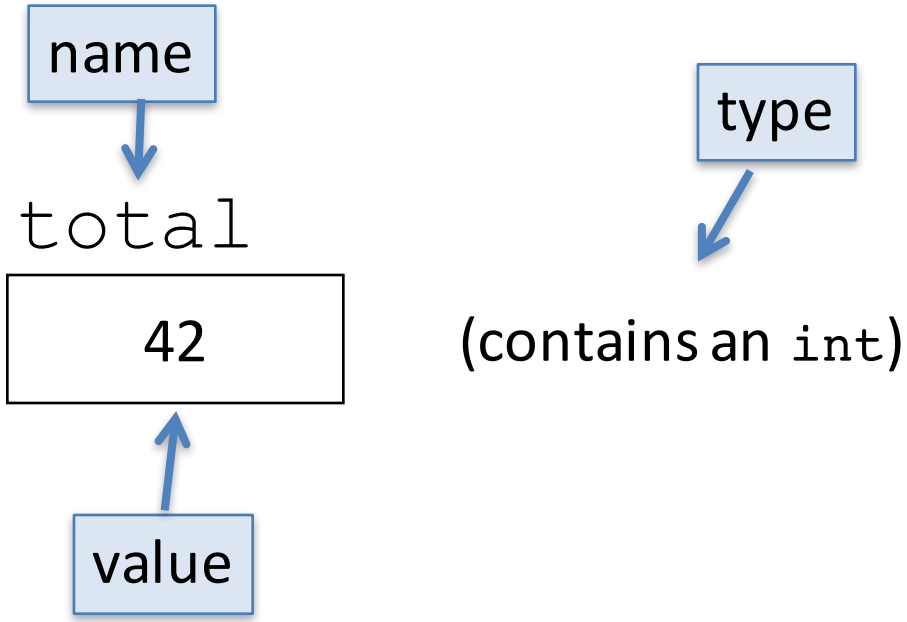
Review

Java


LELAND STANFORD JUNIOR UNIVERSITY
DIE LUFT DER FREIHEIT WEHT
1891

Three Properties of Variables



Types

```
// integer values
int num = 5;

// real values
double fraction = 0.2;

// letters
char letter = 'c';

// true or false
boolean isLove = true;
```

* Why is it called a double?

Binary Operators

+	Addition	*	Multiplication
–	Subtraction	/	Division
		%	Remainder

Today is your day, tio

Remainder

```
// an example of the % operator
println(17 % 4);

// reads a number from the user
int num = readInt("?:");

// stores the ones digit
int onesDigit = num % 10;

// equal to 1 if num is odd,
// 0 if num is even.
int isOdd = num % 2;
```


Binary Operators

+	Addition	*	Multiplication
–	Subtraction	/	Division
		%	Remainder

Challenge Carbon Dating

Write a program that can turn a measurement of C14 into an estimate of age.

CarbonDating

```
Radioactive molecule = C14
Halflife = 5730 years
C14 in living organisms = 13.6 dpm
-----

What is the amount of C14 remaining in your sample: 10.2
Your sample is 2378.0 years old.
```

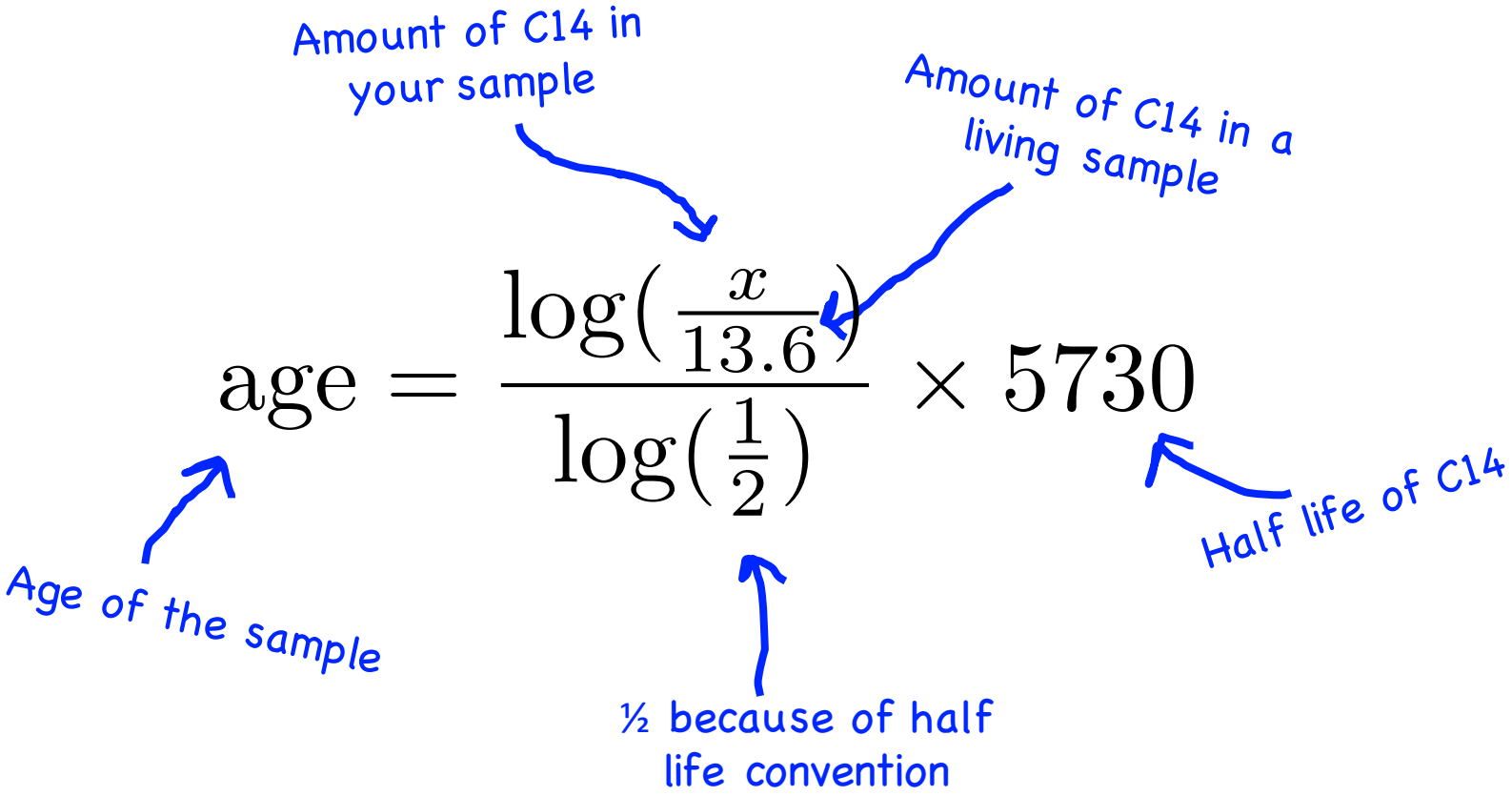
Example: Carbon Dating

C14 = 1.2 dpm

C14 = 13.6 dpm

We can calculate that the raptor died 20,000 years ago

Carbon Dating Equation



* Some of these values are constants

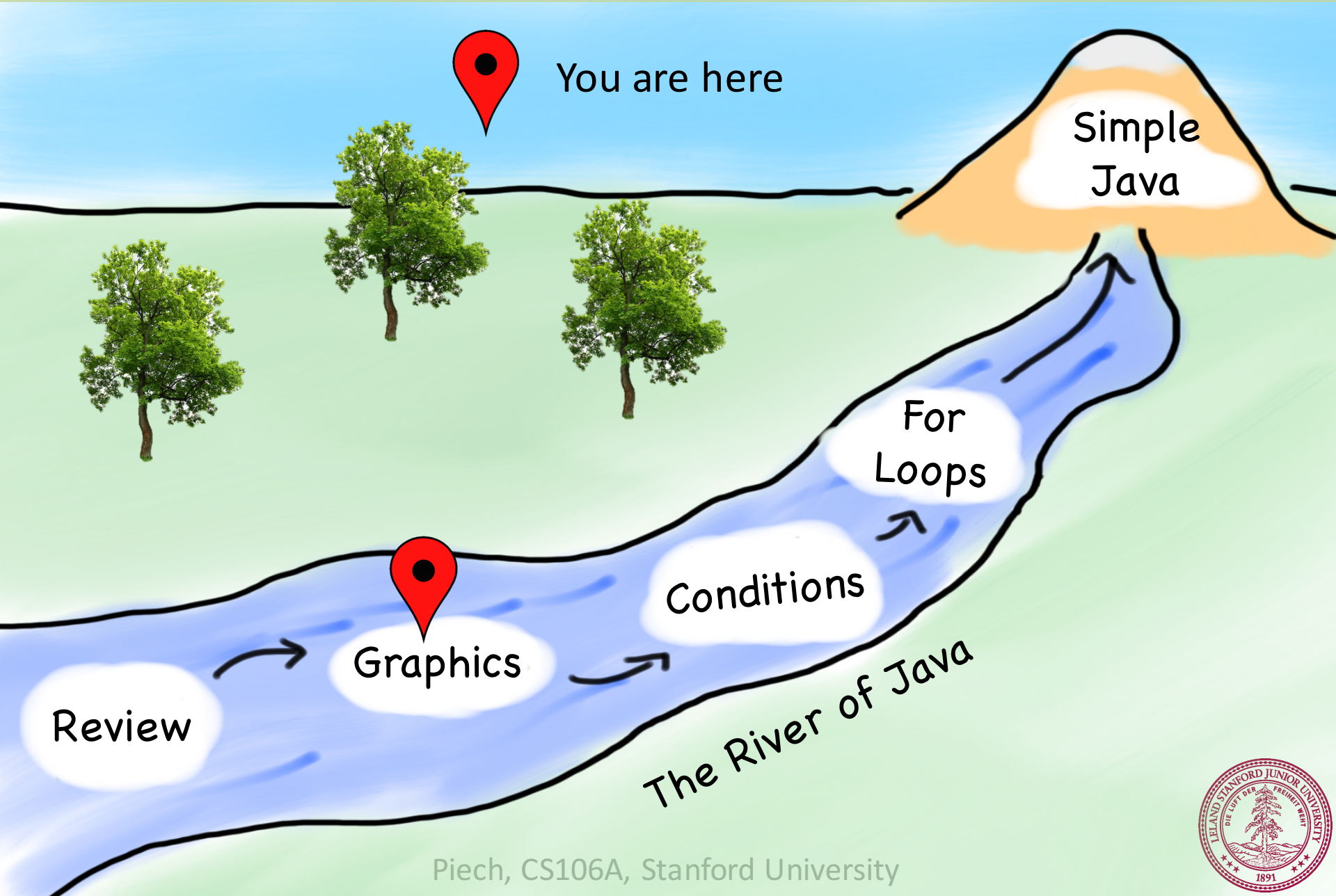
** Use the function: `Math.log( num )`

End Review

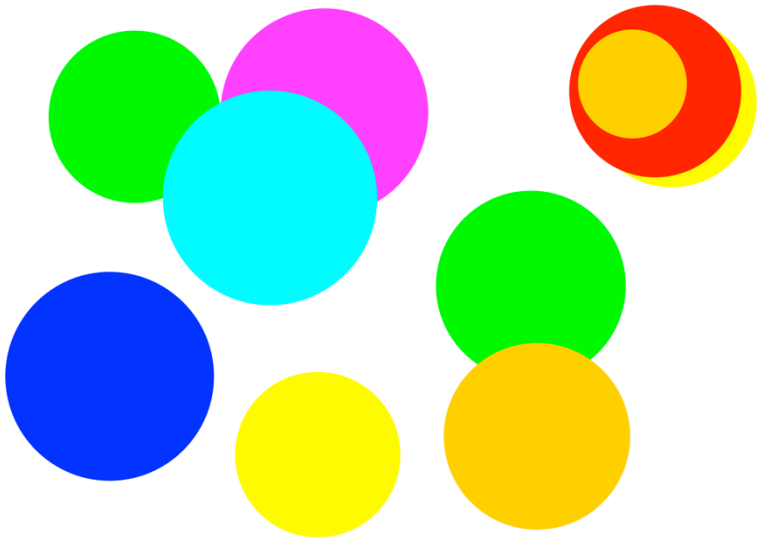
Today's Goal

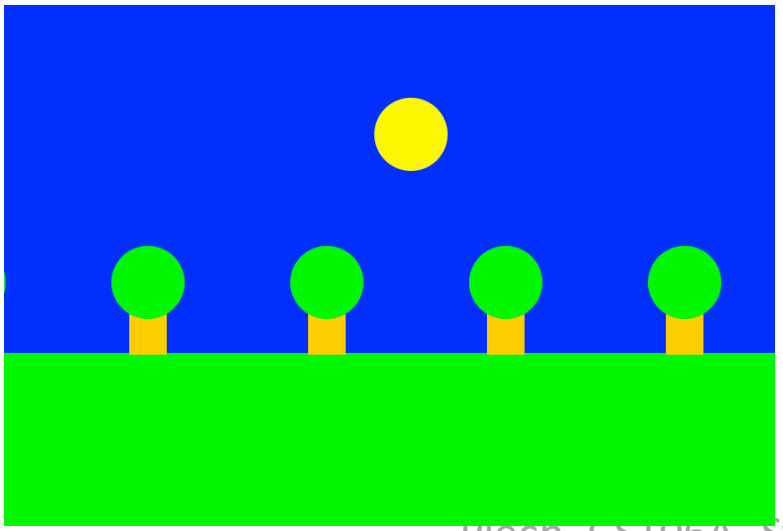
1. Know how to use graphics variables
2. Understand For / While / If in Java

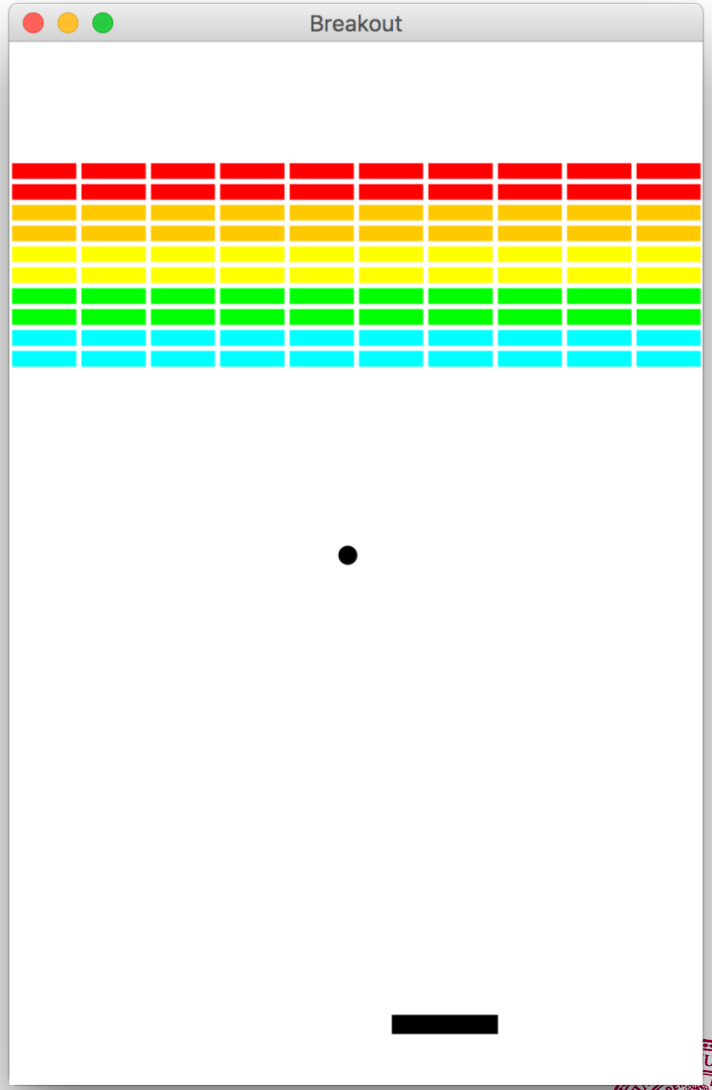
Today's Route



Graphics Programs





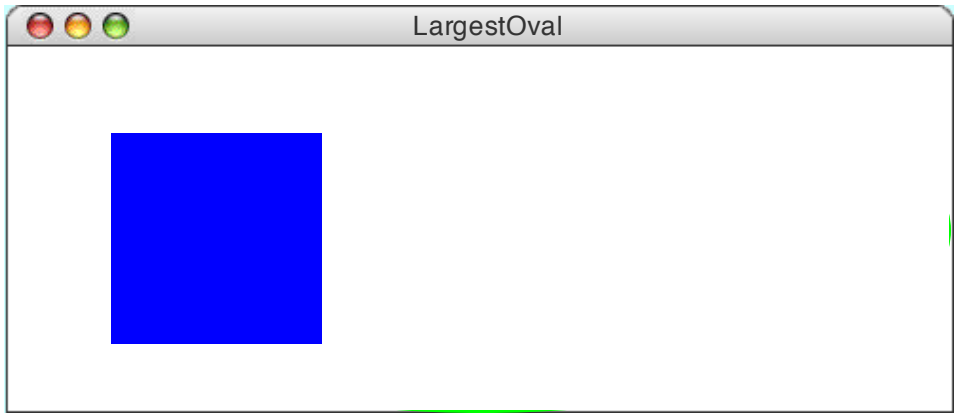


GRect

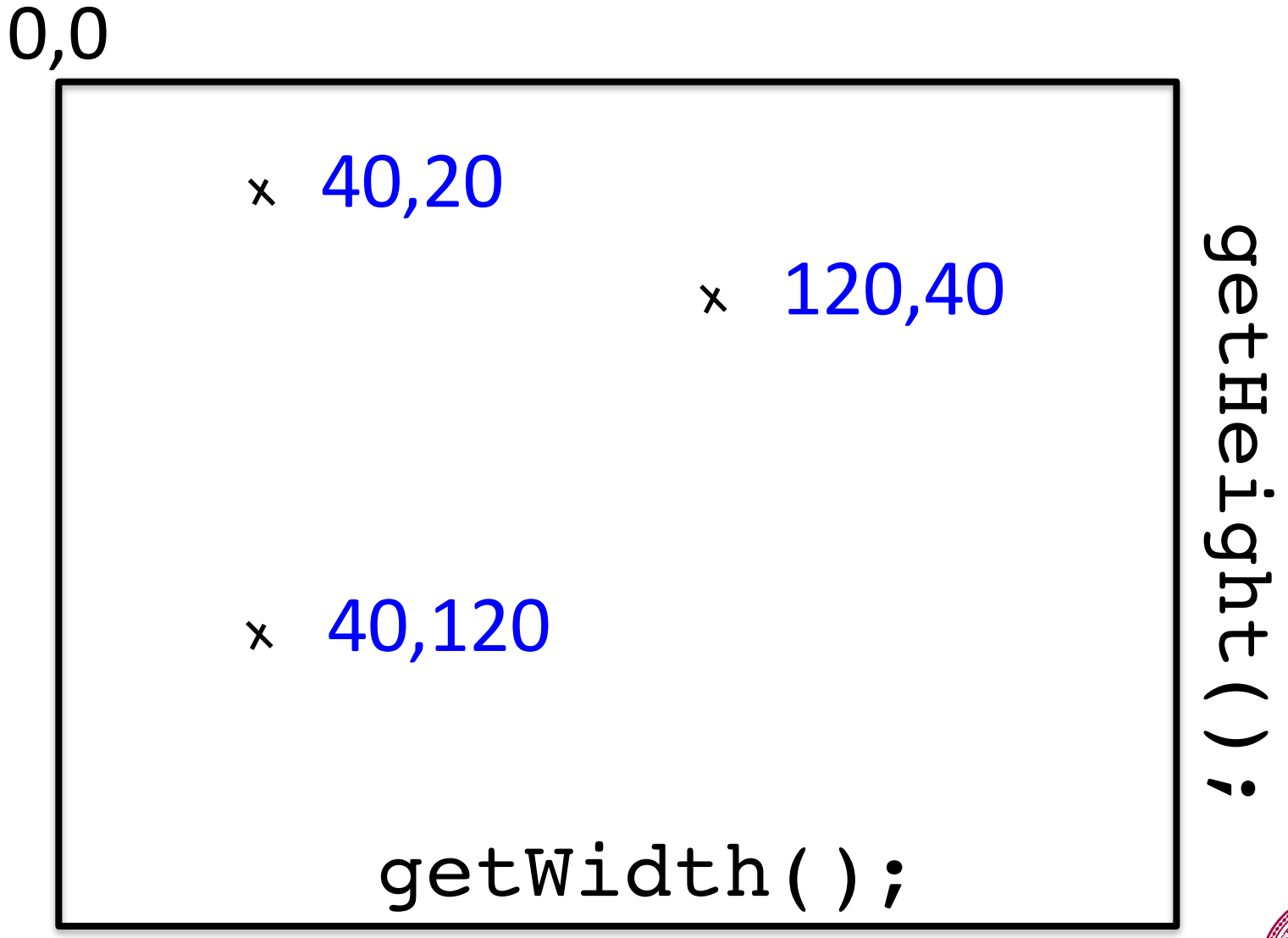
`GRect` is a variable type that stores a rectangle.

As an example, the following `run` method displays a blue square

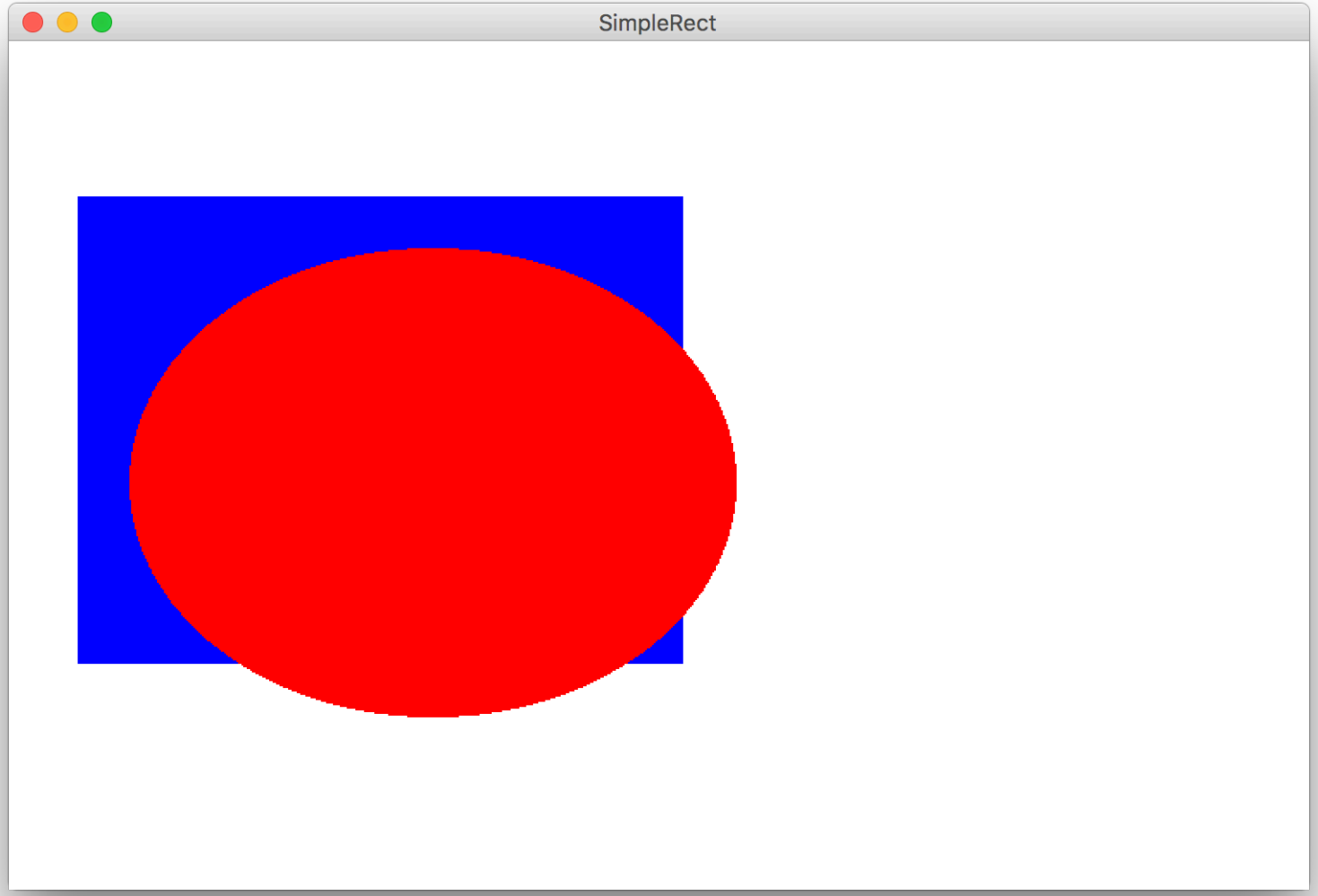
```
public void run() {
   Grect rect = new GRect(200, 200);
   rect.setFilled(true);
   rect.setColor(Color.BLUE);
   add(rect, 50, 50);
}
```



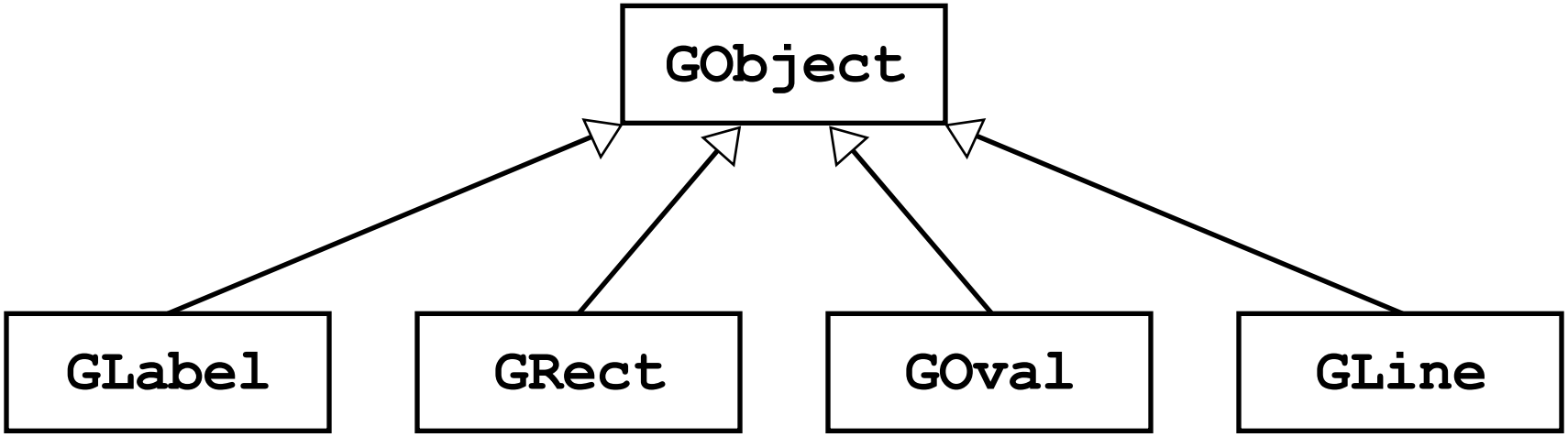
Graphics Coordinates



Learn By Example



Graphics Variable Types




```
GRect myRect = new GRect(350, 270);
```

Primitives vs Classes

Primitive Variable Types

```
int
double
char
boolean
```

Class Variable Types

```
GRect
GOval
GLine
…
```

Class variables:

1. Have upper camel case types
2. You can call methods on them
3. Are constructed using `new`
4. Are stored in a special way

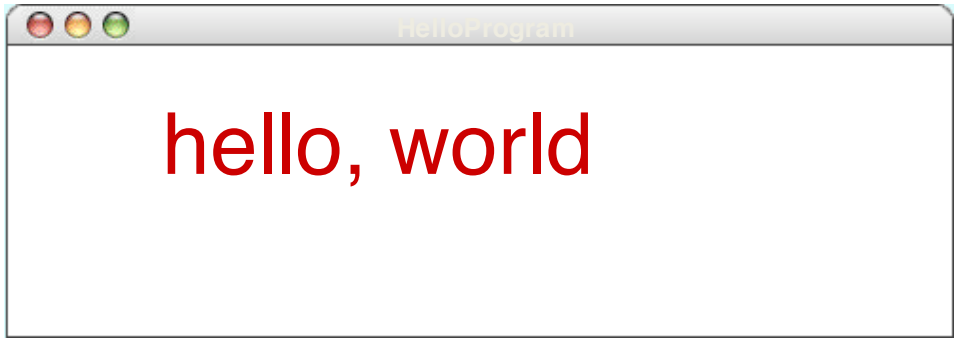
Graphics Trace

The following program illustrates sending a message to an object. Note that the label doesn't appear until it is added to the canvas.

```
public class HelloProgram extends GraphicsProgram {
    public void run() {
        GLabel label = new GLabel("hello, world");
        label.setFont("SansSerif-36");
        label.setColor(Color.RED);
        add(label, 100, 75);
    }
}
```

label

hello, world



Label Location

- **`GLabel`** coordinates are baseline of first character

Operations on GRect

object`.setColor(`*color*`)`
Sets the color of the object to the specified color constant.

The standard color names are defined in the **`java.awt`** package:

`Color.BLACK`	`Color.RED`	`Color.BLUE`
`Color.DARK_GRAY`	`Color.YELLOW`	`Color.MAGENTA`
`Color.GRAY`	`Color.GREEN`	`Color.ORANGE`
`Color.LIGHT_GRAY`	`Color.CYAN`	`Color.PINK`
`Color.WHITE`		

Operations on GRect

object`.setColor(`*color*`)`
Sets the color of the object to the specified color constant.

object`.setLocation(`*x*`, `*y*`)`
Changes the location of the object to the point (*x*, *y*).

object`.move(`*dx*`, `*dy*`)`
Moves the object on the screen by adding *dx* and *dy* to its current coordinates.

object`.setFilled(`*fill*`)`
If *fill* is `true`, fills in the interior of the object; if `false`, shows only the outline.

object`.setFillColor(`*color*`)`
Sets the color used to fill the interior, which can be different from the border.

object`.getWidth()`
Returns the width of the rectangle.

object`.getHeight()`
Returns the height of the rectangle.

Operations on GOval

object`.setColor(`*color*`)`
Sets the color of the object to the specified color constant.

object`.setLocation(`*x*`, `*y*`)`
Changes the location of the object to the point (*x*, *y*).

object`.move(`*dx*`, `*dy*`)`
Moves the object on the screen by adding *dx* and *dy* to its current coordinates.

object`.setFilled(`*fill*`)`
If *fill* is `true`, fills in the interior of the object; if `false`, shows only the outline.

object`.setFillColor(`*color*`)`
Sets the color used to fill the interior, which can be different from the border.

object`.getWidth()`
Returns the width of the rectangle.

object`.getHeight()`
Returns the height of the rectangle.

Operations on GLabel

Methods specific to the **`GLabel`** class

label`.`**`setFont(`***font***`)`** Sets the font used to display the label as specified by the font string.
label`.`**`getAscent()`** Returns the height of the label above its baseline.
label`.`**`getWidth()`** Returns the width of the label.

The font is typically specified as a string in the form

"*family*`-`*style*`-`*size*"

family is the name of a font family
style is either **`PLAIN`**, **`BOLD`**, **`ITALIC`**, or **`BOLDITALIC`**
size is an integer indicating the point size

Construction

new GRect(*width***,** *height***)**
Creates a rectangle with dimensions width and height.

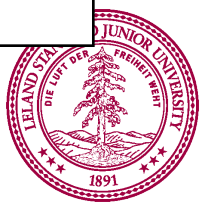
new GOval(*width***,** *height***)**
Creates an oval that fits inside the rectangle with the same dimensions.

new GRect(*x***,** *y***,** *width***,** *height***)**
Creates a rectangle whose upper left corner is at (x, y) of the specified size.

new GOval(*x***,** *y***,** *width***,** *height***)**
Creates an oval that fits inside the rectangle with the same dimensions.

new GLabel(*text***)**
Creates a label with the given text.

new GLine(*x1, y1, x2, y2***)**
Creates a line with the given start and end points.

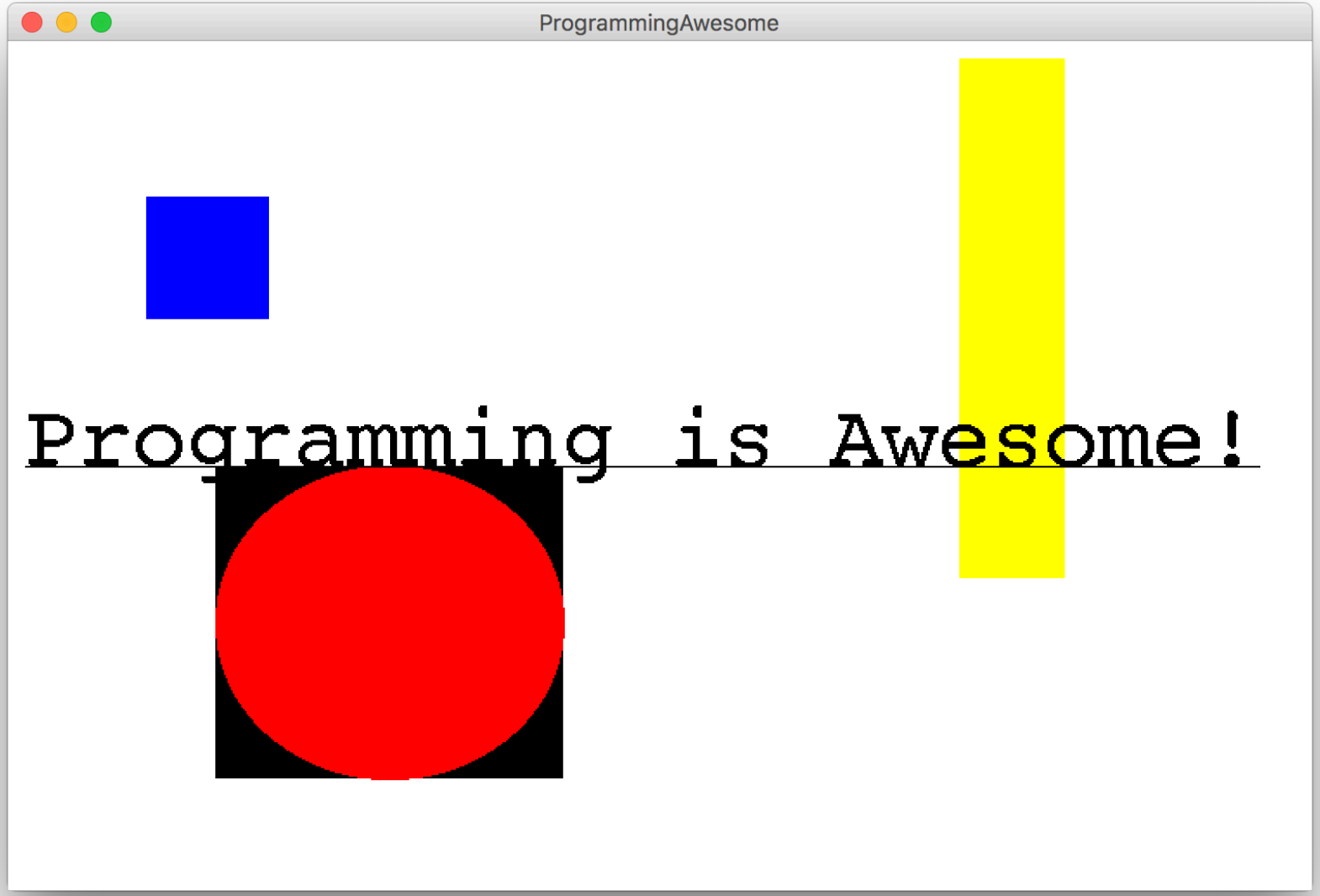


Graphics Methods

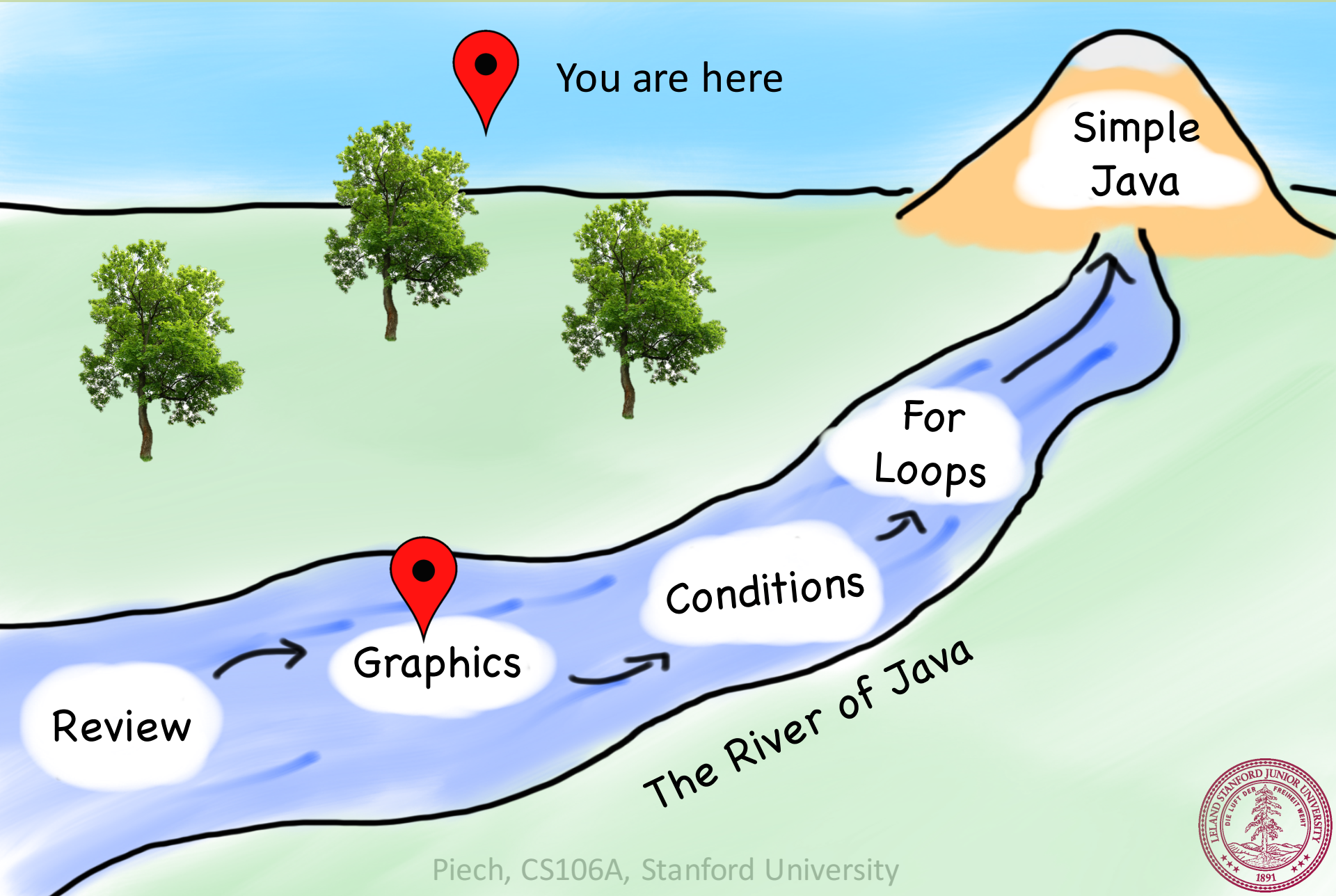
`add(` *object*`)` Adds an object to the canvas. Location is not specified.
`add(` *object, x, y*`)` Adds an object to the canvas at a specific location.

`getWidth()` Returns the width of the screen.
`getHeight(` *object, x, y*`)` Returns the height of the screen.

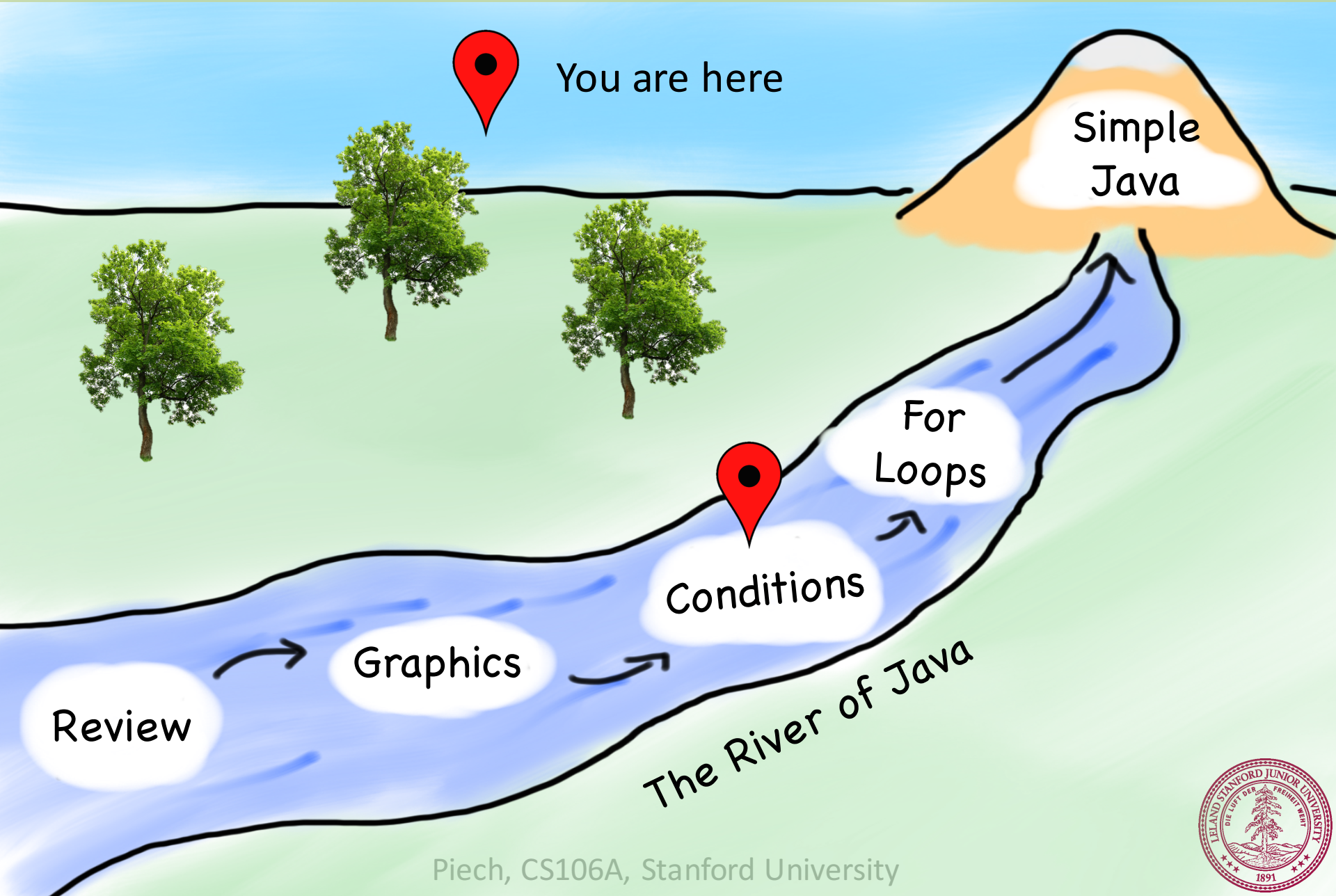
Another Example



Today's Route



Today's Route



While Loop in Karel

```
while(frontIsClear()) {
    body
}
```

```
if(beepersPresent()) {
    body
}
```


While Loop Redux

```
while(condition) {
    body
}
```

```
if(condition) {
    body
}
```

The condition should be a "boolean" which is either **`true`** or **`false`**

What Does This Do?

```
// read the amount of C14 from the user
double amount = readDouble("Amount of C14 in your sample: ");

// use the half life formula to calculate the age
double age = Math.log(amount/LIVING_C14) / Math.log(0.5) *
             HALF_LIFE;
age = Math.round(age);
println("Your sample is " + age + " years old.");
```

* It calculates the age of a C14 sample

What Does This Do?

Before repeating the body, check if this statement evaluates to true

```
while(true) {
    // read the amount of C14 from the user
    double amount = readDouble("Amount of C14 in your sample: ");

    // use the half life formula to calculate the age
    double age = Math.log(amount/LIVING_C14) / Math.log(0.5) *
                 HALF_LIFE;
    age = Math.round(age);
    println("Your sample is " + age + " years old.");

    // add an extra line between queries
    println("");
}
```

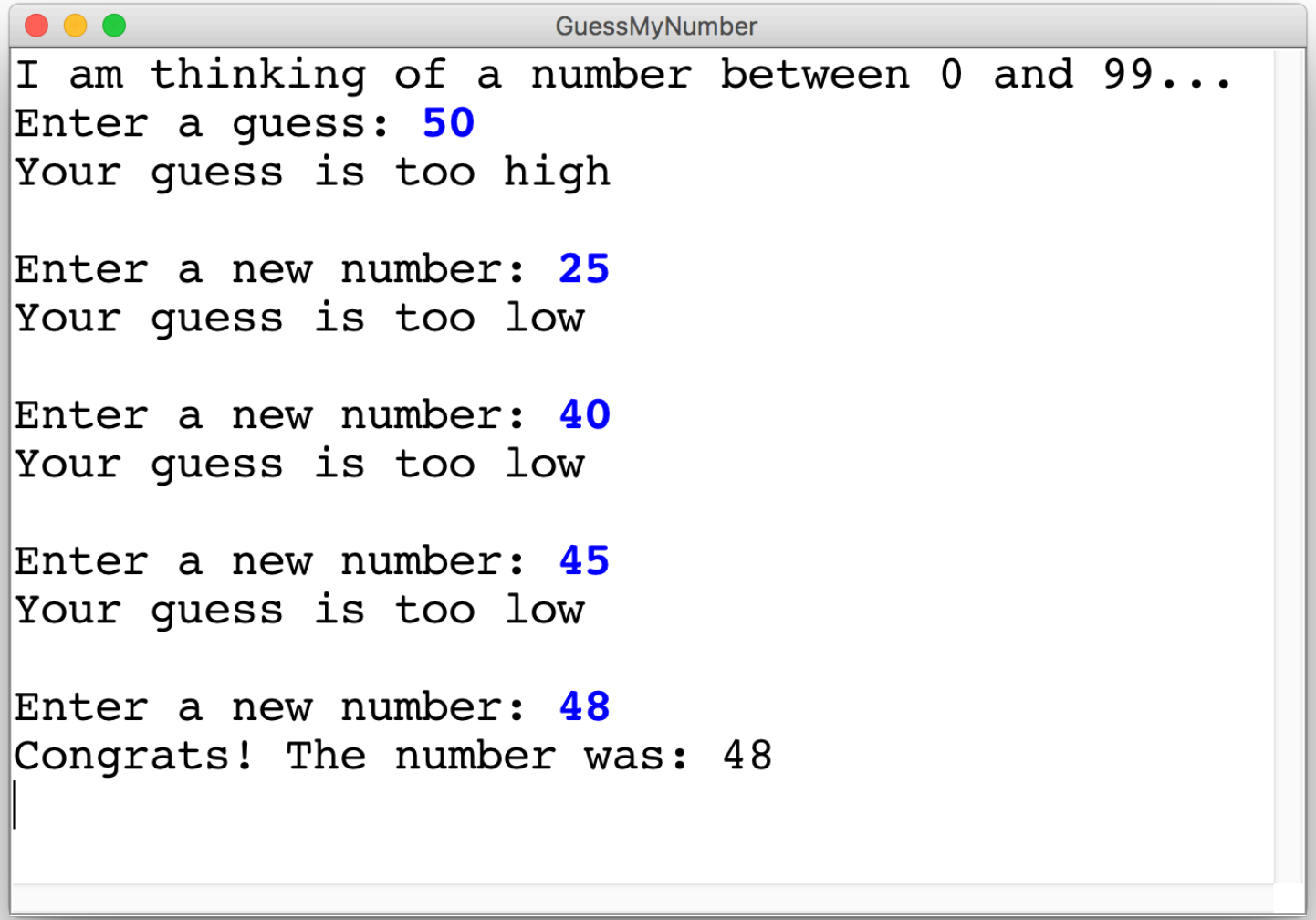
* It repeatedly calculates the age of a C14 sample

Boolean Comparisons

`==`	Equals Test
`!=`	Not equals

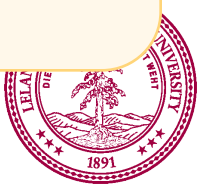
`<`	Less than	`>`	Greater than
`<=`	Less than equals	`>=`	Greater than equals

Guess My Number



Guess My Number

```
int secretNumber = getHeight() % 100;
println("I am thinking of a number between 0 and 99...");
int guess = readInt("Enter a guess: ");
// true if guess is not equal to secret number
while(guess != secretNumber) {
    // true if guess is less than secret number
    if(guess < secretNumber) {
        println("Your guess is too low");
    } else {
        println("Your guess is too high");
    }
    println(""); // an empty line
    guess = readInt("Enter a new number: ");
}
println("Congrats! The number was: " + secretNumber);
```



How would you println “Nick rocks socks” 100 times

For Loop Redux

```
public void run() {
   for(int i = 0; i < 100; i++) {
      println("Nick rocks socks!");
   }
}
```


For Loop Redux

This line is run once, just before the for loop starts

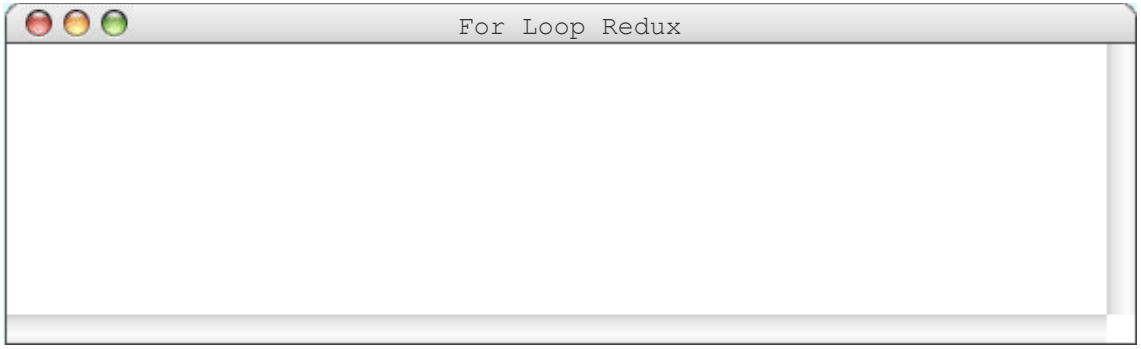
Enters the loop if this condition passes

This line is run each time the code gets to the end of the 'body'

```
for(int i = 0; i < 100; i++) {
   println("Nick rocks socks!");
}
```

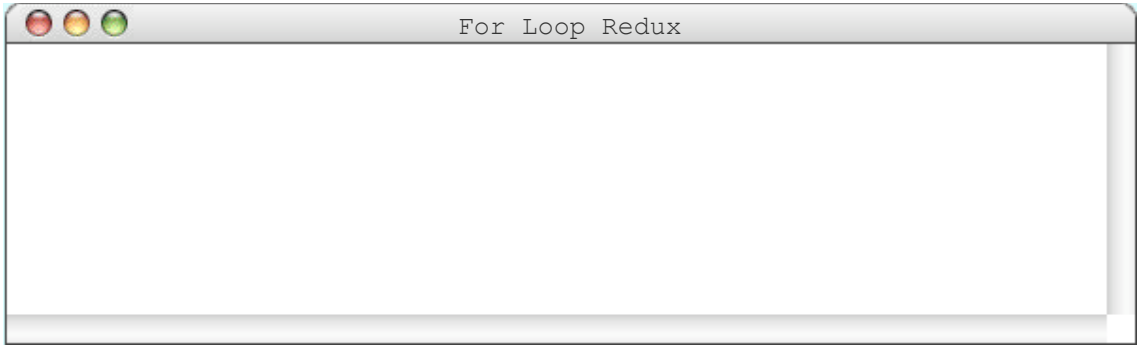

For Loop Redux

```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

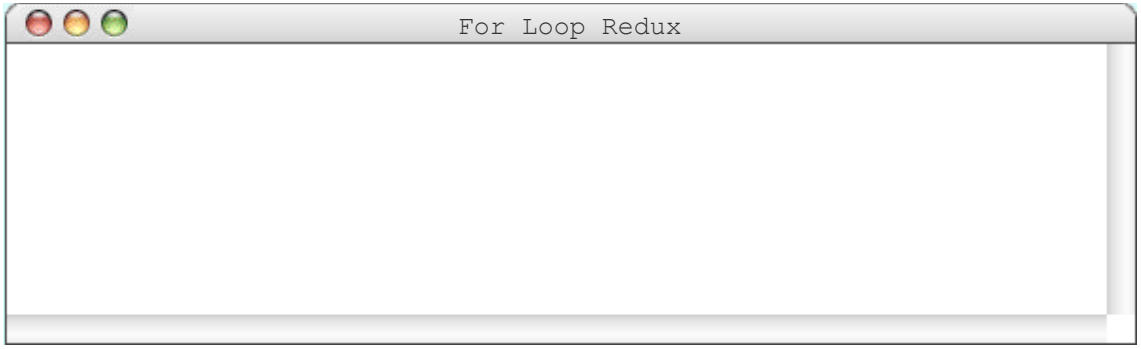
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i	0

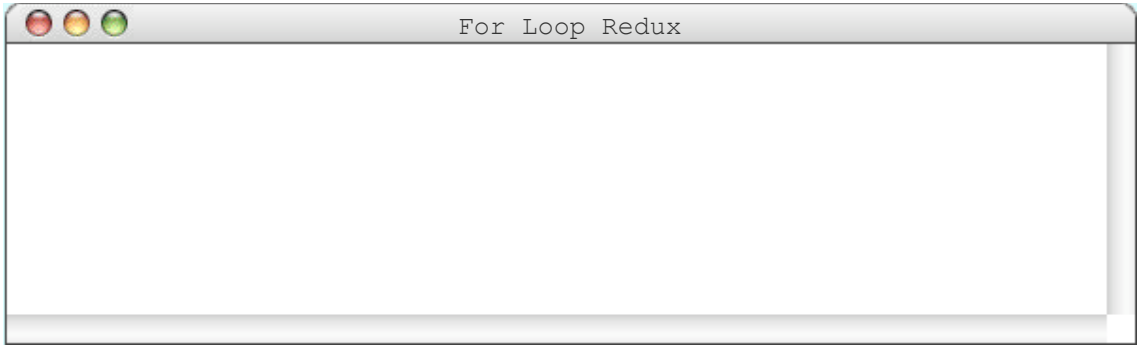
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i	0

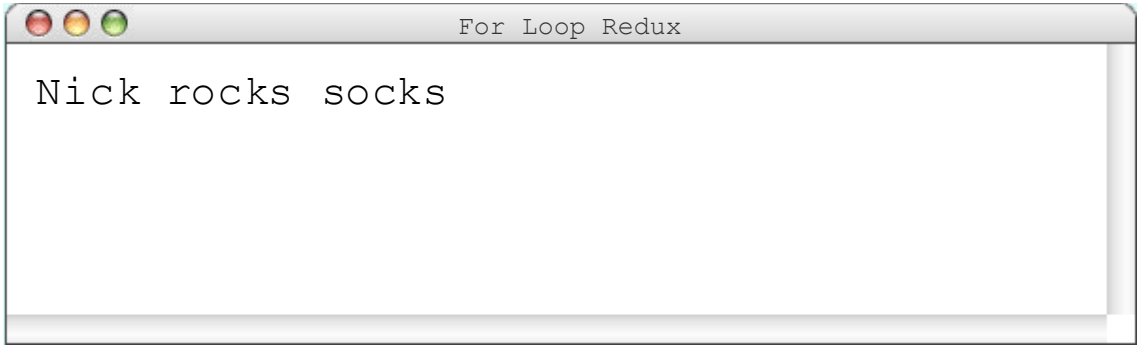
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i	0

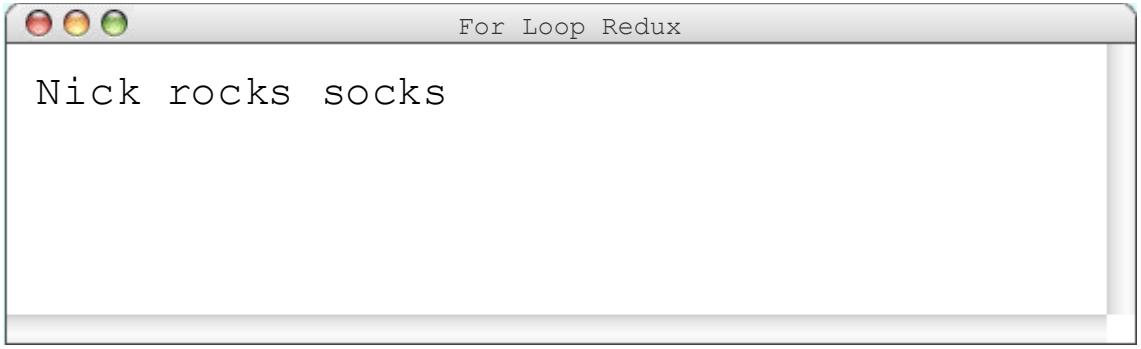
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i 1

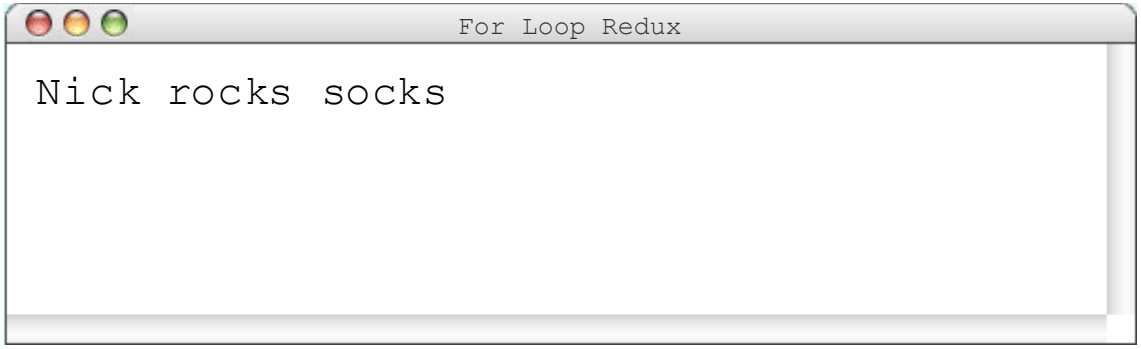
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i | 1

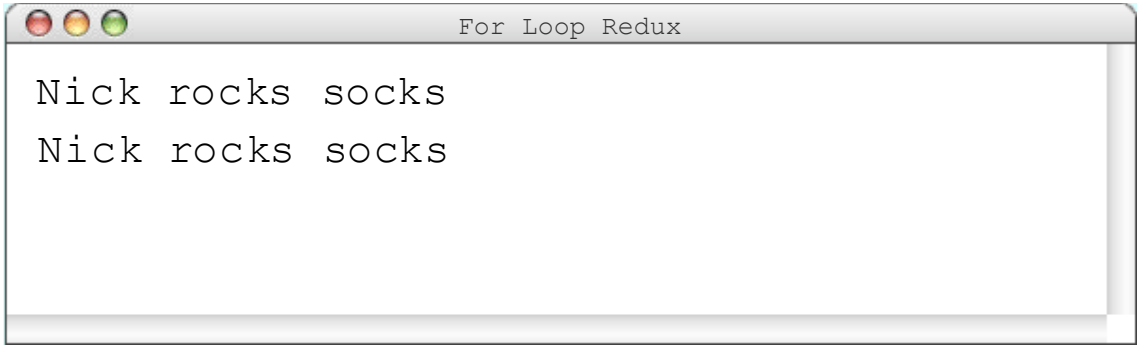
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i | 1

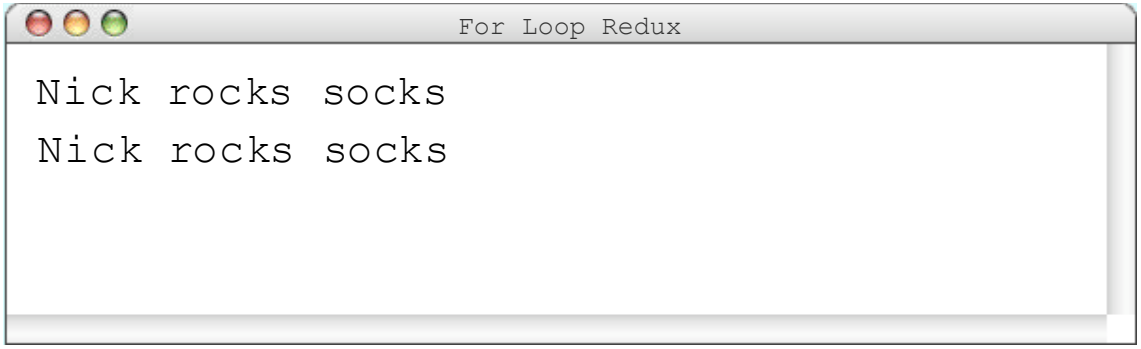
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i 2

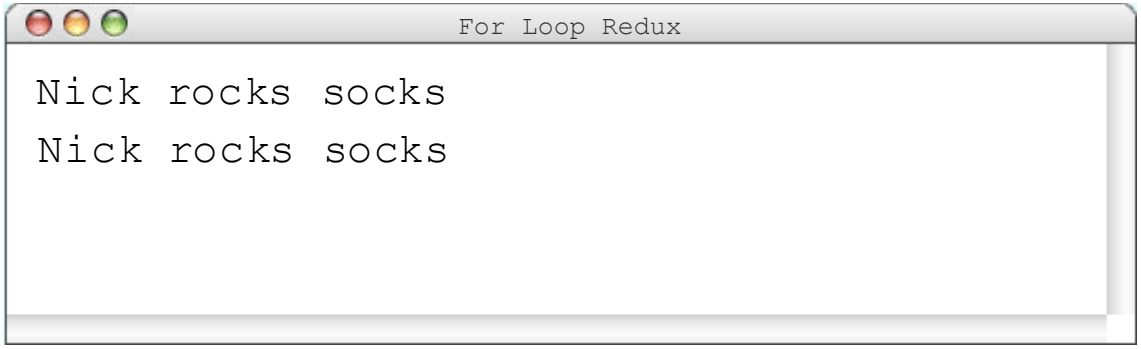
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i	2

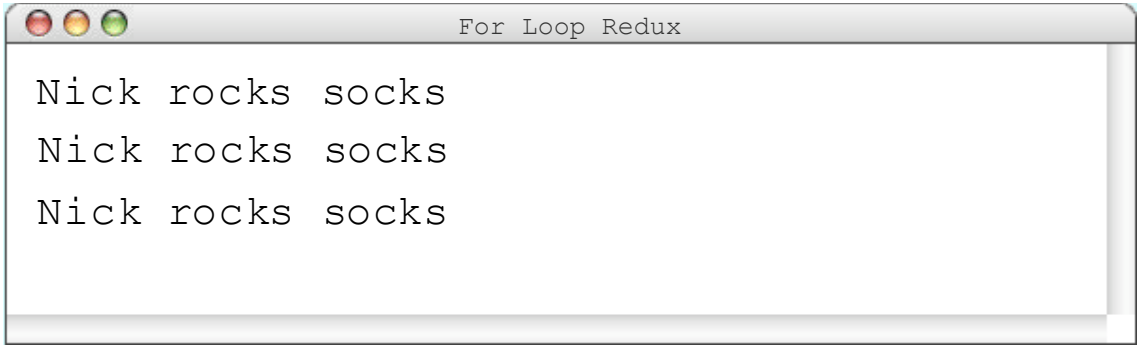
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i	2

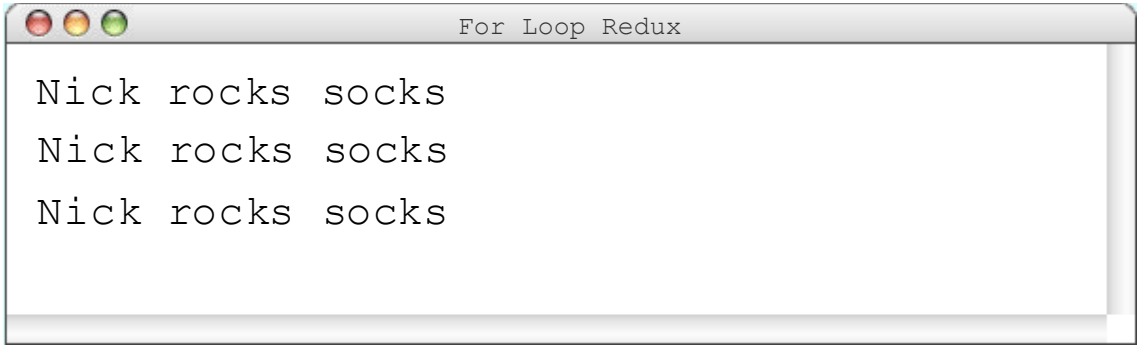
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

i 3

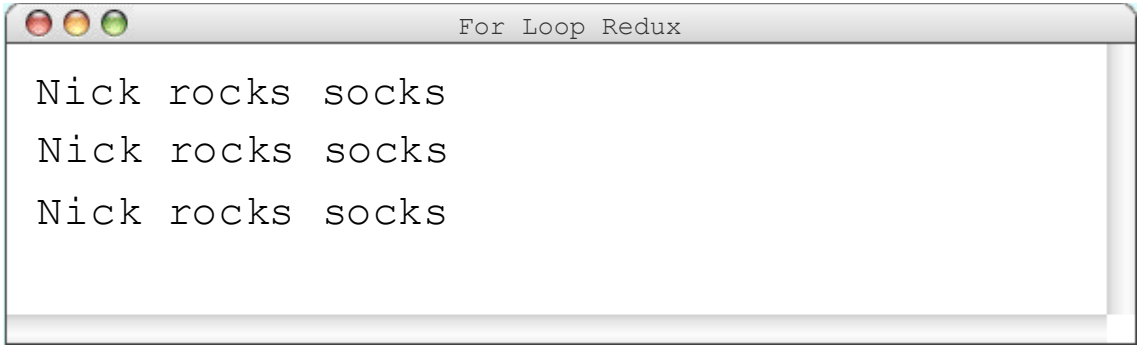
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

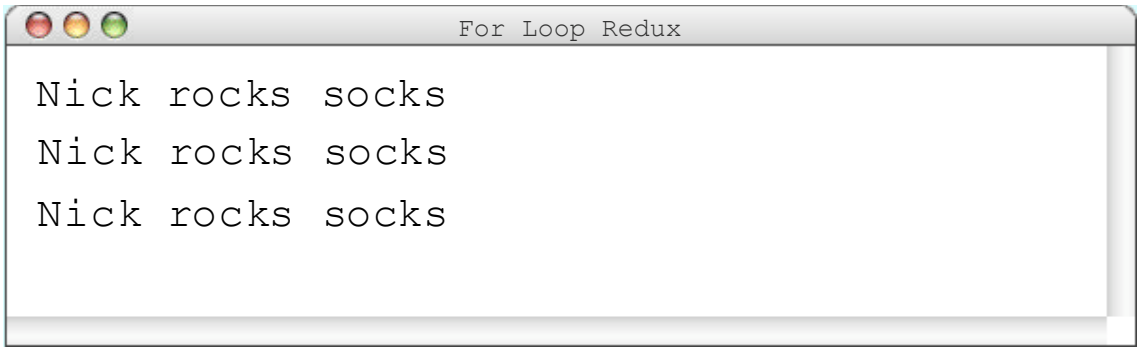
i	3

```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



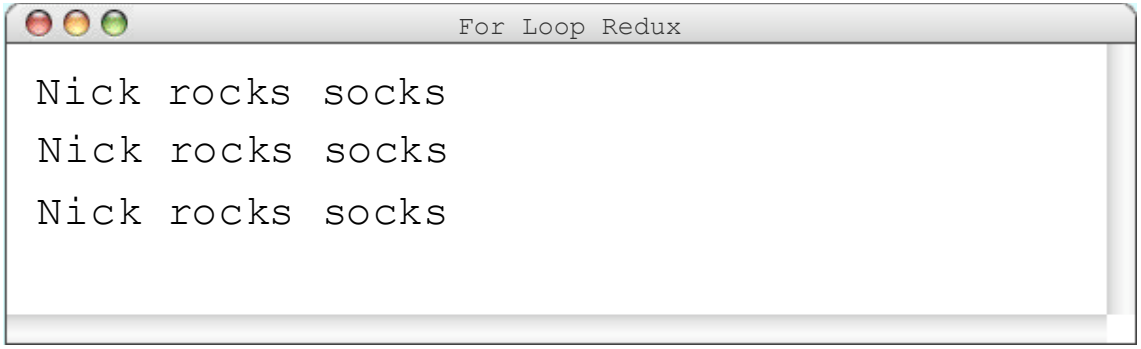
For Loop Redux

```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



For Loop Redux

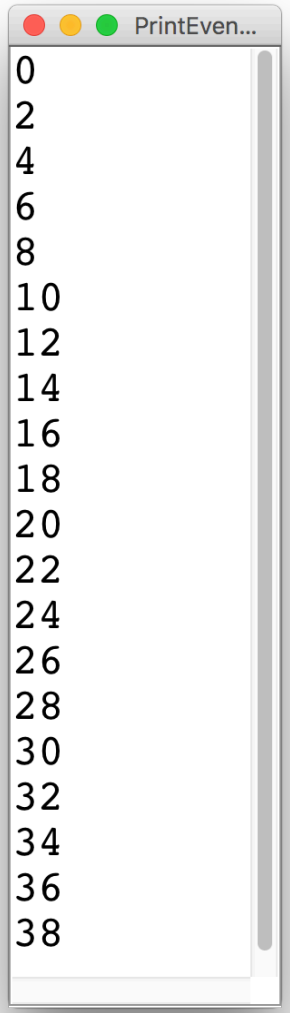
```
for(int i = 0; i < 3; i++) {
   println("Nick rocks socks!");
}
```



You can use the for loop variable

How would you println the first 100 even numbers?

Printing Even Numbers

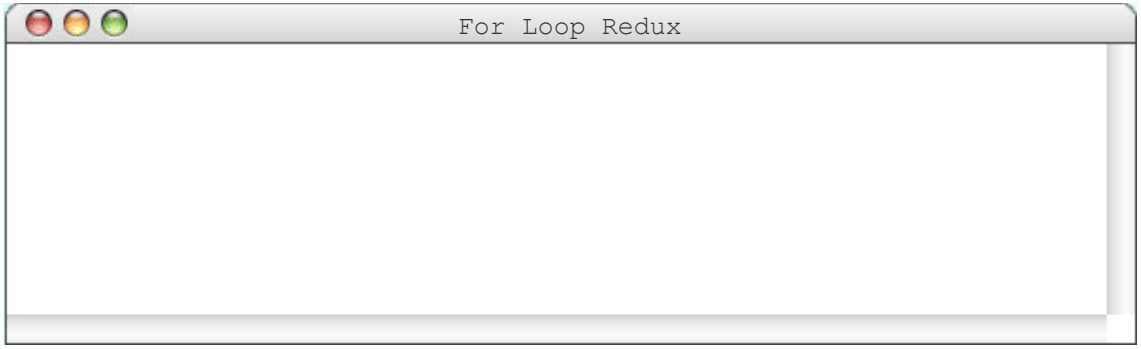


Printing Even Numbers

```
for(int i = 0; i < NUM_NUMS; i++) {
   println(i * 2);
}
```

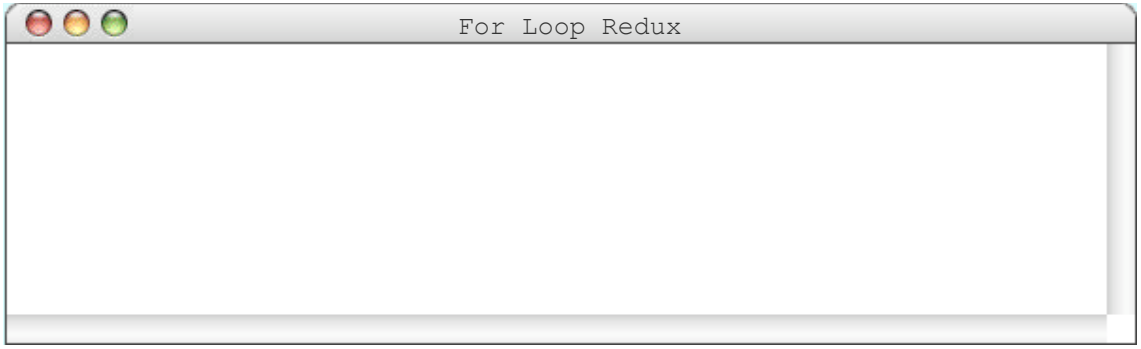

Printing Even Numbers

```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

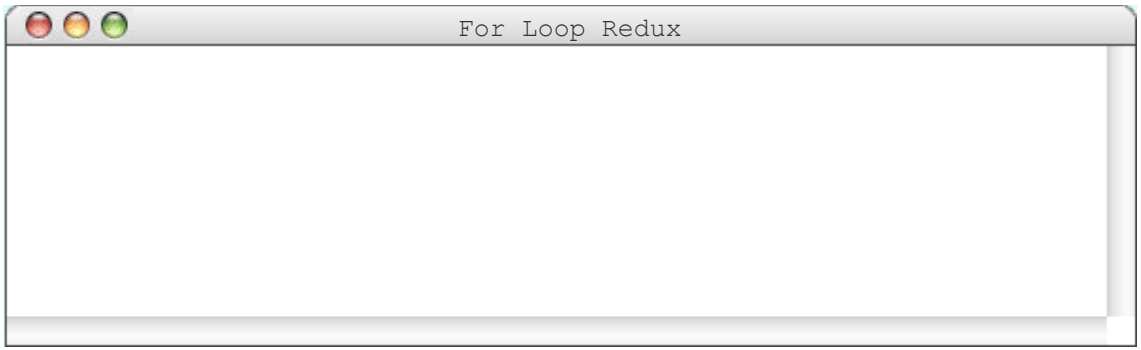
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i	0

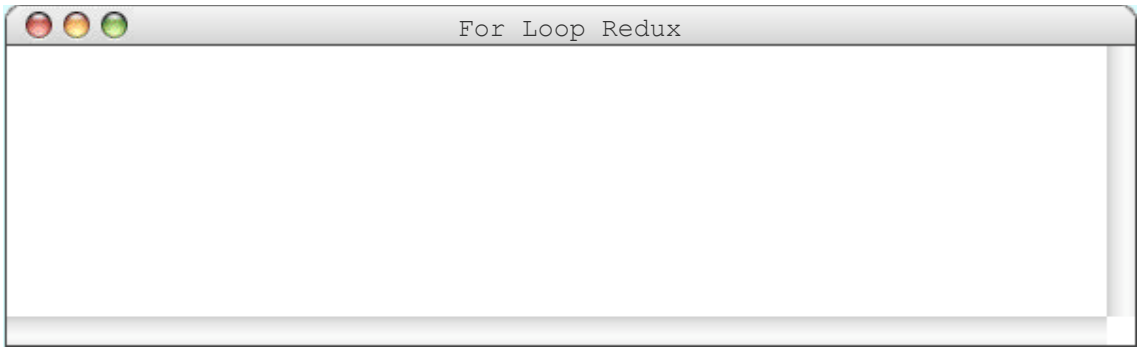
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i	0

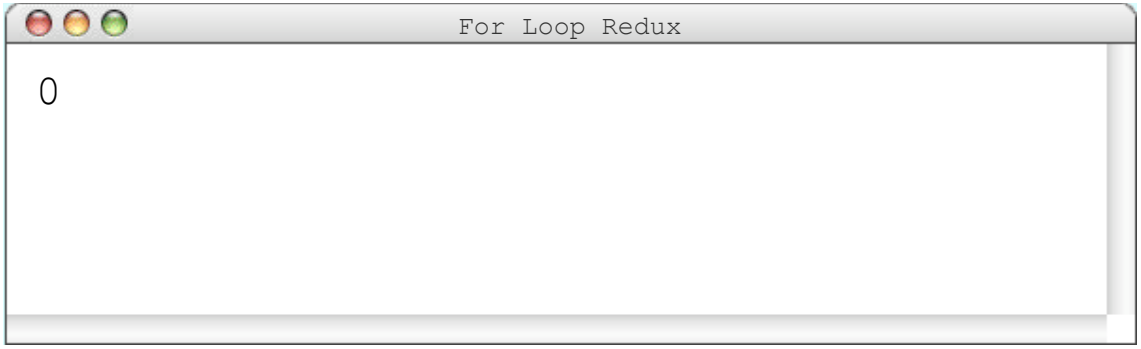
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i	0

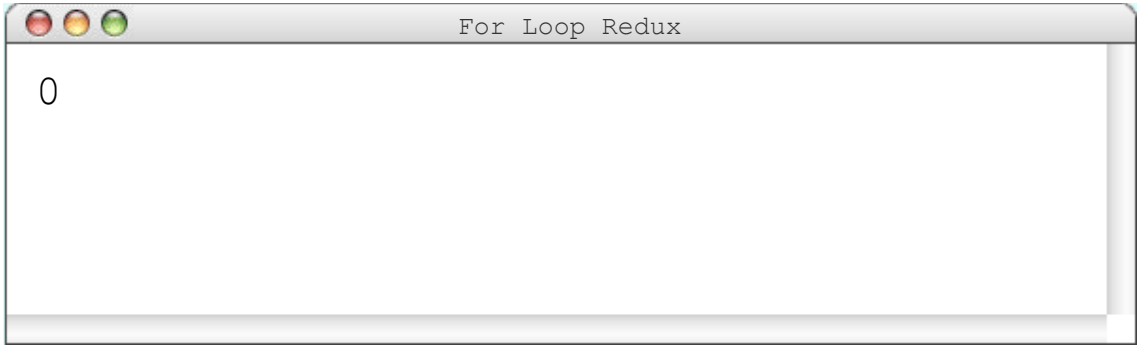
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i 1

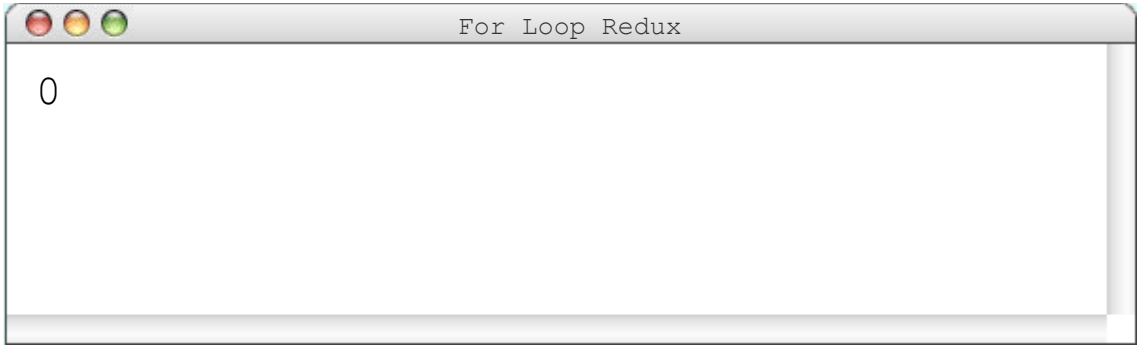
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i 1

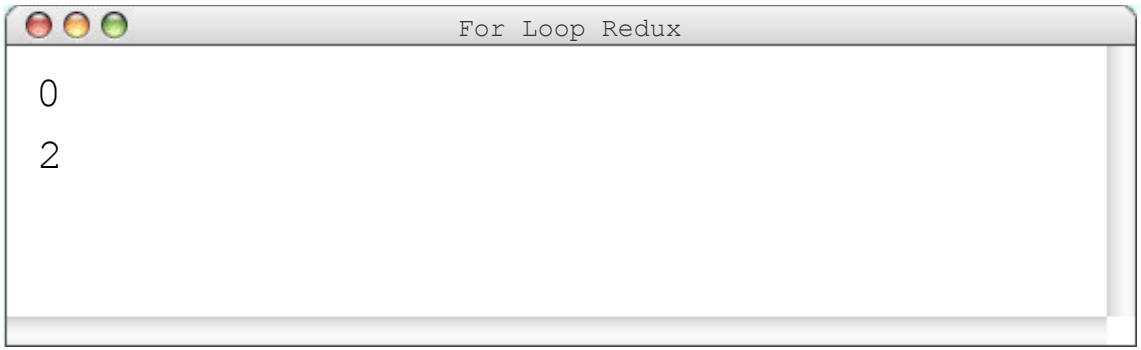
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i | 1

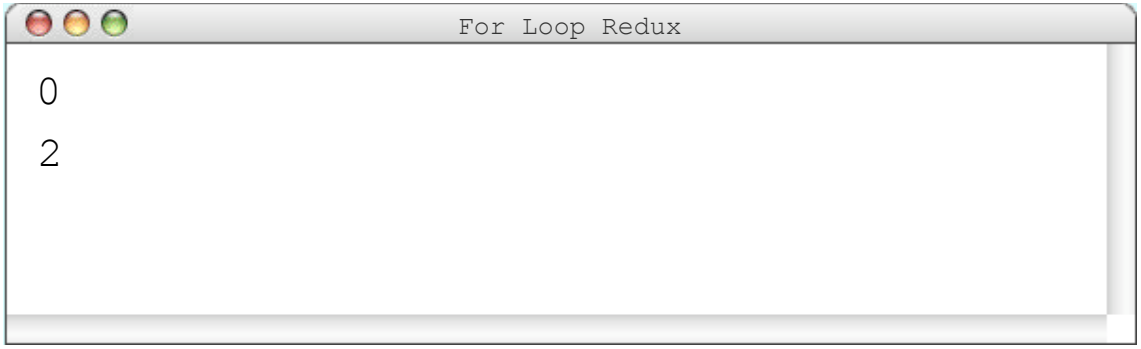
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

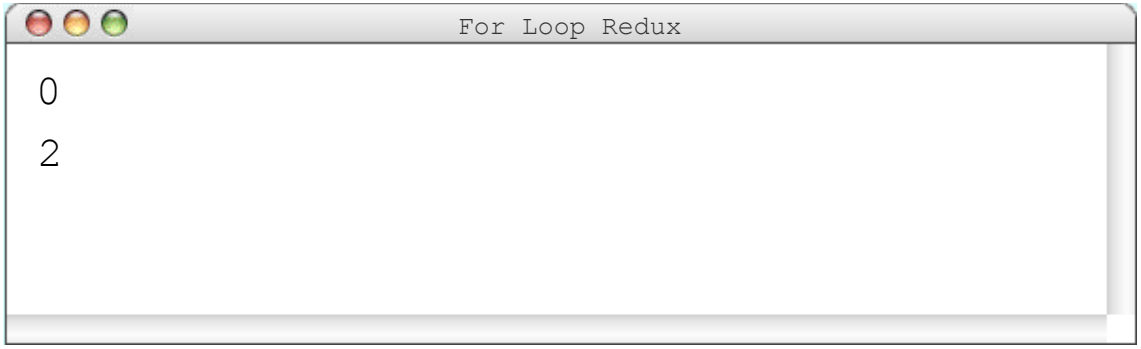
i 2

```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



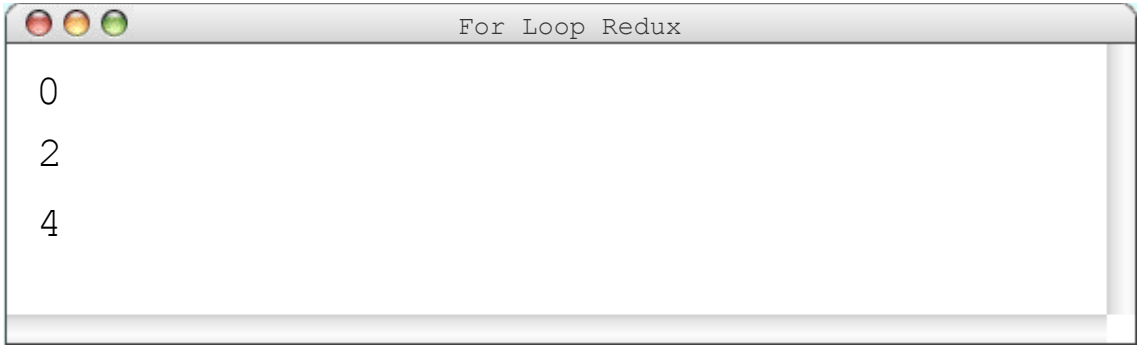
Printing Even Numbers


```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

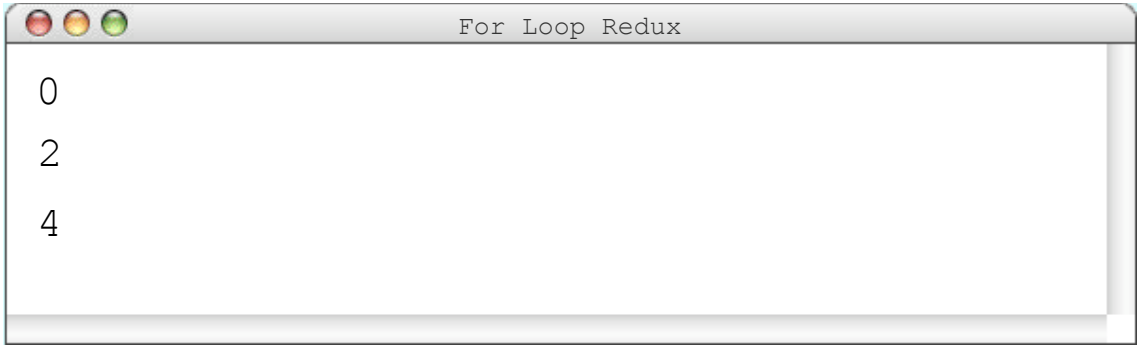

```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i 3

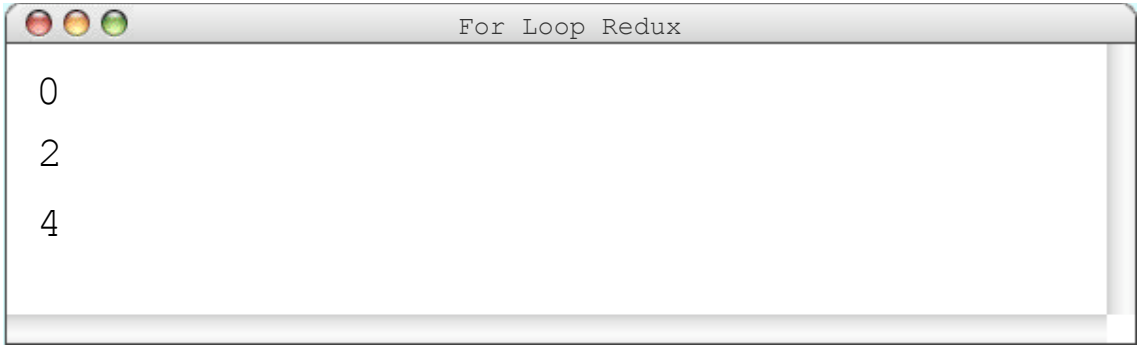
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

i	3

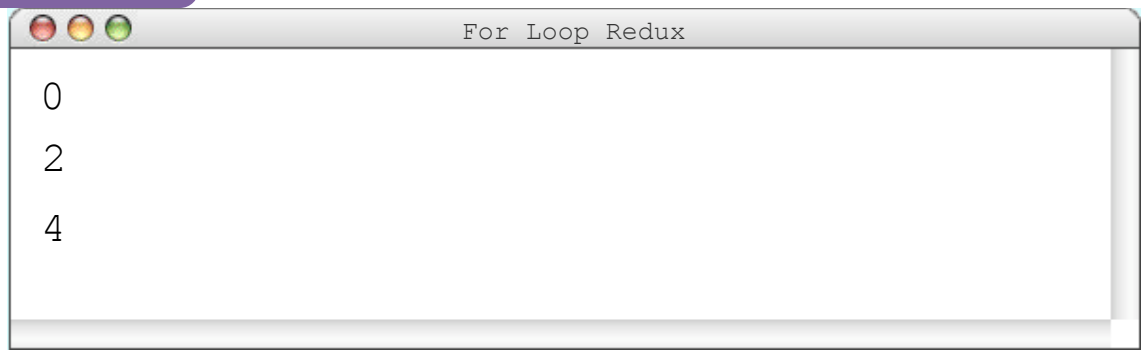
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Printing Even Numbers

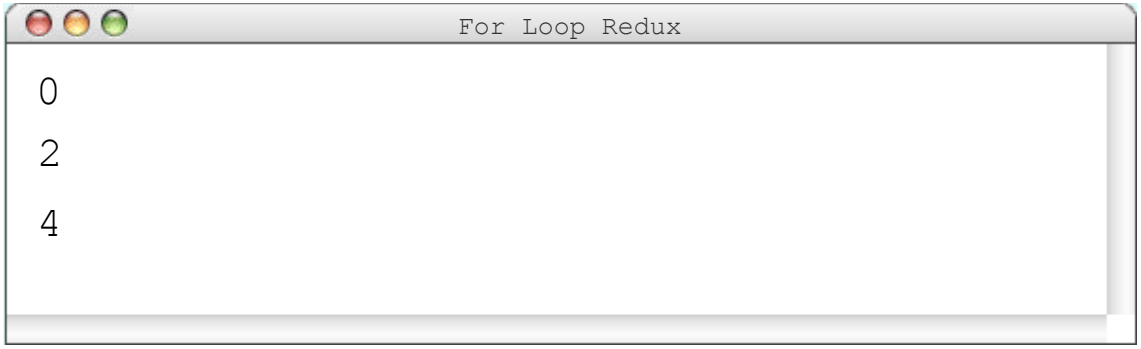
i	3

```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```

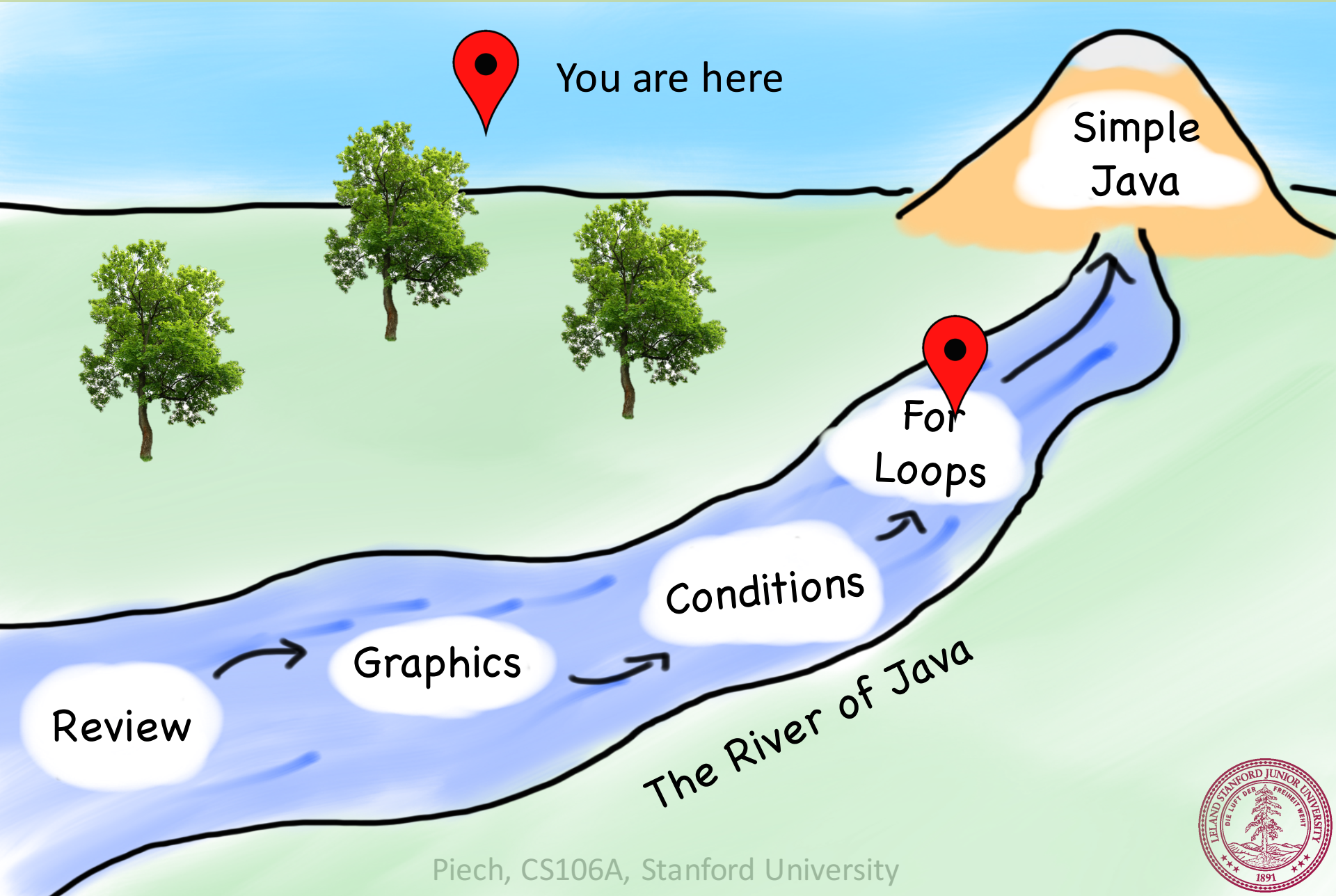


Printing Even Numbers

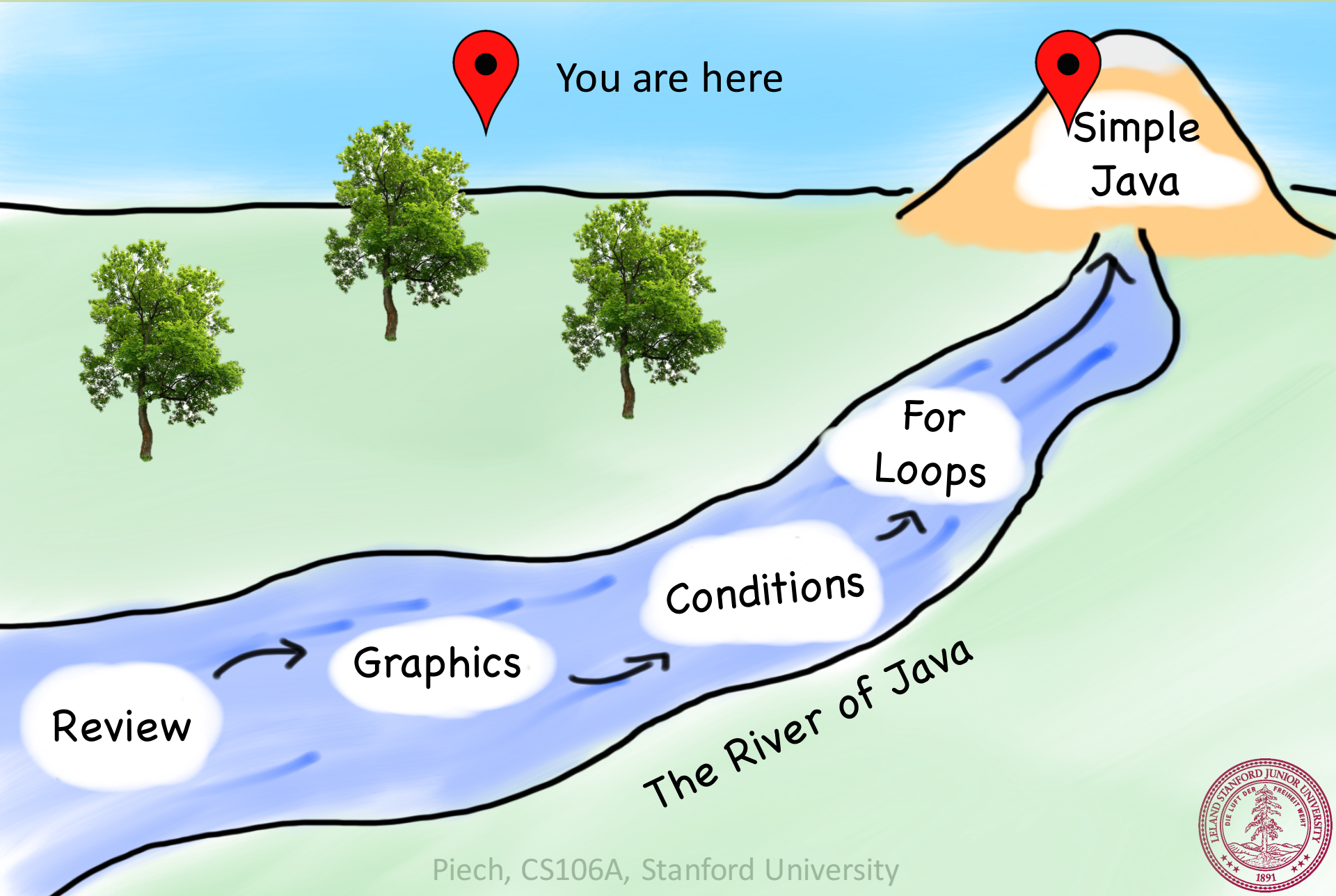
```
for(int i = 0; i < 3; i++) {
   println(i * 2);
}
```



Today's Route



Today's Route



Today's Goal

1. Know how to use graphics variables
2. Understand For / While / If in Java

String Art