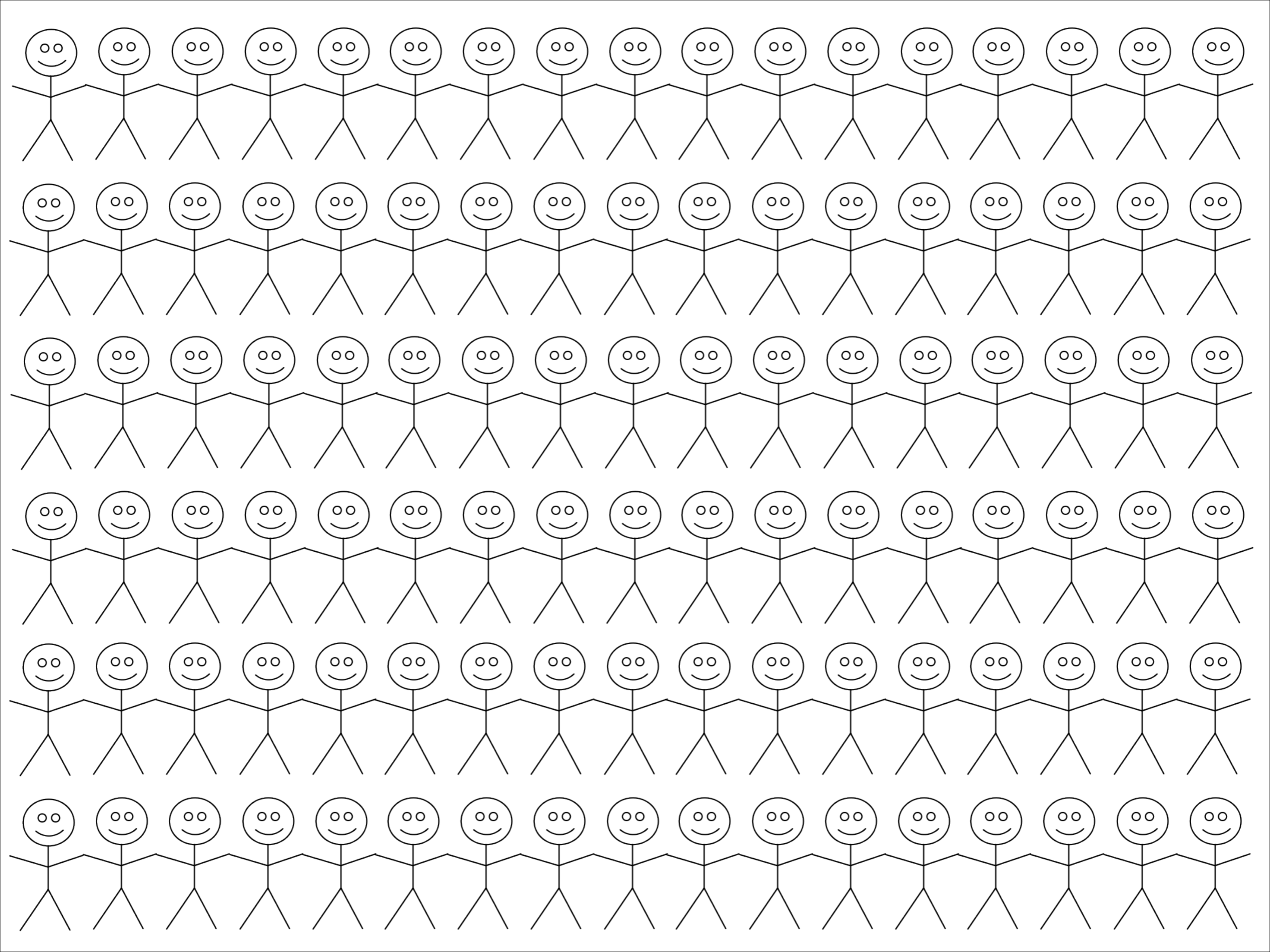


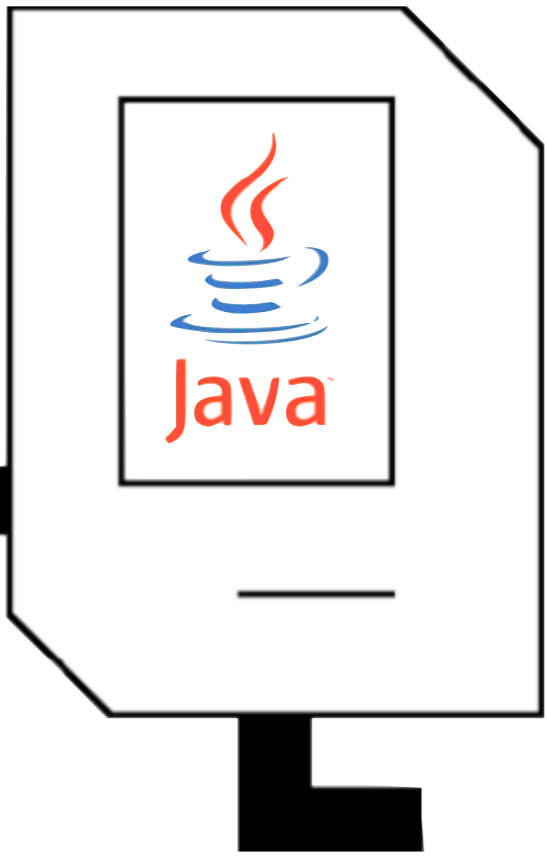
# 106A review session schedule

- YEAH 2 - now!
- YEAH 3 - 4 Feb 7:30pm at Braun Auditorium
- Future YEAH sessions to be scheduled soon
  
- Midterm 1 review session - 9 Feb 1p at Hewlett 200
- Midterm 2 review session - 2 Mar 1p at Hewlett 200









# Why Java?

- Graphics
- Console
- Data structures
- Networking



# Variable types

- `int`: Integers. (counting)
- `double`: Real numbers. (measuring)
- `boolean`: Logical `true` and `false`.
- `char`: Letters, numbers, and punctuation.

# Variable naming

- ~~value~~ numDots
- x
- y
- brickNumInRow
- ~~RESULT\_VALUE~~ sum
- ~~constant~~ NUM\_BRICKS\_IN\_ROW

# Constants

- Not all variables actually change; those that don't should be made into constants.
- **UPPERCASE\_WITH\_UNDERSCORES**

```
private static final type NAME_GOES_HERE = value;
```

# Control structures

# for versus while

```
for (init ; test ; step) {  
    statements  
}
```

- **for** loop used for *definite* iteration.
- Generally, we know how many times we want to iterate.

```
init  
while ( test ) {  
    statements  
    step  
}
```

- **while** loop used for *indefinite* iteration.
- Generally, don't know how many times to iterate beforehand.

```
while (true) {  
    /* ... get a value from the user ... */  
    if (condition)  
        break;  
  
    /* ... process the value ... */  
}
```

# Error-checking input

```
int n;

while (true) {
    n = readInt("Enter a positive integer: ");

    if (n > 0) {
        break;
    }

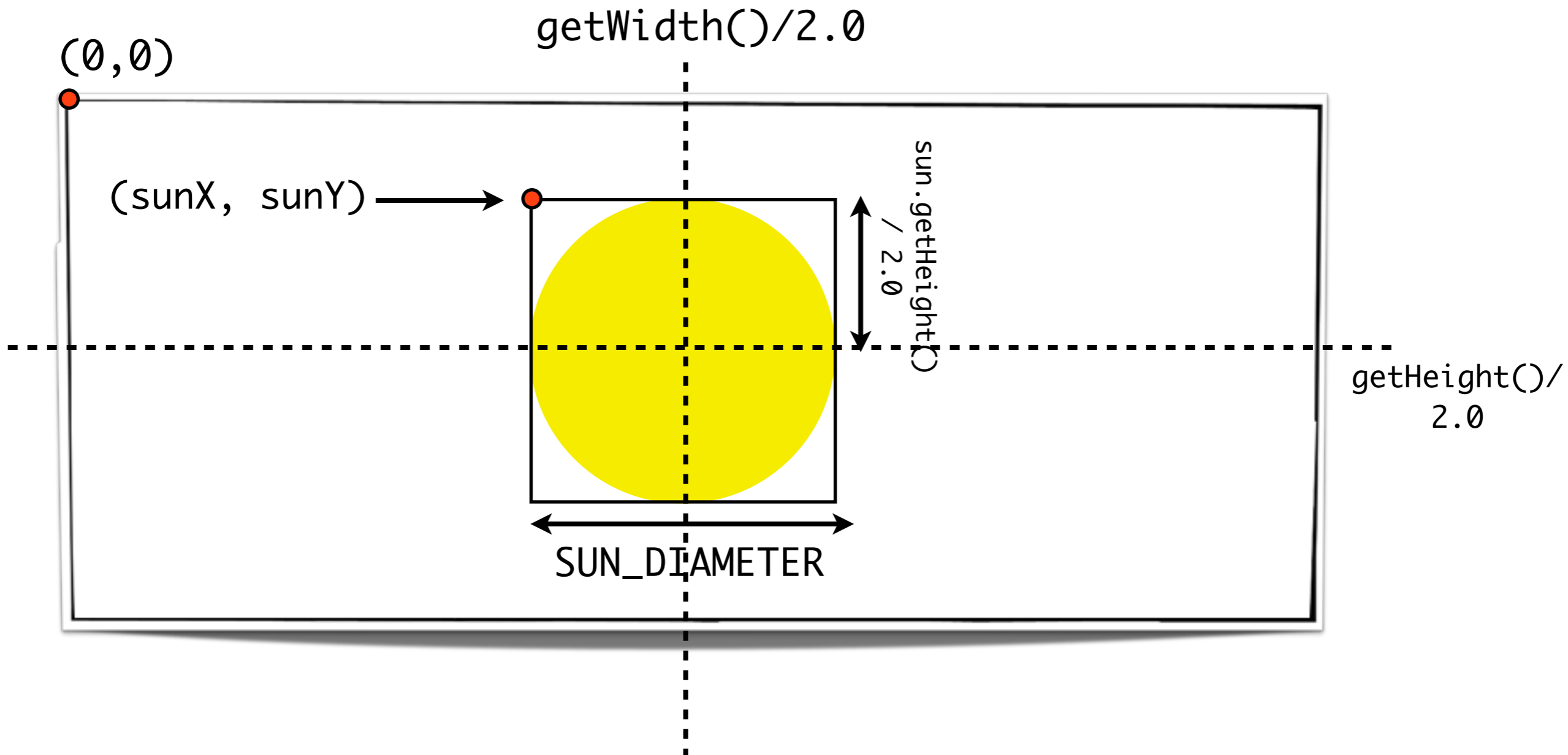
    println("Invalid input. Try again.");
}

// use n here (it's guaranteed positive)
```

# Graphics warm-up



```
private void drawSun() {
    G0val sun = new G0val(SUN_DIAMETER, SUN_DIAMETER);
    sun.setColor(Color.YELLOW);
    sun.setFilled(true);
    sun.setFill(Color.YELLOW);
    double sunX = getWidth()/2.0 - sun.getWidth()/2.0;
    double sunY = getHeight()/2.0 - sun.getHeight()/2.0;
    add(sun, sunX, sunY);
}
```



# Methods and parameters

```
public void run() {  
    println("This program finds the sum of  
           two numbers.");  
    int n1 = readInt("Enter n1: ");  
    int n2 = readInt("Enter n2: ");
```

```
    int total = addTwoNumbers(n1, n2);  
    println("The total is " + total + ".");  
}
```

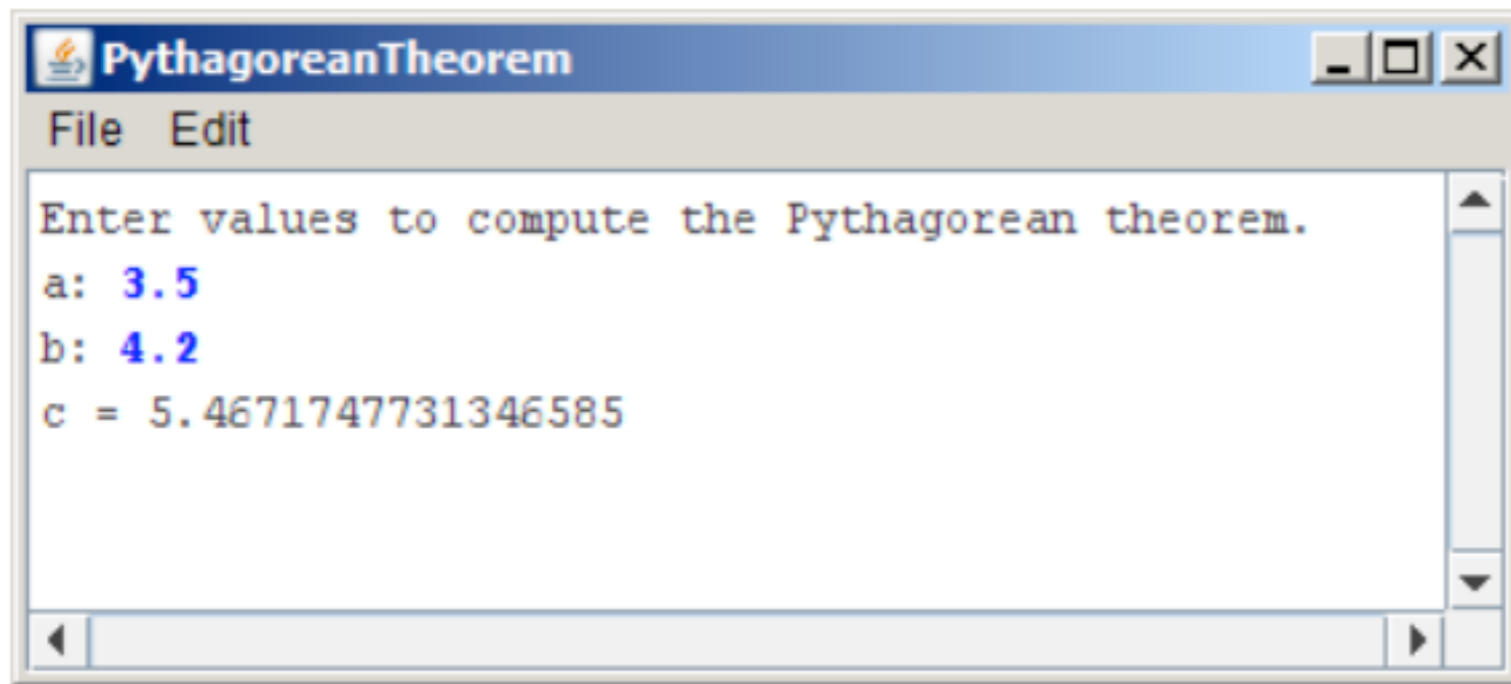
```
private int addTwoNumbers(int num1, int num2) {  
    int sum = num1 + num2;  
    return sum;  
}
```

**Let's dive in!**

# Welcome to Java!

- Due on Friday
- First three are console, last four are graphics
- No particular order of difficulty
- Key for style: use methods/parameters to decompose

# I. Pythagorean



```
PythagoreanTheorem
File Edit
Enter values to compute the Pythagorean theorem.
a: 3.5
b: 4.2
c = 5.4671747731346585
```

$$c = \sqrt{a^2 + b^2}$$

```
double y = Math.sqrt(x);
```

- Negative numbers are fine
- Make sure to use `double`
- `+` (addition)
- `-` (subtraction)
- `*` (multiplication)
- `/` (division)

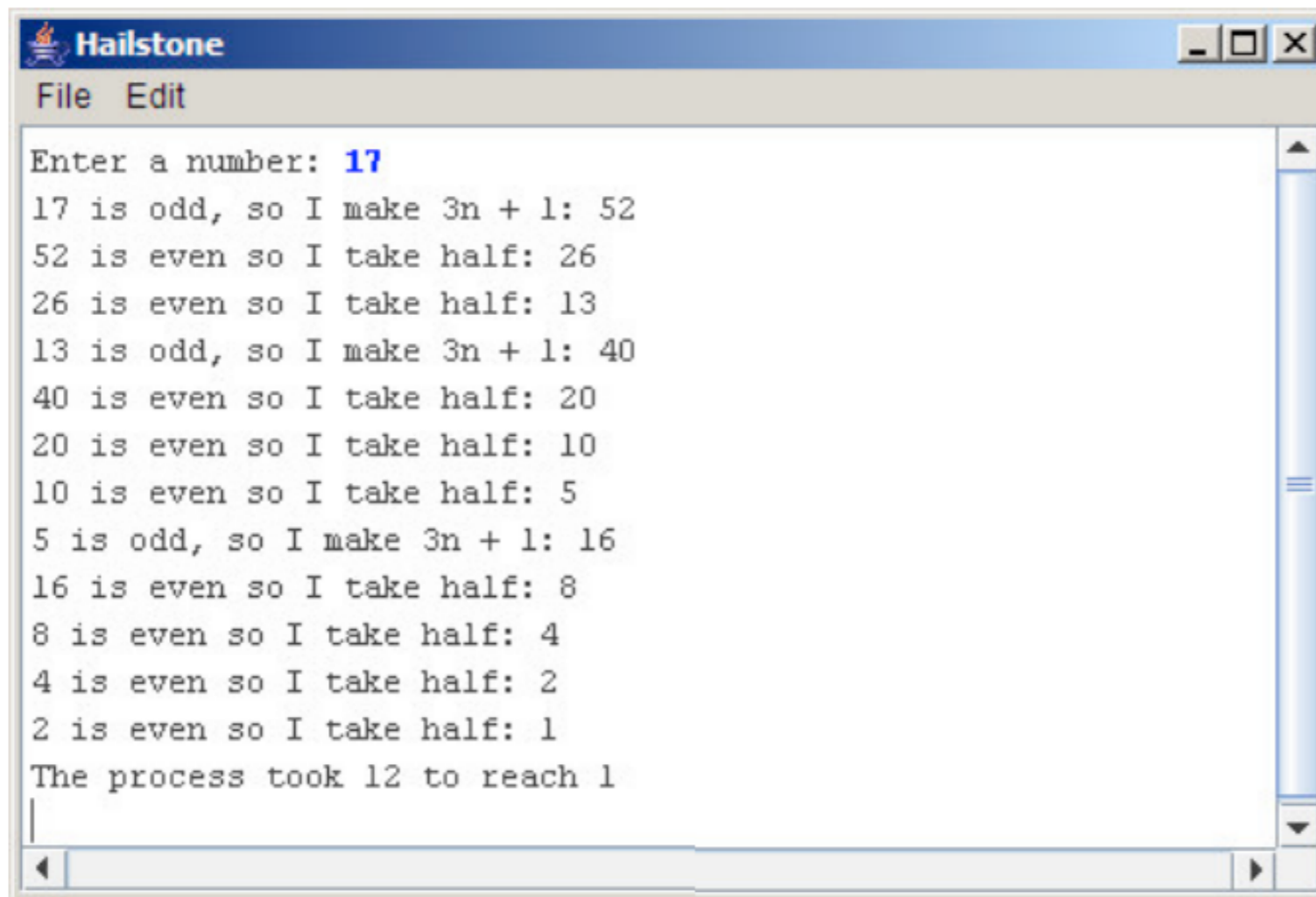
# 2. Hailstone

*Pick some positive integer and call it  $n$ .*

*If  $n$  is even, divide it by two.*

*If  $n$  is odd, multiply it by three and add one.*

*Continue this process until  $n$  is equal to one.*



```
Hailstone
File Edit
Enter a number: 17
17 is odd, so I make 3n + 1: 52
52 is even so I take half: 26
26 is even so I take half: 13
13 is odd, so I make 3n + 1: 40
40 is even so I take half: 20
20 is even so I take half: 10
10 is even so I take half: 5
5 is odd, so I make 3n + 1: 16
16 is even so I take half: 8
8 is even so I take half: 4
4 is even so I take half: 2
2 is even so I take half: 1
The process took 12 to reach 1
```

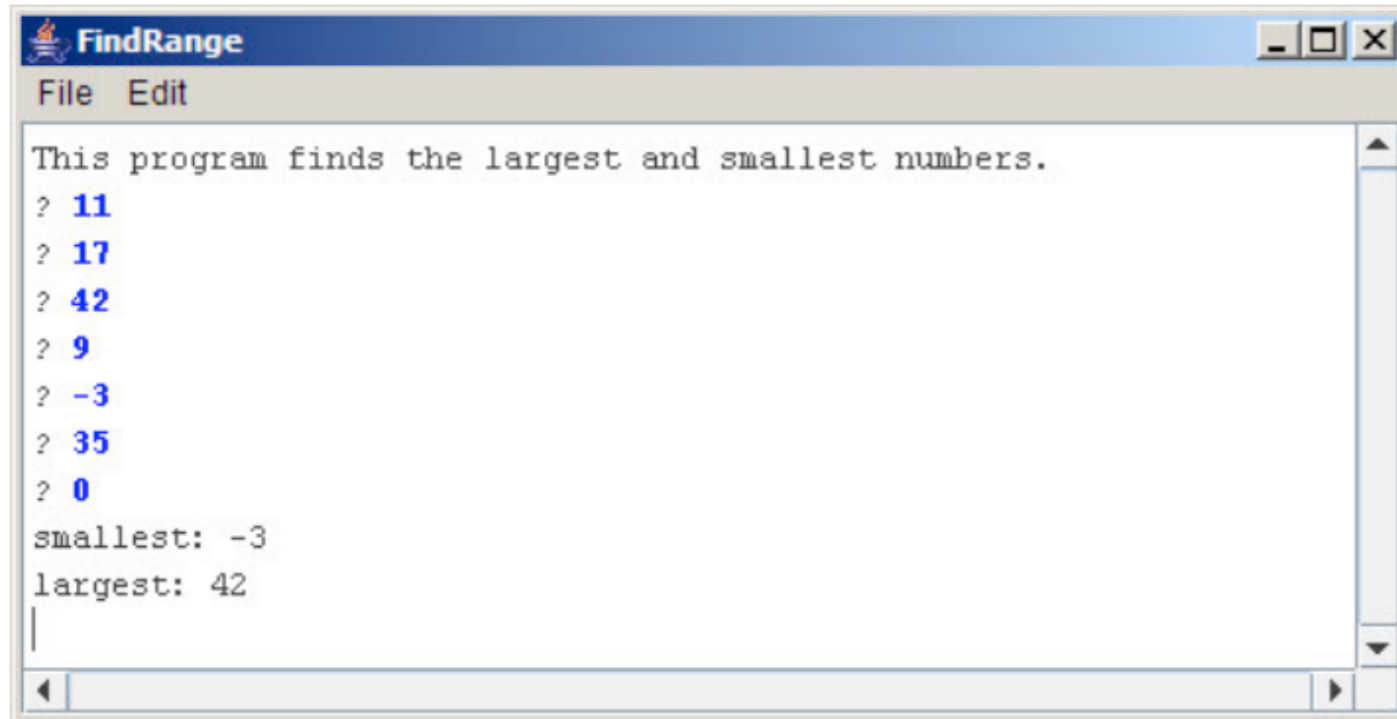
- Determining odd and even,
- Testing: “weird” numbers

# The Remainder Operator

- $a \% b$  is pronounced “a mod b.”
  - $15 \% 3 = 0$
  - $14 \% 8 = 6$
  - $21 \% 2 = 1$
  - $14 \% 17 = 14$



# 3. Find Range



```
FindRange
File Edit
This program finds the largest and smallest numbers.
? 11
? 17
? 42
? 9
? -3
? 35
? 0
smallest: -3
largest: 42
```

*If the user enters only one value before the sentinel, the program should report that value as both the largest and smallest.*

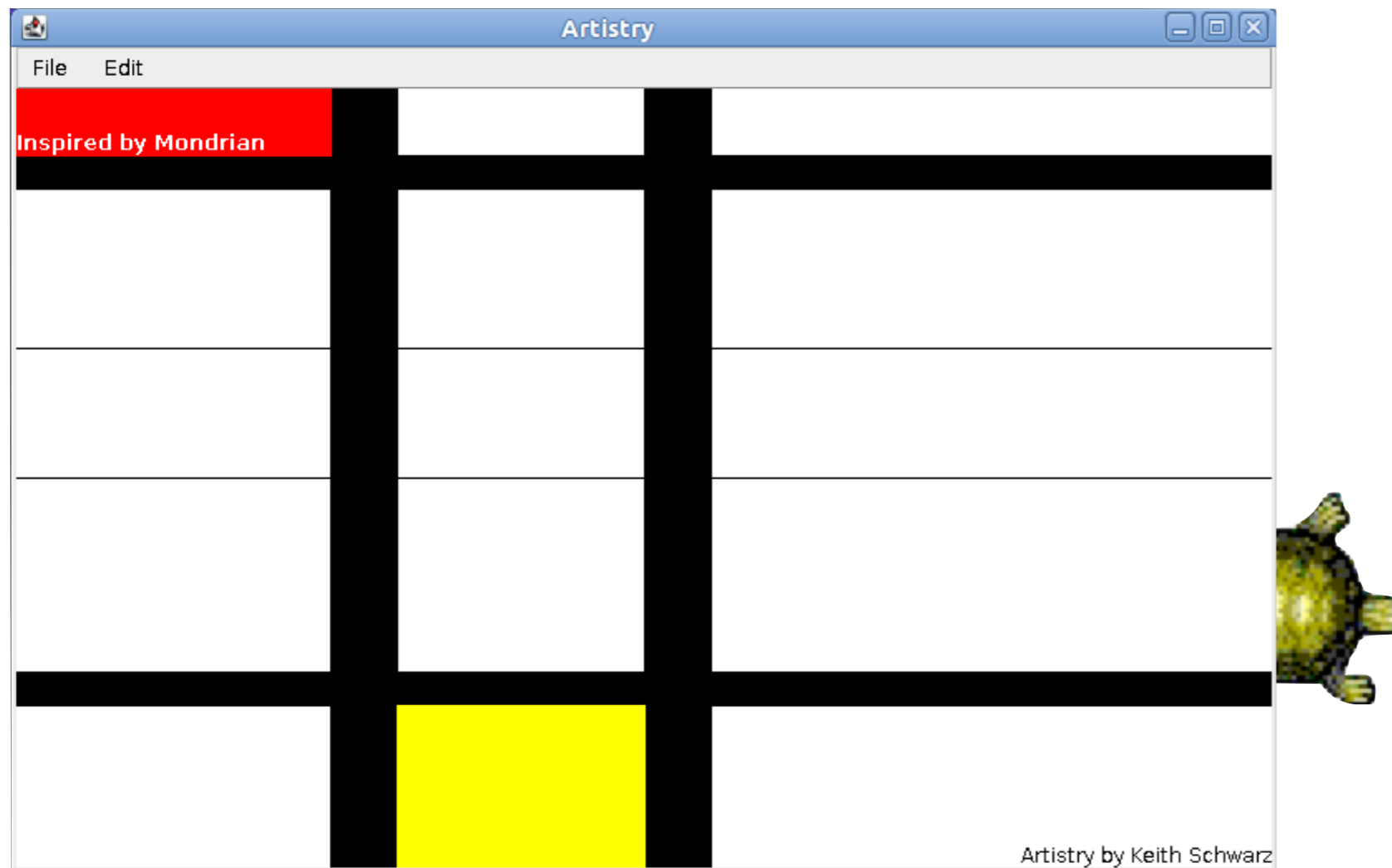
*If the user enters the sentinel on the very first input line, then no values have been entered, and your program should display a message to that effect. (from handout)*

- Use variables (what type?) to determine the min and max
- Handle the specified special cases
- The sentinel should be a constant
- Testing: One number, negative numbers, no numbers

# General graphics tips

- Draw pictures, many graphics problems are simple geometry in disguise
- Always use `double` when calculating coordinates

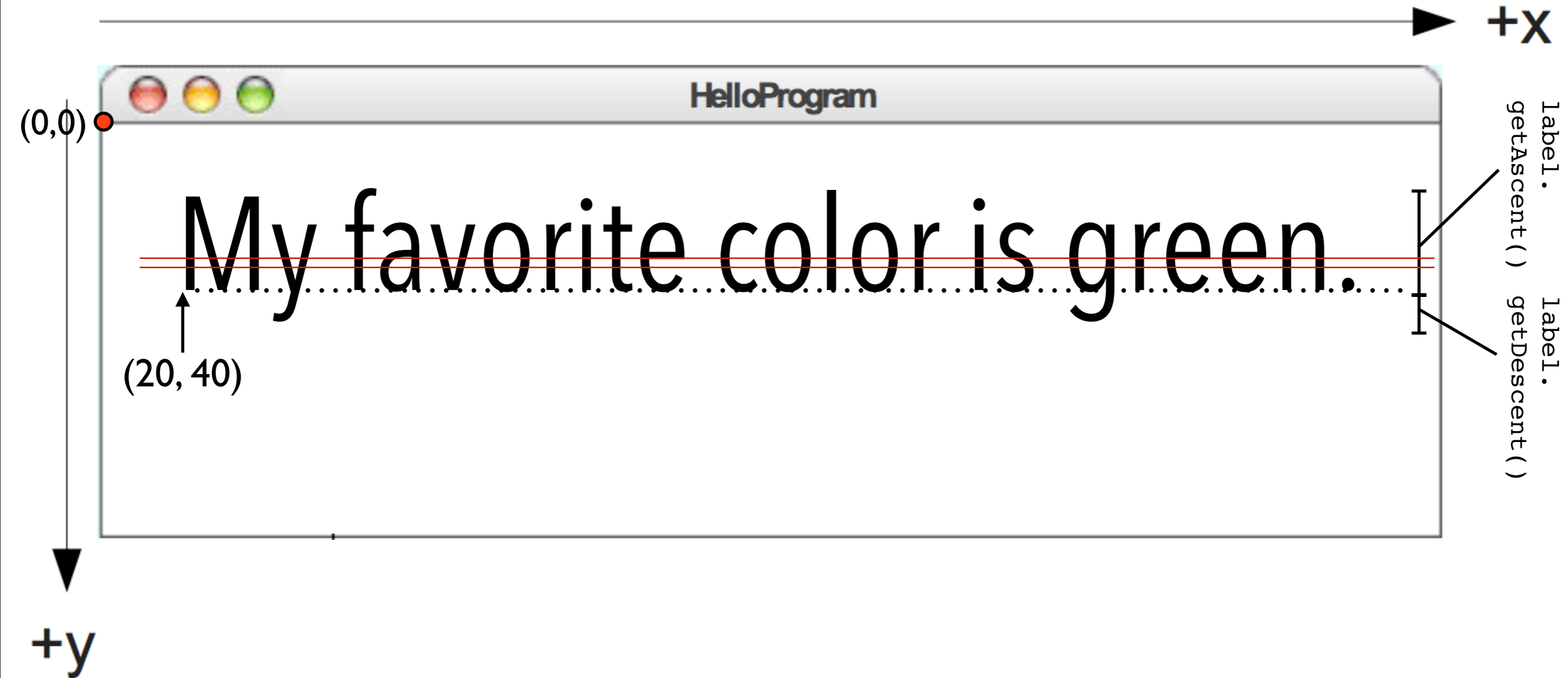
# 4. Artistry!



- ACM documentation
- Extensions: animation, custom color, `GTurtle`

1. Your picture must use *at least* three different types of **GObjects** – for example, you could use **GLine**, **GRect**, and **GOval**.
2. Your picture must have *at least* one filled object.
3. Your picture must have *at least* two different colors of objects.
4. You must sign your name in the bottom-right corner. To do this, create a **GLabel** with the text “Artistry by *name*,” where *name* is your name, and align it so that it is flush up against the bottom-right corner of the window. Be sure that all the text is visible and that none of the letters in the **GLabel** are cut off. (This **GLabel** doesn't count as one of the three different types of **GObjects** that you're required to have. If you want to count **GLabel** as one of the **GObject** types you're using, you'll need to have a second **GLabel** in your picture).

# GLabel is special



```
GLabel label = new GLabel("My favorite color is Green.")  
add(label, 20, 40);
```

# 5. Target

*The outer circle should have a radius of one inch (72 pixels), the white circle has a radius of 0.65 inches, and the inner red circle has a radius of 0.3 inches. (from handout)*



- What is actually changing between each circle?
- Decompose the problem so you don't copy and paste code
- Testing: try changing the given sizes

# 6. Fixing Broken Java

```
FixingBrokenJava.java ✕
* File: FixingBrokenJava.java
public class FixingBrokenJava extends ConsoleProgram {
    /* Reads a number from the user and reports whether or not it
    * is prime.
    */
    public void run() {
        /* Get the value from the user. */
        int value = readInput();

        /* Check whether or not it is prime. */
        if (isPrime(value)) {
            println(value + " is prime.")
        } else {
            println(value + " is composite.");
        }
    }

    /**
    * Given a positive integer, returns whether that integer is
    * prime.
    *
    * @param value The value to test.
    * @return Whether or not it is prime.
    */
    private boolean isPrime(int value) {
        /* Try all possible divisors of the number. If any of them
        * cleanly divide the number, we return that the number is
        * composite.
        */
        for (int divisor = 2; divisor <= value; divisor++) {
            if (value % divisor == 0) {
                return false;
            }
        }
    }

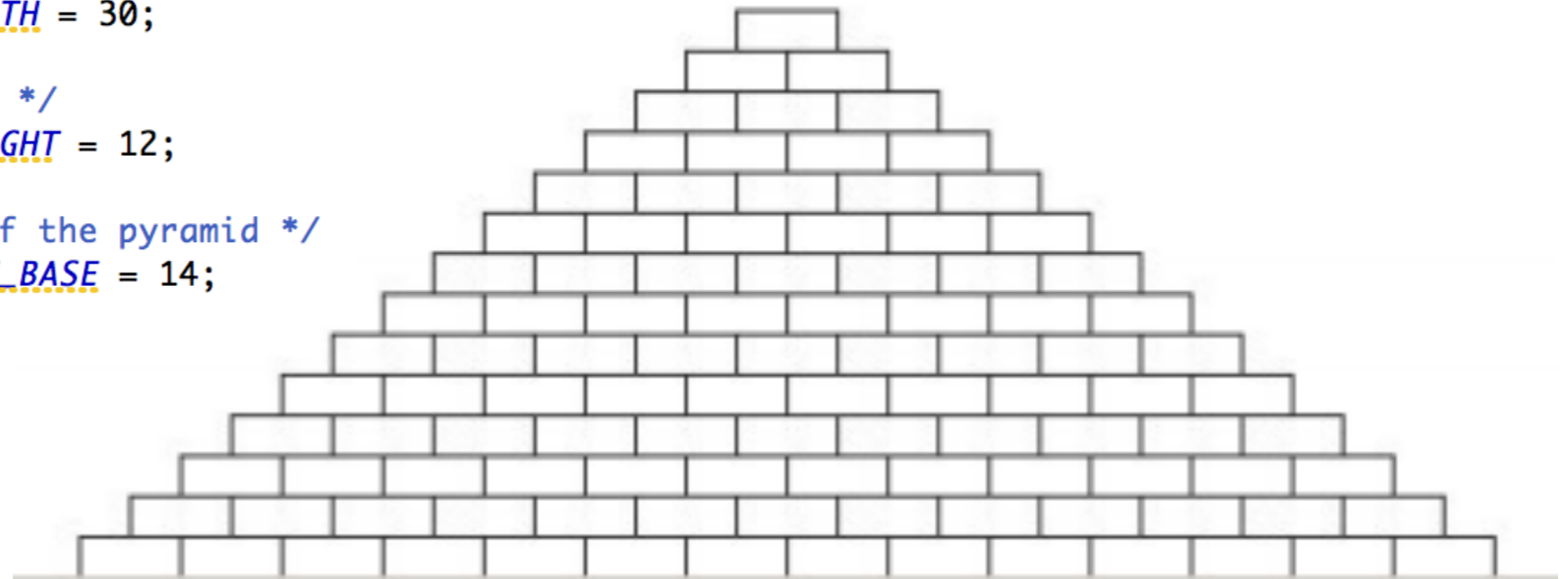
    /**
    * Reads an integer greater than one from the user.
    */
}
```

Read an integer greater than one from the user and check whether that integer is prime (whether its only divisors are 1 and itself).

If the number is prime, it prints a message saying that the number is prime; otherwise it says that the number is composite.

# 7. Pyramid

```
/** Width of each brick in pixels */  
private static final int BRICK_WIDTH = 30;  
  
/** Height of each brick in pixels */  
private static final int BRICK_HEIGHT = 12;  
  
/** Number of bricks in the base of the pyramid */  
private static final int BRICKS_IN_BASE = 14;
```



- Try looking below the window
- Testing: try changing the given constants
- Extensions?

- Follow the specifications carefully
- Comment
- Go to the LalR if you get stuck
- **Incorporate IG feedback!**
  
- Have fun!