



Nested Loops

Chris Piech

CS106A, Stanford University

Boolean Variables

```
// Store expressions that evaluate to true/false
boolean x = 1 < 2;      // true
boolean y = 5.0 == 4.0;  // false
```



Boolean Variables

```
// Store expressions that evaluate to true/false
boolean x = 1 < 2;      // true
boolean y = 5.0 == 4.0;  // false

// Directly set to true/false
boolean isFamilyVisiting = true;
boolean isRaining = false;
```



Boolean Variables

```
// Store expressions that evaluate to true/false
boolean x = 1 < 2;      // true
boolean y = 5.0 == 4.0;  // false

// Directly set to true/false
boolean isFamilyVisiting = true;
boolean isRaining = false;

// Ask the user a true/false (yes/no) question
boolean playAgain = readBoolean("Play again?", "y", "n");
if (playAgain) {
    ...
}
```



Please ...

**NO FOOD OR
DRINKS**

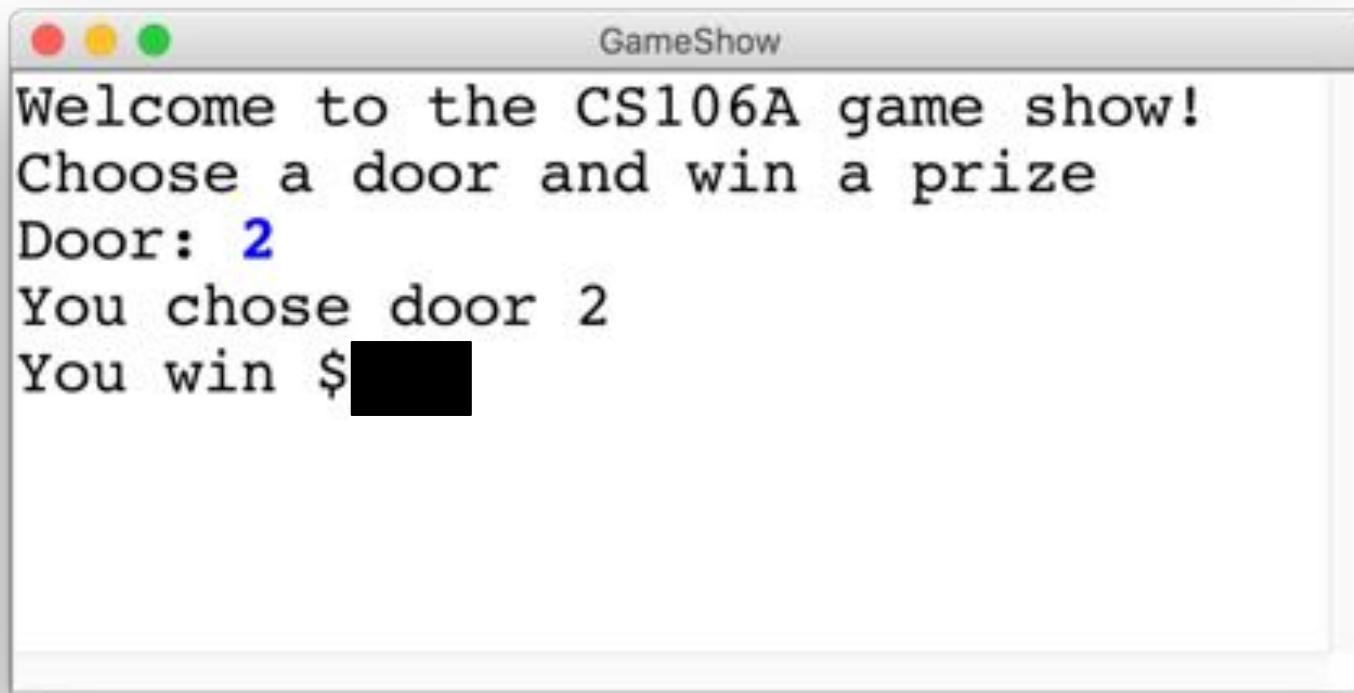
FreeSignsPrinter.com

*know your logical precedence

Piech, CS106A, Stanford University



Game Show



Choose a Door

```
int door = readInt("Door: ");
// while the input is invalid
while(door < 1 || door > 3) {
    // tell the user the input was invalid
    println("Invalid door!");
    // ask for a new input
    door = readInt("Door: ");
}
```

|| or
&& and



The Door Logic

```
int prize = 4;  
if(door == 1) {  
    prize = 2 + 9 / 10 * 100;  
} else if(door == 2) {  
    boolean locked = prize % 2 != 0;  
    if(!locked) {  
        prize += 6;  
    }  
} else if(door == 3) {  
    prize++;  
}
```



The Door Logic

```
int prize = 4;  
if(door == 1) {  
    prize = 2 + 9 / 10 * 100;  
} else if(door == 2) {  
    boolean locked = prize % 2 != 0;  
    if(!locked) {  
        prize += 6;  
    }  
} else if(door == 3) {  
    prize++;  
}
```



The Door Logic

```
int prize = 4;  
if(door == 1) {  
    prize = 2 + 9 / 10 * 100;  
} else if(door == 2) {  
    boolean locked = prize % 2 != 0;  
    if(!locked) {  
        prize += 6;  
    }  
} else if(door == 3) {  
    prize++;  
}
```



The Door Logic

```
int prize = 4;  
if(door == 1) {  
    prize = 2 + 9 / 10 * 100;  
} else if(door == 2) {  
    boolean locked = prize % 2 != 0;  
    if(!locked) {  
        prize += 6;  
    }  
} else if(door == 3) {  
    prize++;  
}  
}
```



The Door Logic

```
int prize = 4;  
if(door == 1) {  
    prize = 2 + 9 / 10 * 100;  
} else if(door == 2) {  
    boolean locked = prize % 2 != 0;  
    if(!locked) {  
        prize += 6;  
    }  
} else if(door == 3) {  
    prize++;  
}
```



How would you `println` “Stanford rocks socks”
100 times

For Loop Redux

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



For Loop Redux

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

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

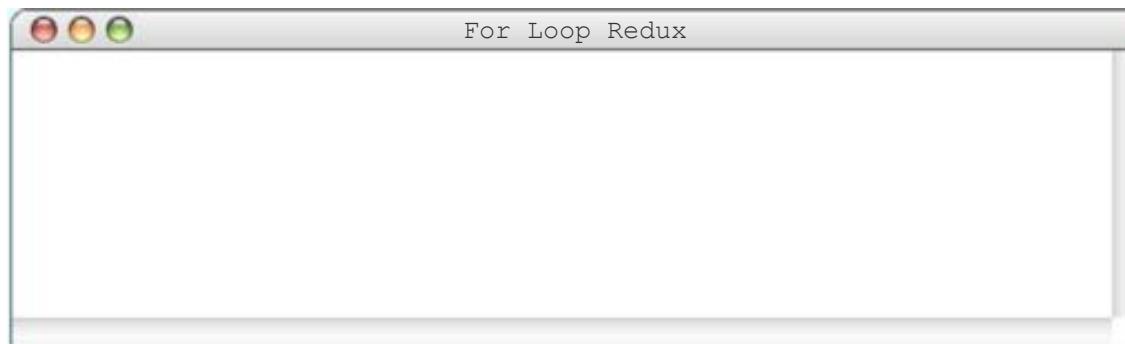
Enters the loop if this condition passes

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



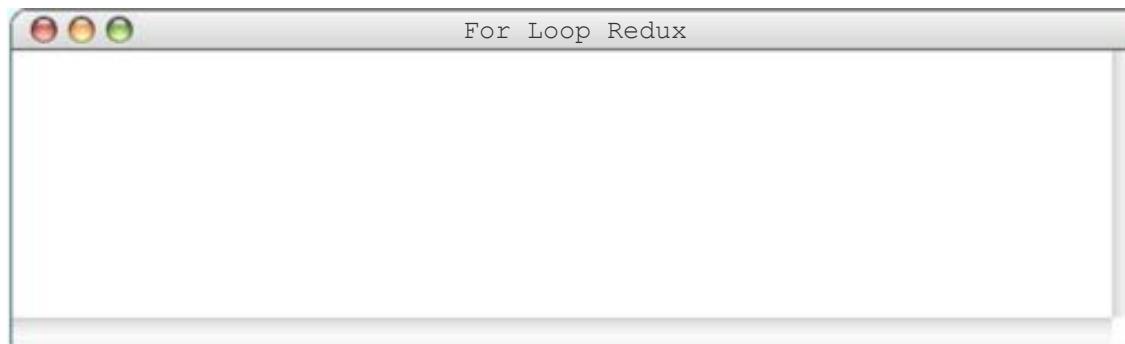
For Loop Redux

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



For Loop Redux

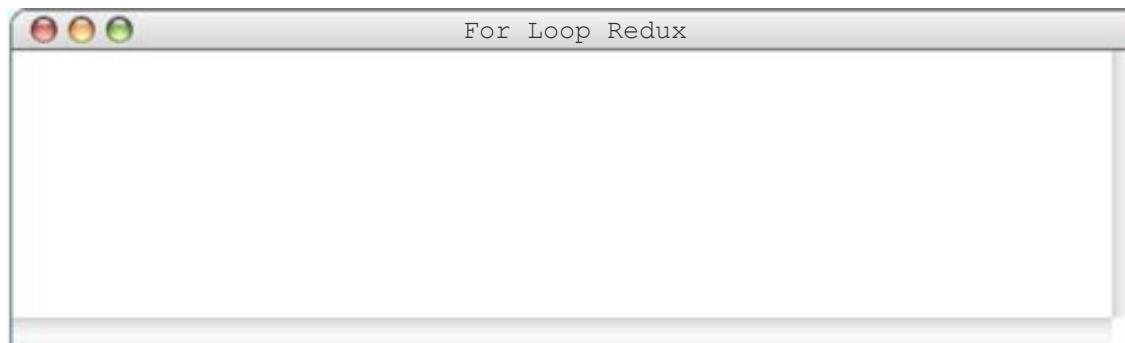
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 0

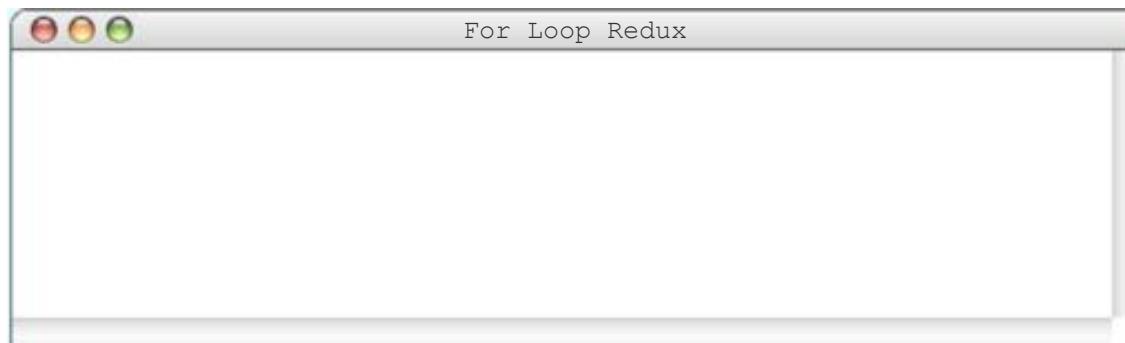
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 0

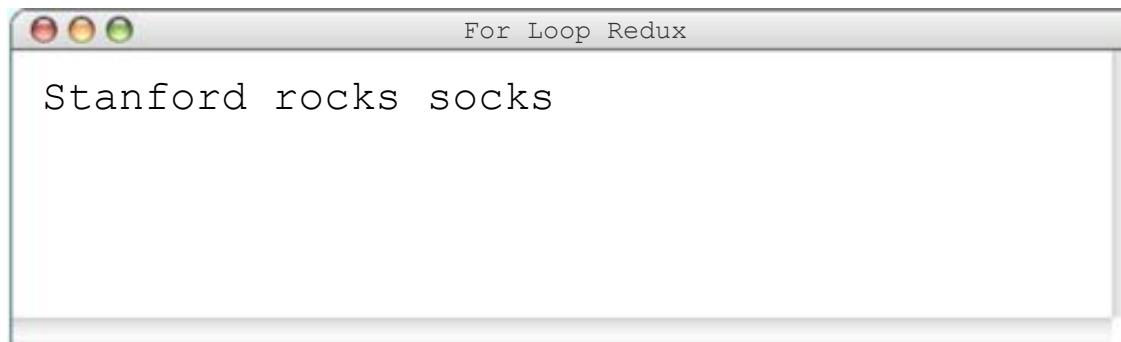
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 0

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



For Loop Redux

i 1

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



For Loop Redux

i 1

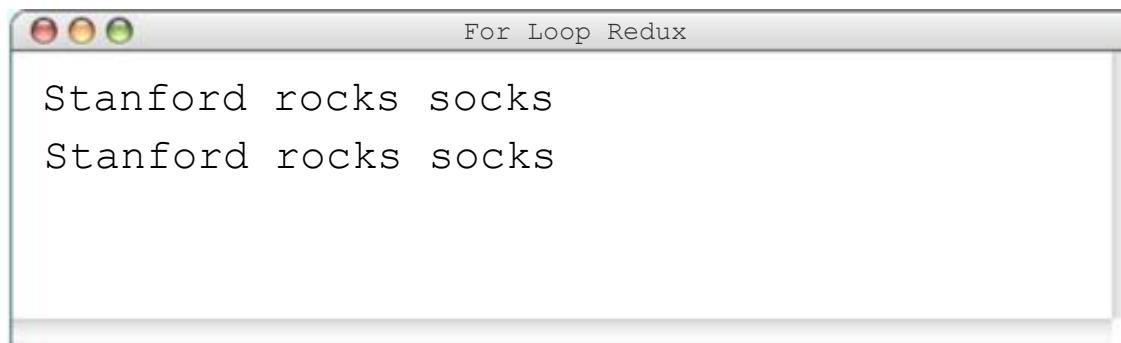
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 1

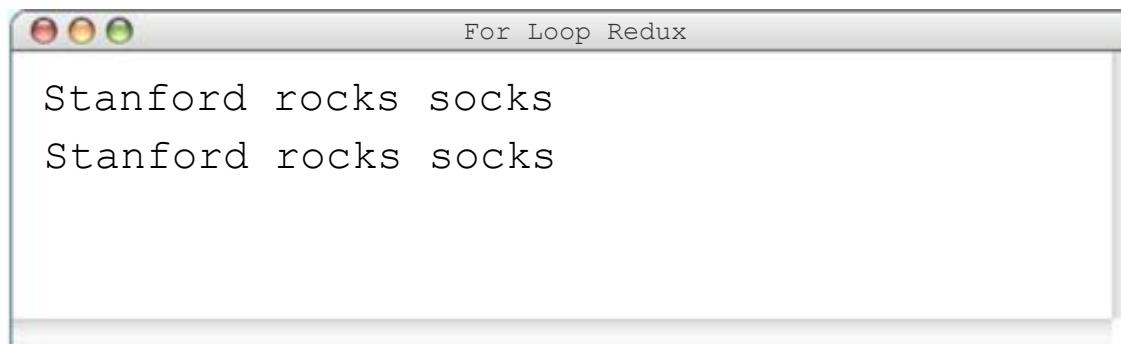
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 2

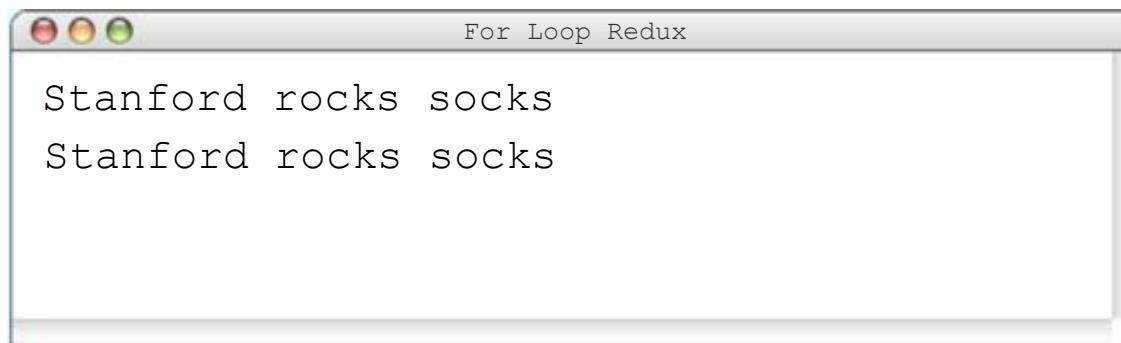
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 2

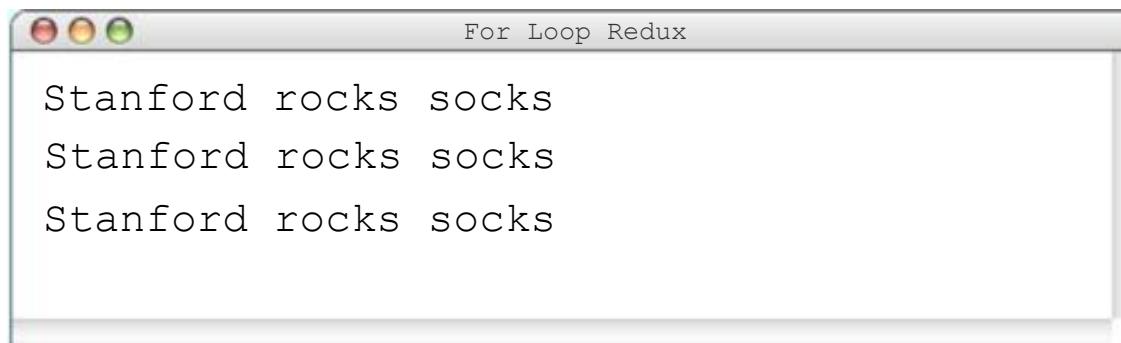
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 2

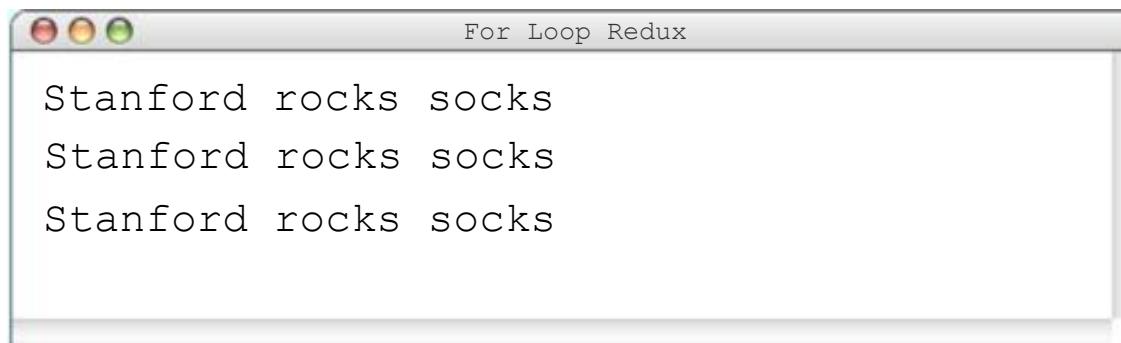
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

i 3

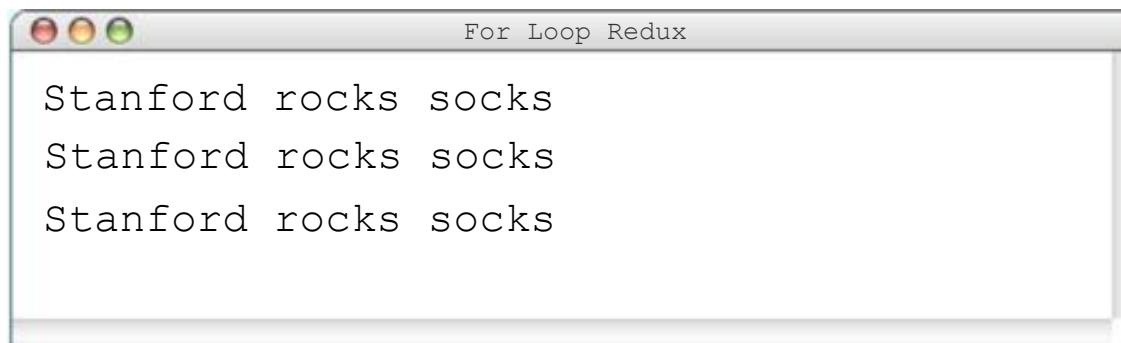
```
for(int i = 0; i < 3; i++) {  
    println("Stanford rocks socks!");  
}
```



For Loop Redux

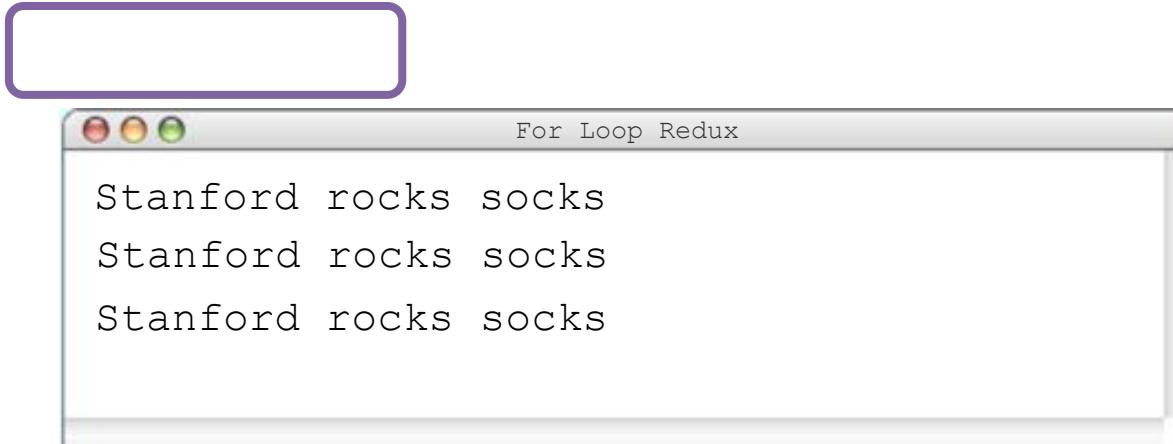
i 3

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



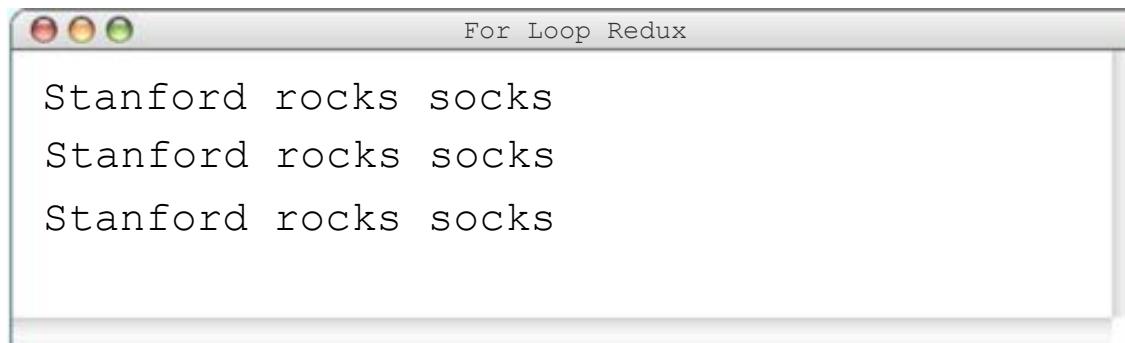
For Loop Redux

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



For Loop Redux

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



You can use the for loop variable



How would you `println` the first 100 even numbers?

Printing Even Numbers

```
PrintEven...  
0  
2  
4  
6  
8  
10  
12  
14  
16  
18  
20  
22  
24  
26  
28  
30  
32  
34  
36  
38
```



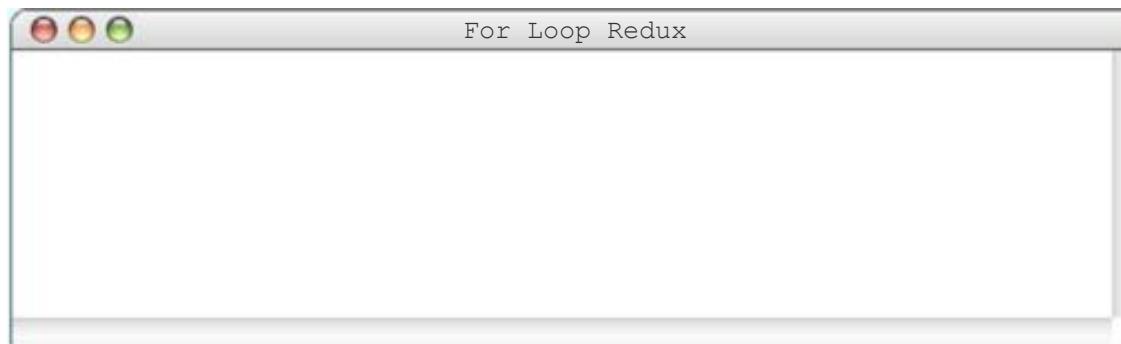
Printing Even Numbers

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



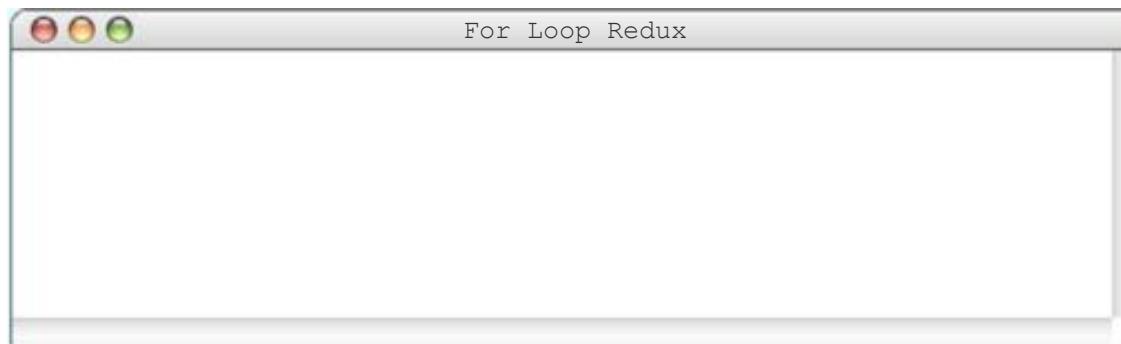
Printing Even Numbers

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



Printing Even Numbers

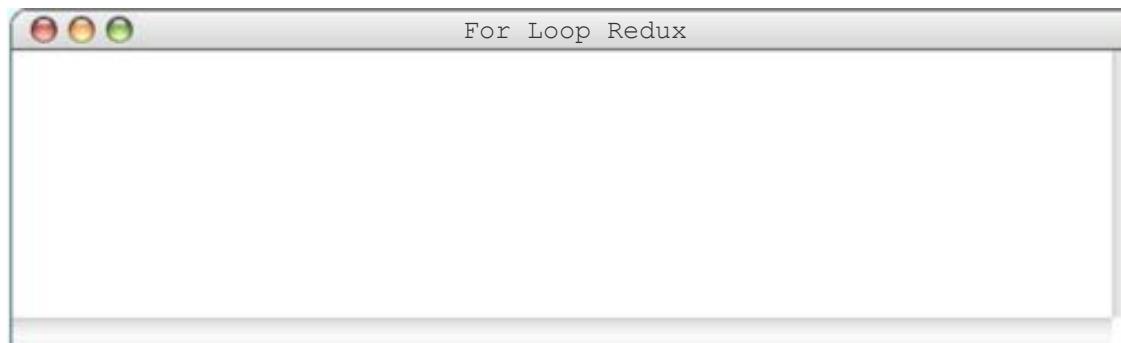
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 0

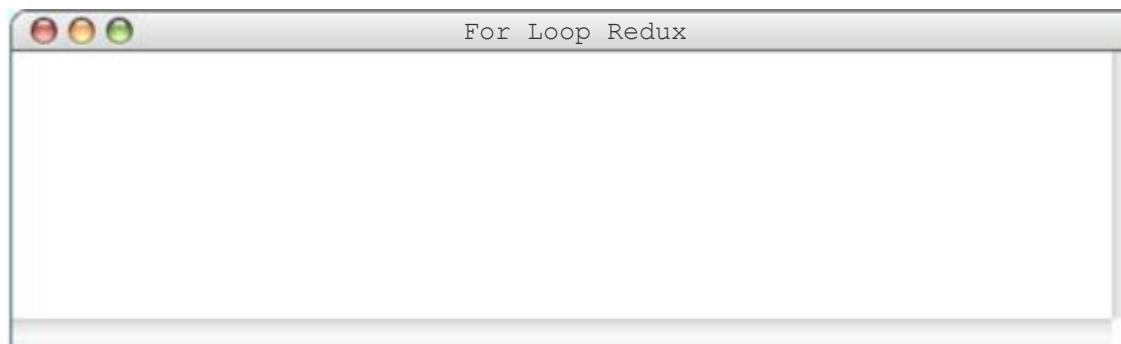
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 0

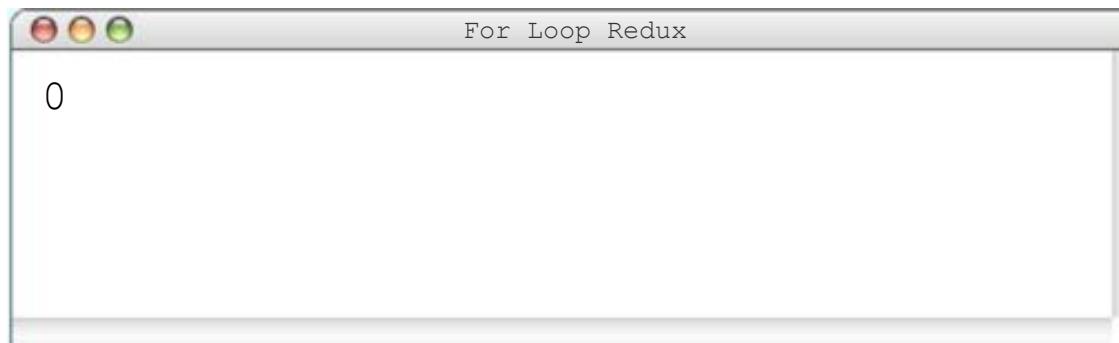
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 0

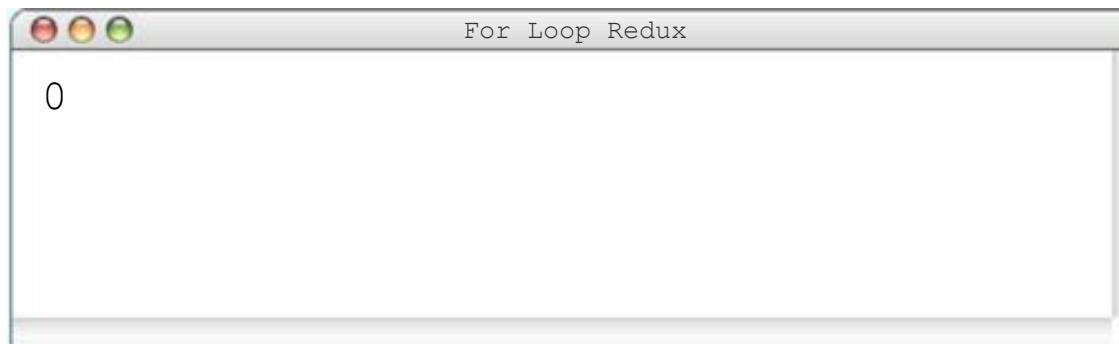
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 1

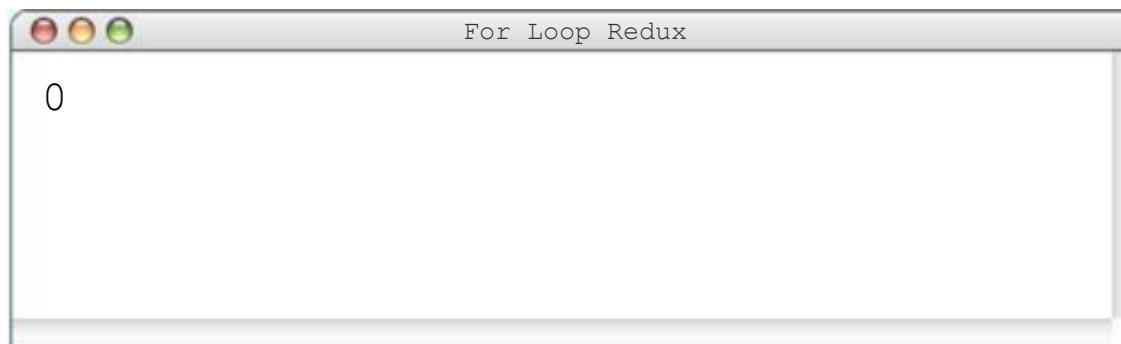
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 1

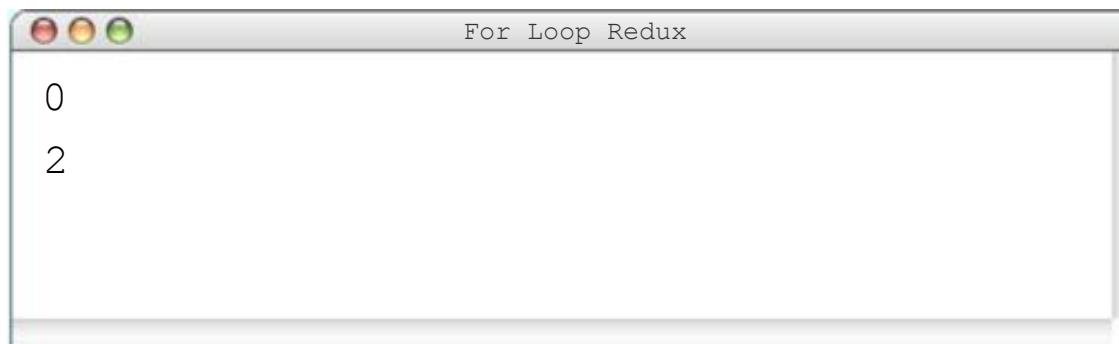
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 1

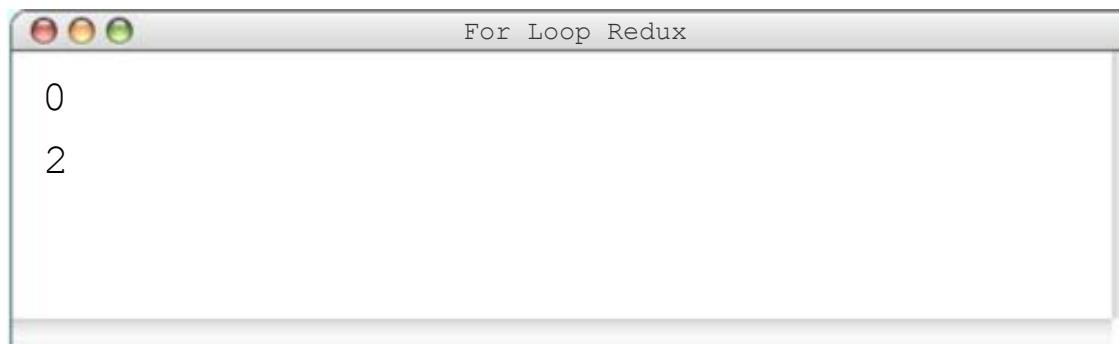
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 2

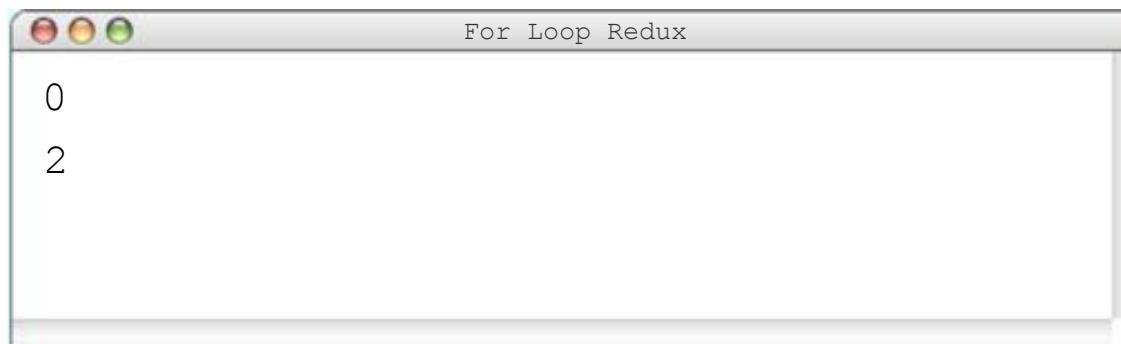
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 2

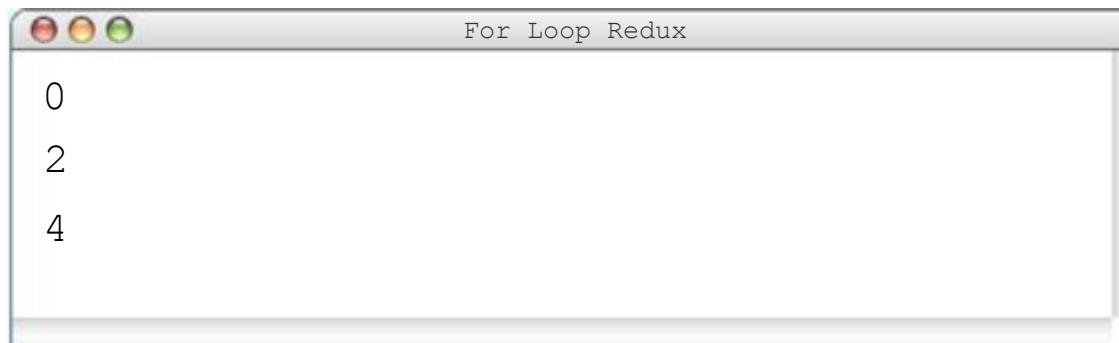
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 2

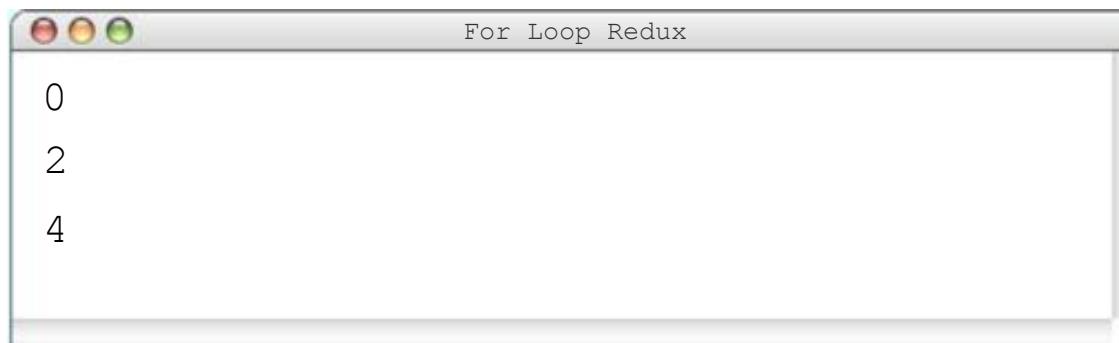
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 3

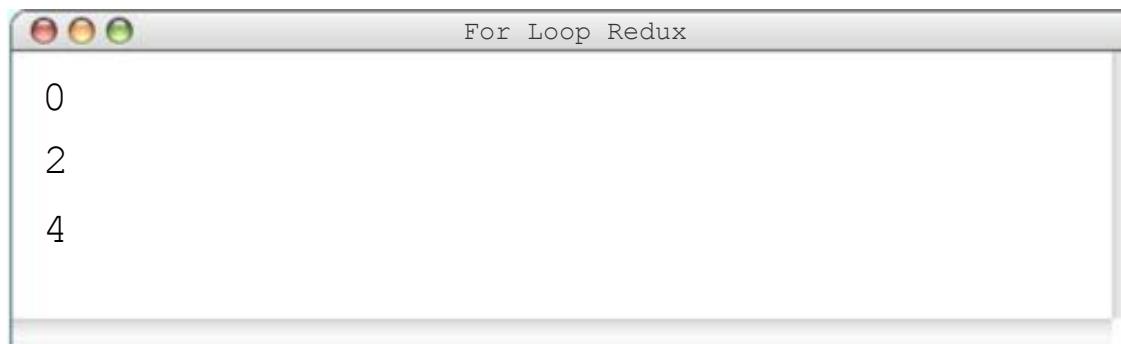
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

i 3

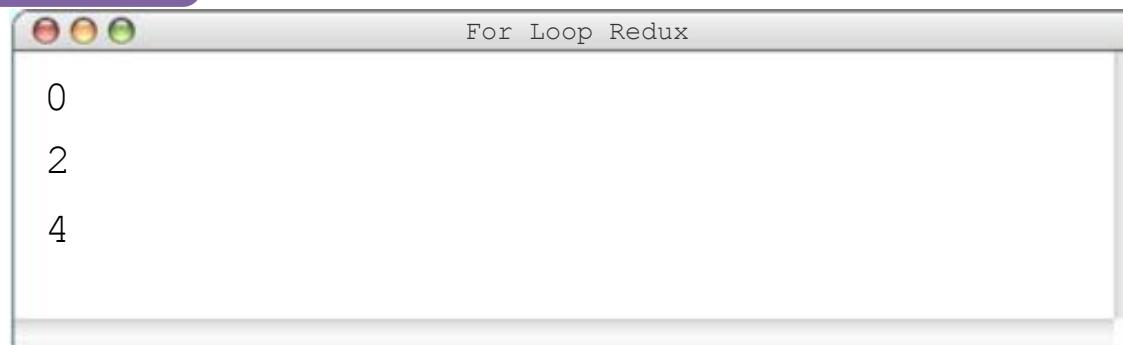
```
for(int i = 0; i < 3; i++) {  
    println(i * 2);  
}
```



Printing Even Numbers

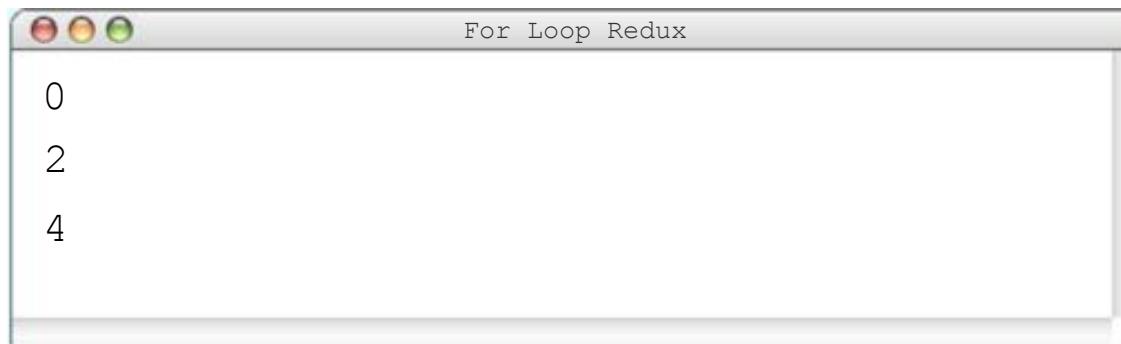
i 3

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



Printing Even Numbers

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



A Variable love story

By Chris

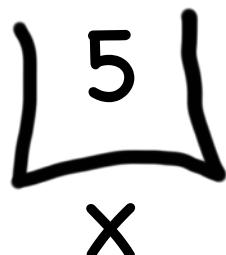
Piech, CS106A, Stanford University



Once upon a time...

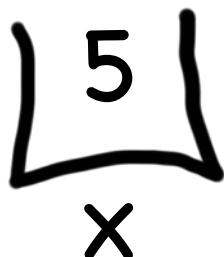
X was looking for love!

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



X was looking for love!

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



X was looking for love!

int x = 5;

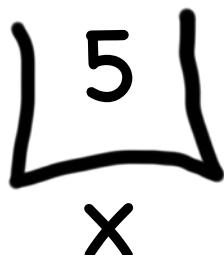
x was definitely
looking for love

```
if(lookingForLove()) {
```

int y = 5;

}

```
println(x + y);
```



And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

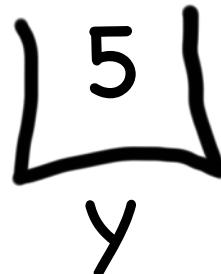
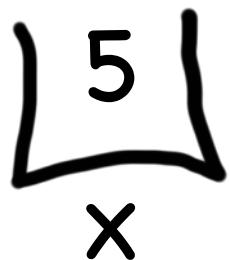
5
x

5
y



And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



Hi, I'm y

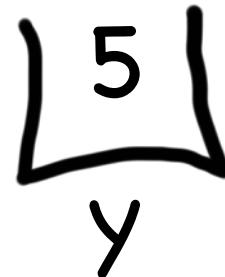
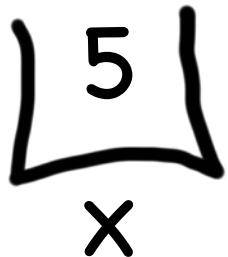


“Wow!”

And met y

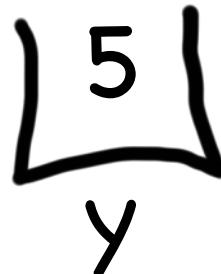
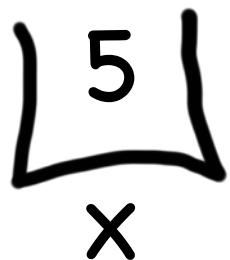
```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

Wow



And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



We have so much
in common



And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

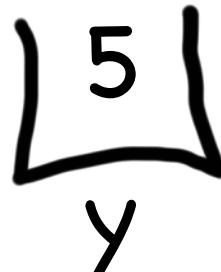
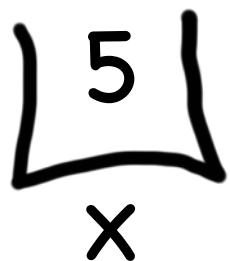
5
y

We both have
value 5!



And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



Maybe one day
we can...



And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y

println together?



They got along

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y



It was a beautiful match...

But then tragedy struck.

Tragedy Struck

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y



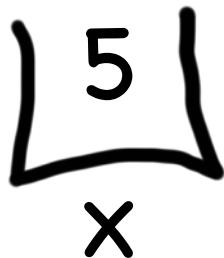
Tragedy Struck

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



Tragedy Struck

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

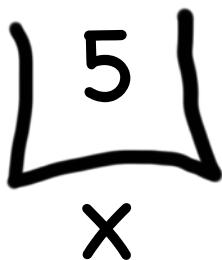


Noooooooooooooo!

You see...

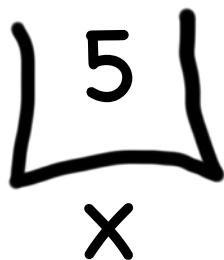
When a program exits a code block...

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



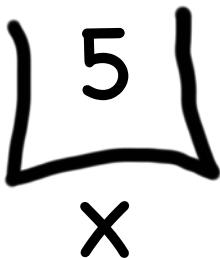
All variables declared inside that block..

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



Get deleted from memory!

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



Since y was declared in the if-block

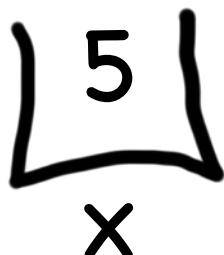
```
int x = 5;
```

```
if(lookingForLove()) {
```

```
    int y = 5;
```

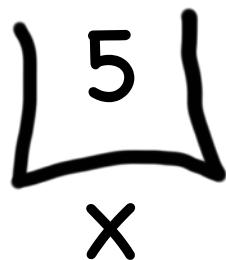
```
}
```

```
println(x + y);
```



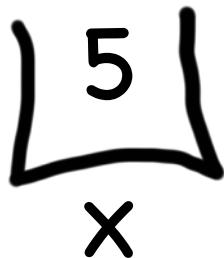
It gets deleted from memory here

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



And doesn't exist here

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

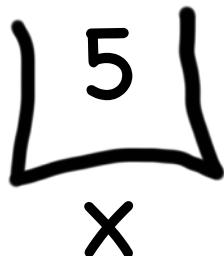


And doesn't exist here

```
int  
if(l  
    Error. Undefined  
        variable y.
```

{}

```
println(x + y);
```



The End

Sad times ☹

Variables have a lifetime (called scope)

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```



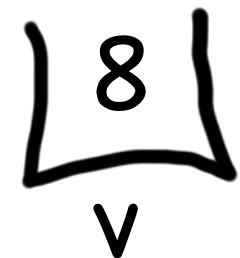
Variables have a lifetime (called scope)

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```



Vars come to existence when declared

```
public void run() {  
    double v = 8; ← Comes to life here  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```

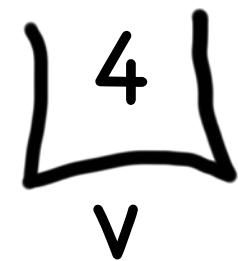


Live until end of their code block

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```

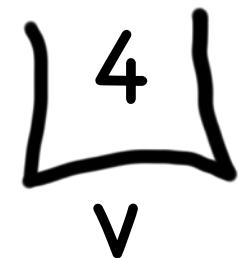


This is the **inner most** code block in which it was declared....



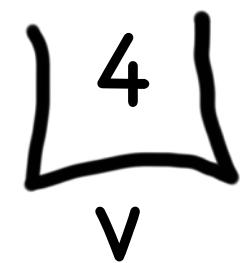
Live until end of their code block

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;      ← Still alive here...  
        ... some code  
    }  
    ... some other code  
}
```



Live until end of their code block

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```



It dies here (at the end of its code block)



Live until end of their code block

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```



It dies here (at the end of its code block)



Example 2

```
public void run() {  
    ... some code  
    if (condition) {  
        int w = 4;  
        ... some code  
    }  
    ... some other code  
}
```

This is the scope of w



Example 2

```
public void run() {  
    ... some code  
    if (condition) {  
        int w = 4;  
        ... some code  
    }  
    ... some other code  
}
```

w comes to life here

w dies here (at the end of its code block)



A Variable Love story

Chapter 2

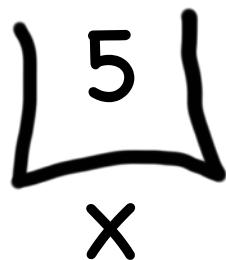
Piech, CS106A, Stanford University



The programmer fixed her bug

x was looking for love!

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
    println(x + y);  
}
```



x was looking for love...

int x = 5;

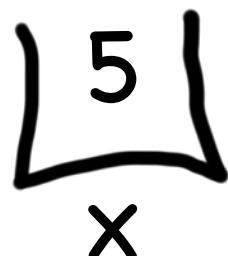
x was definitely
looking for love

```
if(lookingForLove()) {
```

int y = 5;

println(x + y);

}



x met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
    println(x + y);  
}
```

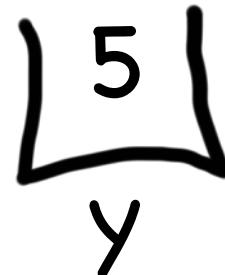
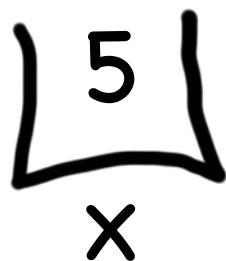
5
x

5
y



Since they were both “in scope”

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
    println(x + y);  
}
```



The story had a happy ending!

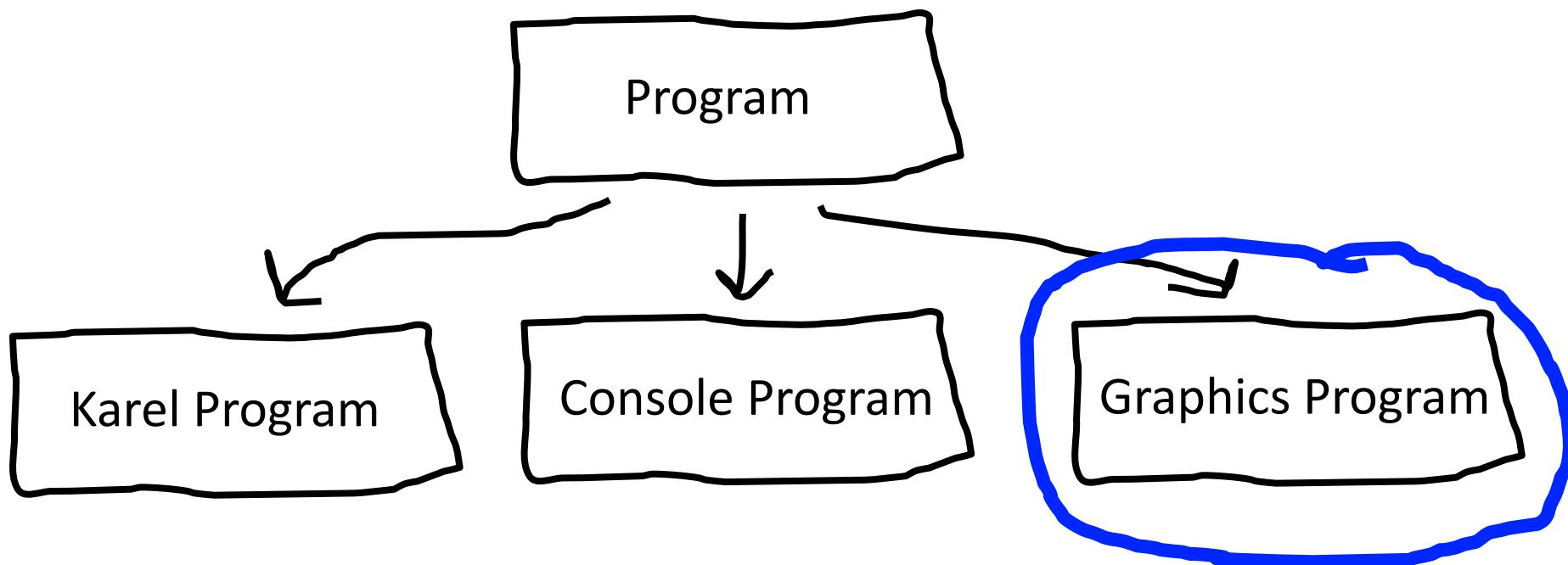
Scope Formally

- The **scope** of a variable refers to the section of code where a variable can be accessed.
- **Scope starts** where the variable is declared.
- **Scope ends** at the termination of the inner-most code block in which the variable was defined.
- A **code block** is a chunk of code between { } brackets

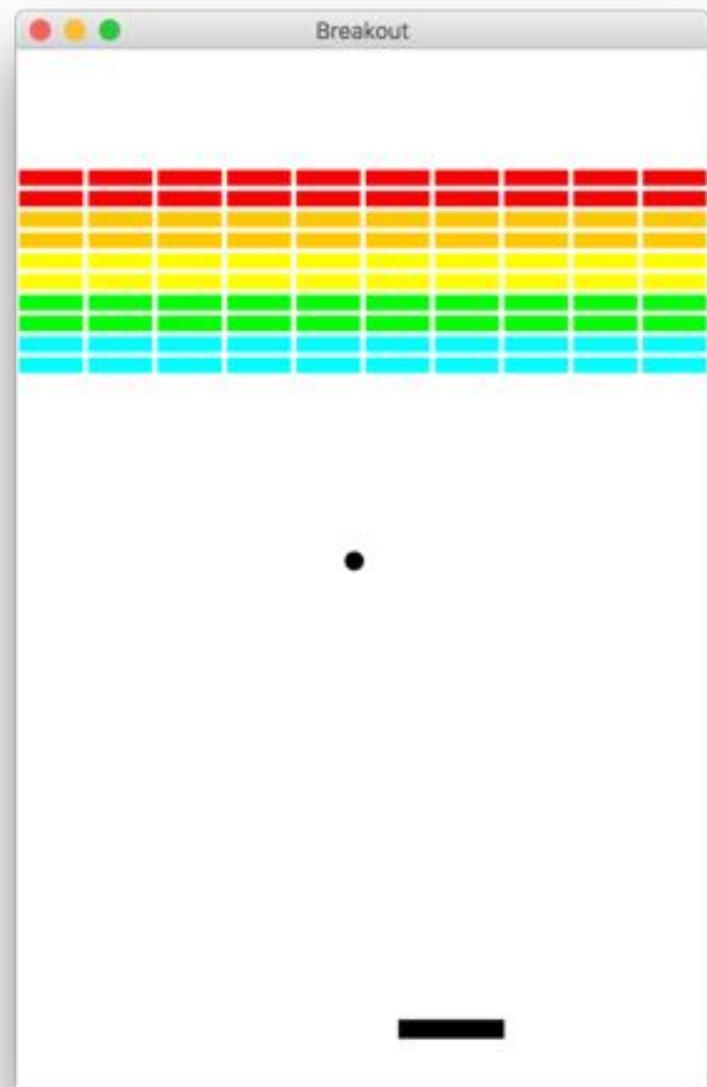
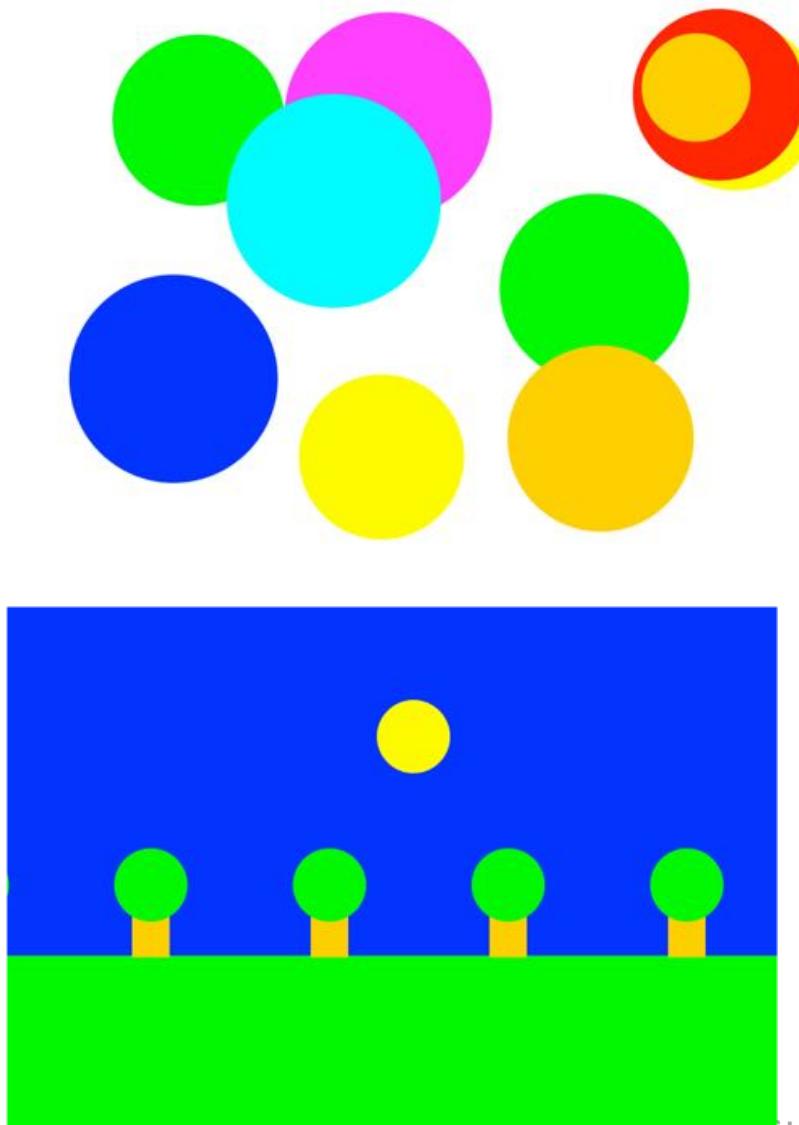


Back to our regularly scheduled program...

Types of Programs



Graphics Programs

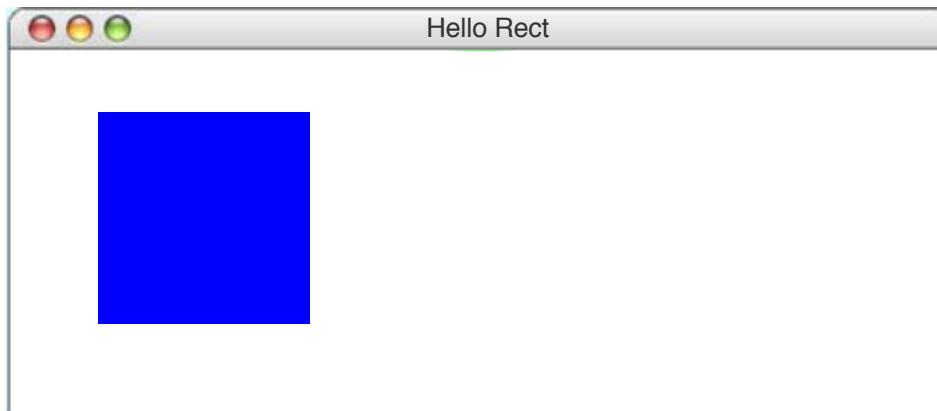


GRect

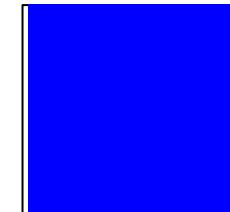
GRect is a variable type that stores a rectangle.

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);  
}
```



rect

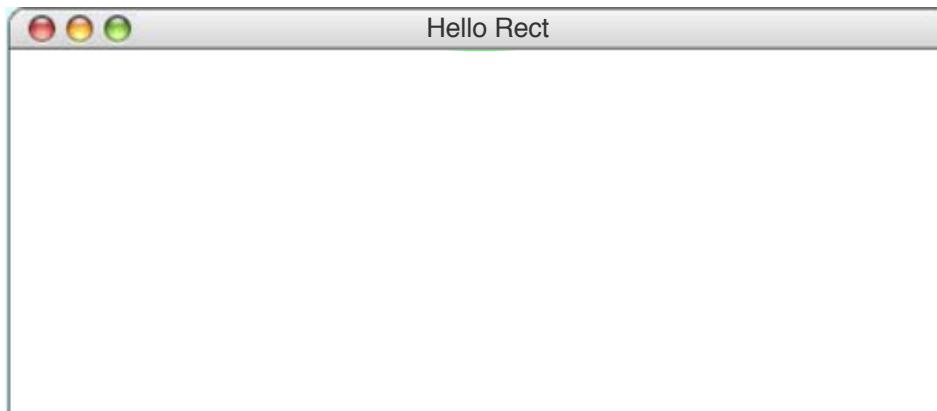
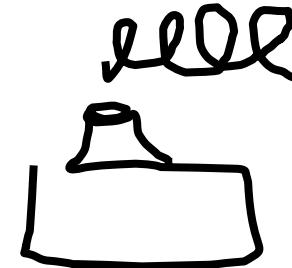


GRect

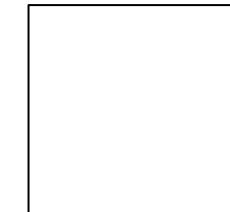
GRect is a variable type that stores a rectangle.

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);  
}
```



rect

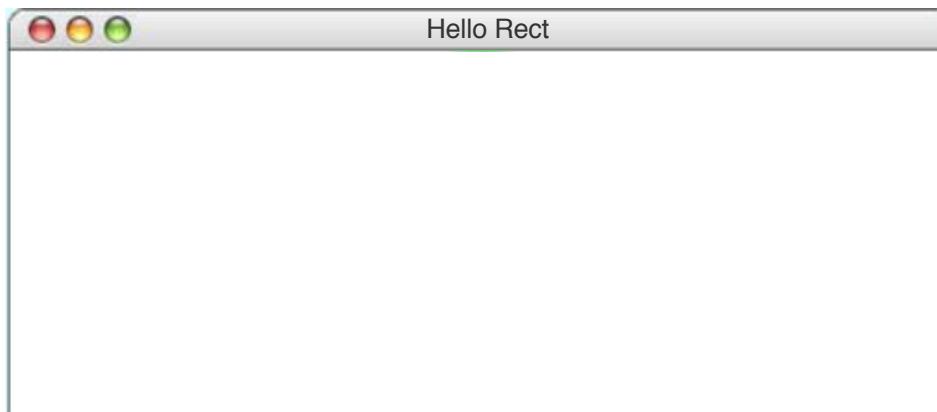


GRect

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);  
}
```



rect

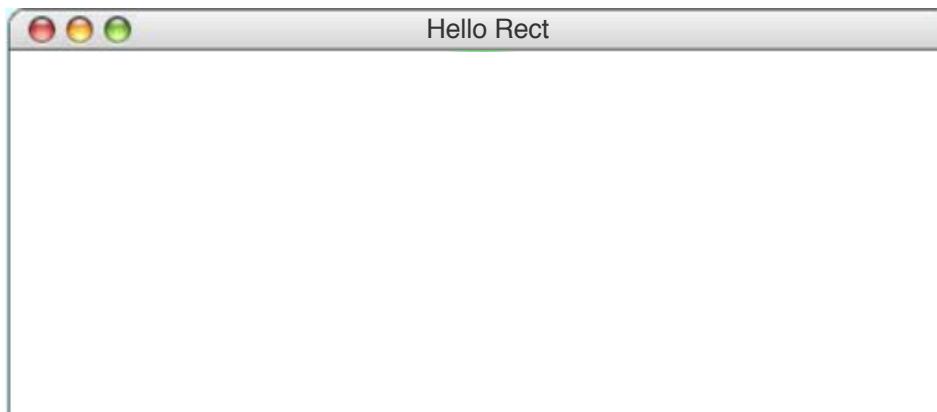


GRect

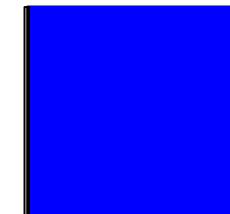
GRect is a variable type that stores a rectangle.

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);  
}
```



rect

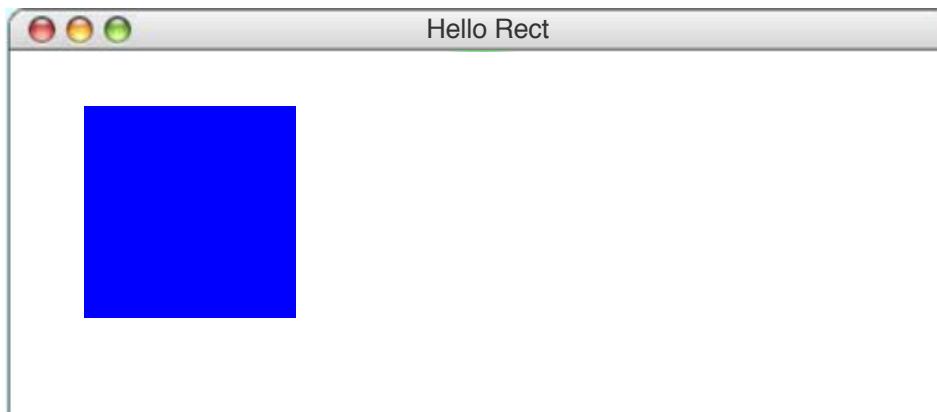


GRect

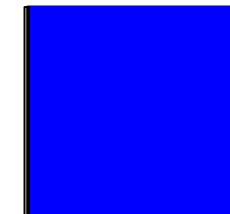
GRect is a variable type that stores a rectangle.

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);  
}
```



rect



Graphics Coordinates

0,0

x 40,20

x 120,40

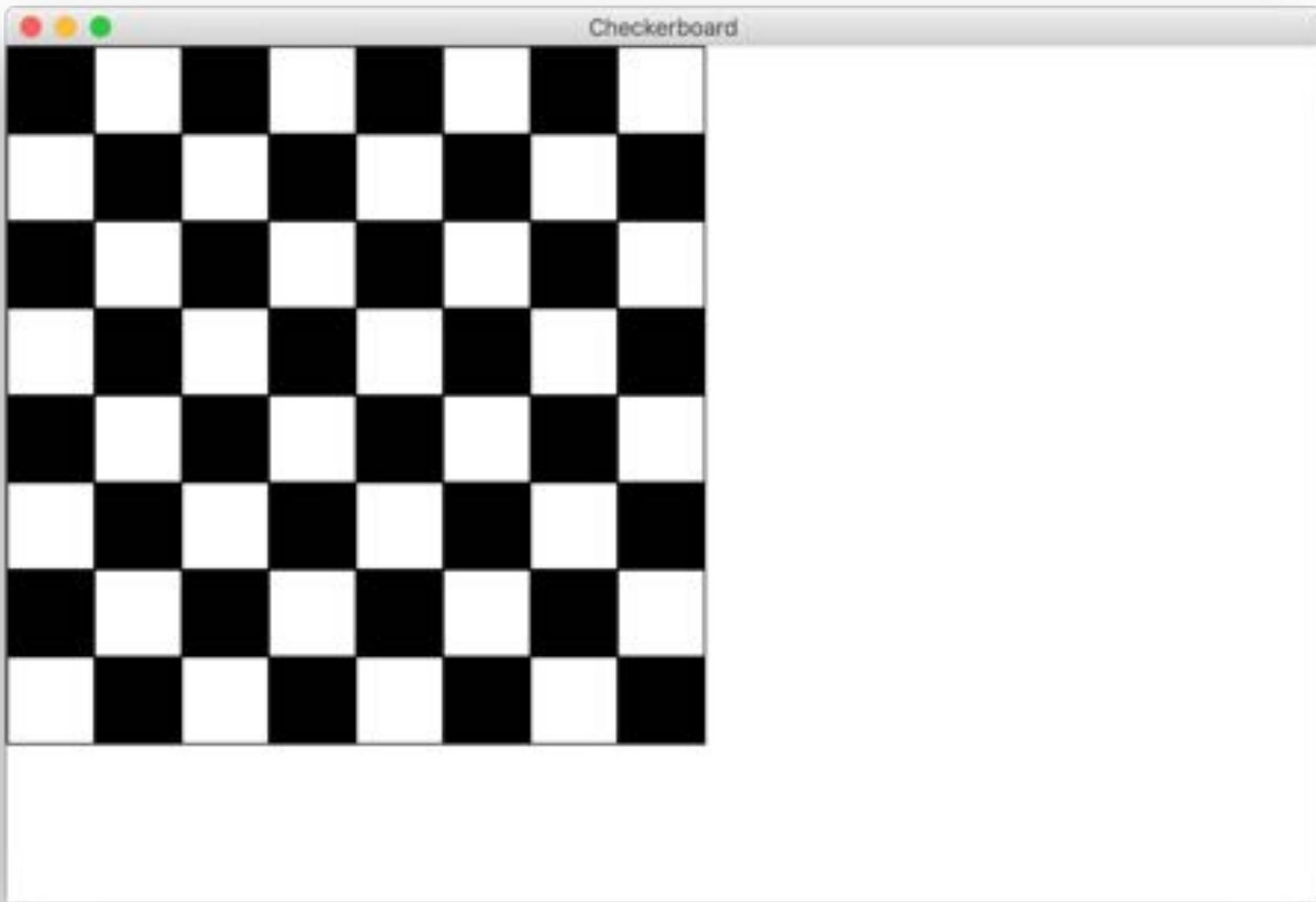
x 40,120

getWidth();

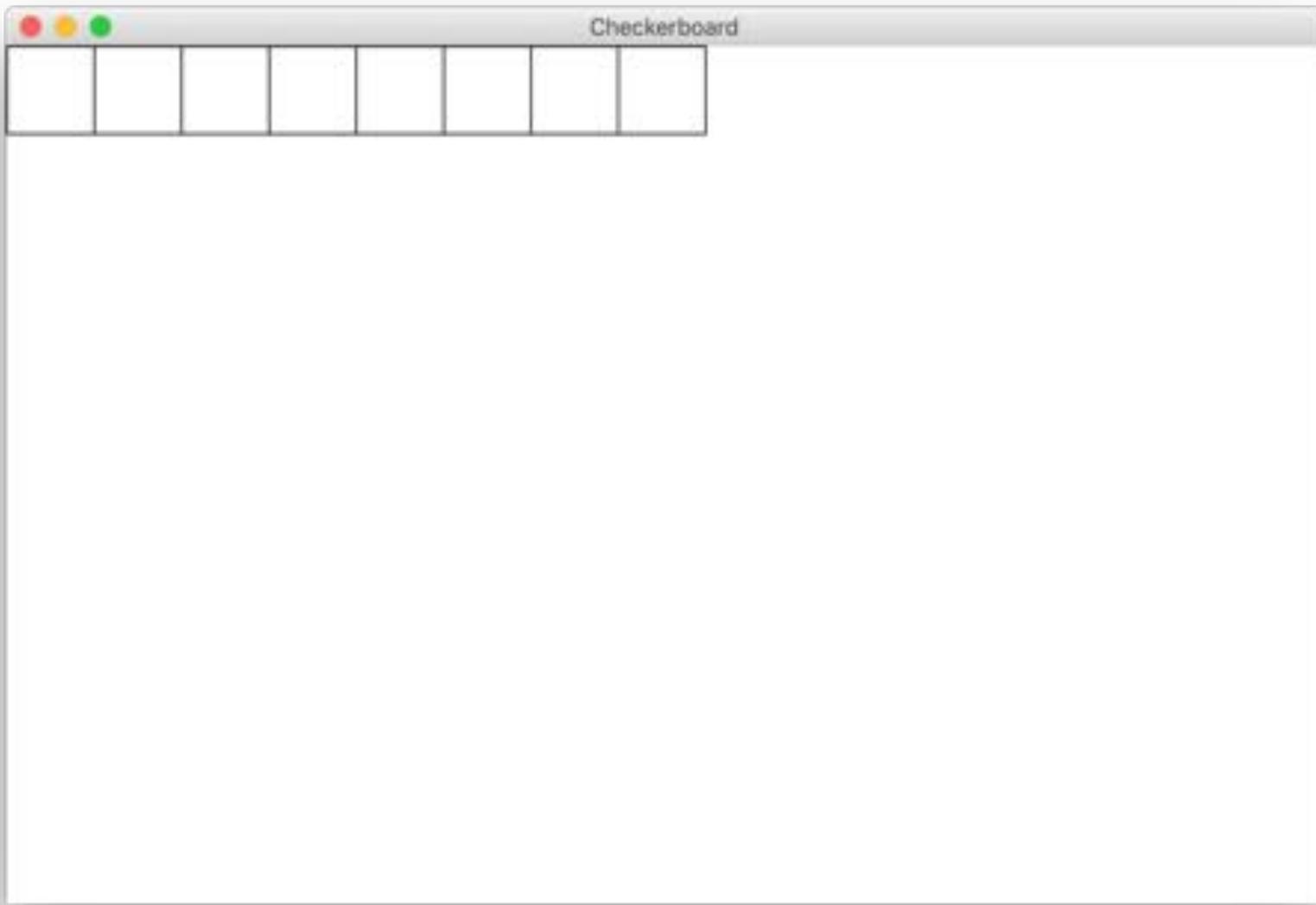
getHeight();



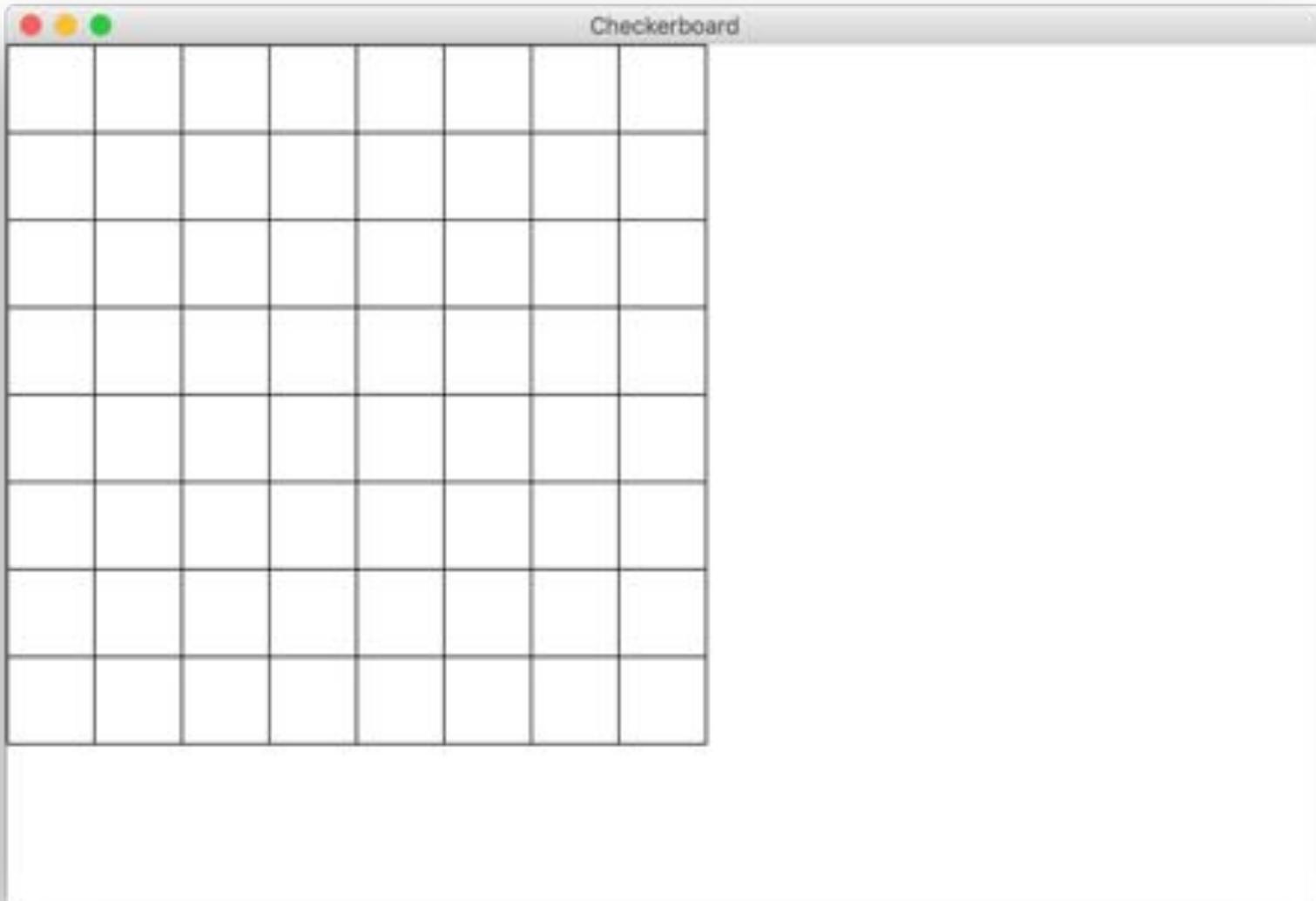
Goal



Milestone 1



Milestone 2



Milestone 3

