

Loops

CF1

Office Hours

- Gabe: Tuesday/Thursday 1–2 and 3:30–4:30 in ATLS 225
- Josie: Tuesdays 12–1 in ATLS 225
- Wolfgang: Wednesdays 4–5 in ATLS 225
- Lindsey: Wednesdays 11–12 in ATLS 225
- Lila: Thursdays 3:15–4:15 in ATLS

Anonymous CF1 poll - in Canvas

Week 4 -- Loops			✓	+	⋮
⋮	📄	Guest Lecture 3: Peter Gyory Sep 22 10 pts	✓	⋮	
⋮	🔗	Guest Lecture Recording: Peter Gyory (PW: 0N!cucx3) 📄	✓	⋮	
⋮	🔗	Peter's Slides 📄	✓	⋮	
⋮	🚀	Quiz 4: Loops Sep 19 10 pts	✓	⋮	
⋮	🚀	Week 4 Lecture 2 Attendance Sep 19 1 pts	✓	⋮	
⋮	🚀	Week 4 Recitation Attendance Multiple Due Dates 1 pts	✓	⋮	
⋮	📄	Homework 4: Patterns Sep 22 10 pts	✓	⋮	
⋮	🔗	(Optional Poll) How CF1 is going for you? 📄	✓	⋮	



“Repeat Loop”

This isn't a canonical term, but it makes total sense!

```
while (someCondition) {  
    doSomething();  
}
```

As long as someCondition is true, it will doSomething. You can use this to repeat for *ever*.

“Repeat Loop”

P5 does this with draw()


```
setup(); // it does this one time  
while (true) {  
    draw(); // it does this infinity times  
}
```

While Loop

Anatomy Lesson


```
while (count < limit) {  
    print("Iteration: " + count);  
    count = count + 1;  
}
```

always start with the 'while' keyword



```
while (count < limit) {  
    print("Iteration: " + count);  
    count = count +1;  
}
```


then put a boolean expression inside parentheses



```
while (count < limit) {  
    print("Iteration: " + count);  
    count = count + 1;  
}
```


then put any number of statements inside curly braces

```
while (count < limit) {  
    print("Iteration: " + count);  
    count = count + 1;  
}
```



While Loop

When to use? When to not use?

- Use it when you loop as long as some condition is met but you don't know in advance how many times
- Advice: be sure it is possible for your condition to eventually become false.
 - (Unless you actually want it to loop forever, which isn't rare)
- Don't use while loops when you want to do something a set number of times

For Loop


Anatomy Lesson

```
let limit = 10;  
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```

always starts with the 'for' keyword

```
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```

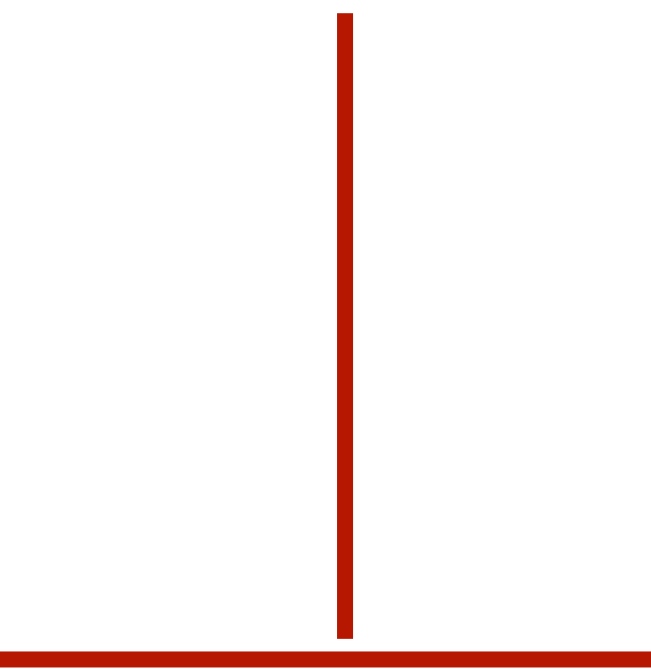
then inside parentheses we will have three statements
that control how the loop will do its thing:



```
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```

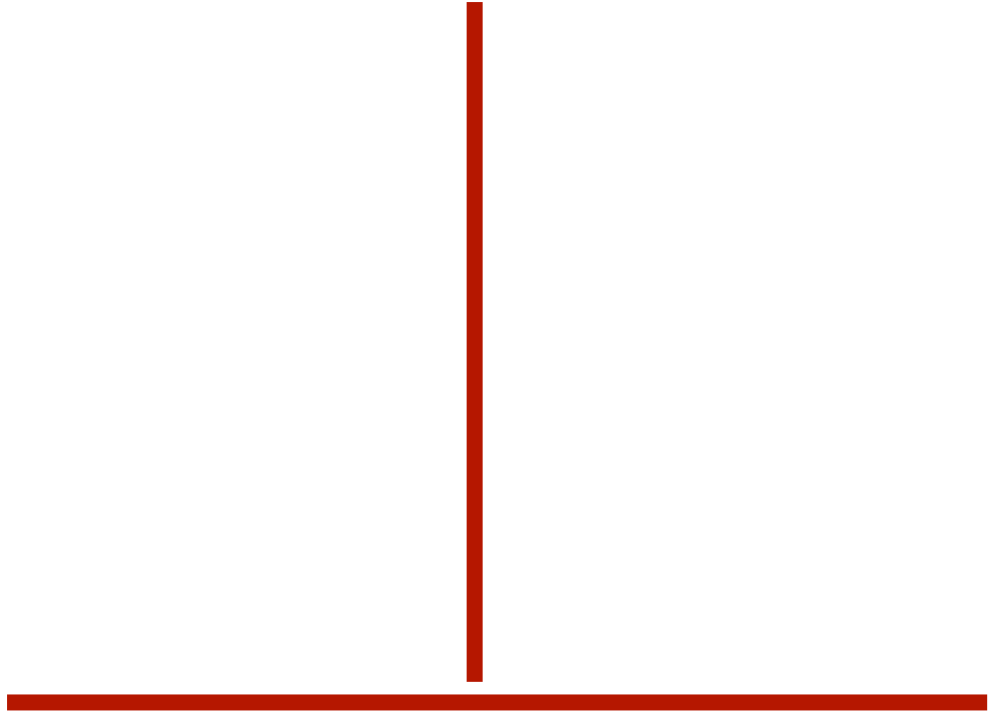
first statement is run one time, before the loop's body is executed.

This is almost always used to initialize a counting variable.



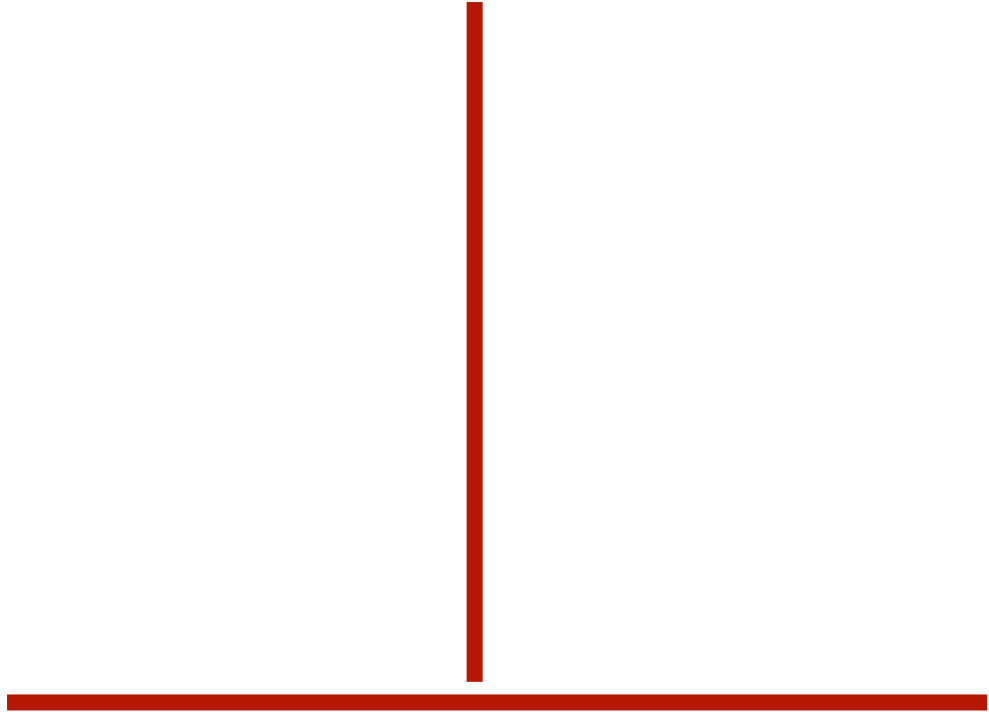
```
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```

the next statement (the one in the middle) is a boolean expression
that tells us if we should run the body's statements.



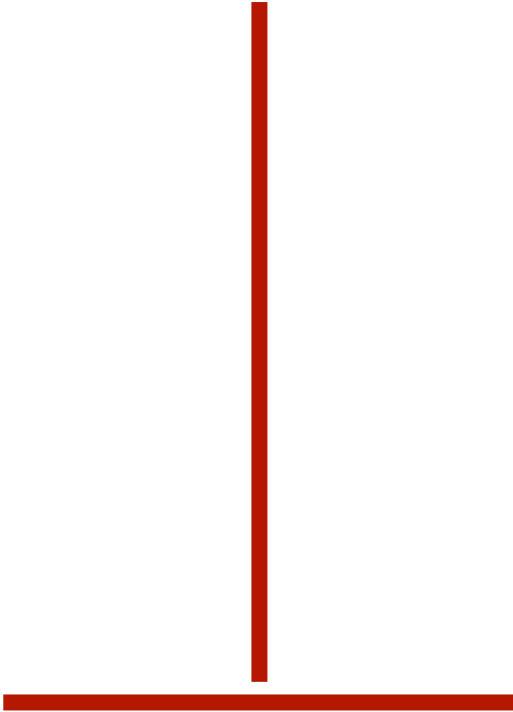
```
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```

it is possible that this boolean expression is false on the very first time. E.g. in this case, 'limit' might be negative, so nothing would print out.



```
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```


the last statement is executed immediately after the body has finished running. Typically used to increment or decrement your counting variable.



```
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```

Last we have the body of the for-loop. This is a simple one with one statement, but there could be lots of statements too.

```
for (let count = 0; count < limit; count++) {  
    print("Iteration: " + count);  
}
```

When & when not to use for-loops

- For loops are good for running through lists of items (we haven't covered lists yet this semester)
- For loops are also good for doing something some number of times, like counting down from ten to one, or drawing a single window in a whole building full of them
- There are other looping structures that we will cover later along with lists
- For-loops are often more complicated than you need them to be. But they are true workhorses, and can do anything.

do-while is also a thing

```
do {  
    print("Iteration: " + count);  
    count++;  
} while(limit < count);
```

The difference here is that the body of the loop is guaranteed to execute at least one time. Its boolean expression is tested at the *end* of its body.

do-while is also a thing

```
let flip = 'tails';  
do {  
    flip = flipACoin();  
} while(flip !== 'tails');
```

Just be sure you know that the condition will eventually complete, or you'll get an infinite loop.

Nested loops are also a thing

```
let numRows = 4;
let numCols = 8;
for (let i = 0; i < numRows; i++) {
    for (let j = 0; j < numCols; j++) {
        print("Row", i, "Col", j);
    }
}
```

You can nest loops as much as you want.

What do you think this code will print out? See if you can predict it before trying it out.

```
for (let i = 0; i < numRows; i++) {  
    print("Col", i);  
    for (let j = 0; j < numCols; j++) {  
        print("Row", i, "Col", j);  
    }  
}
```

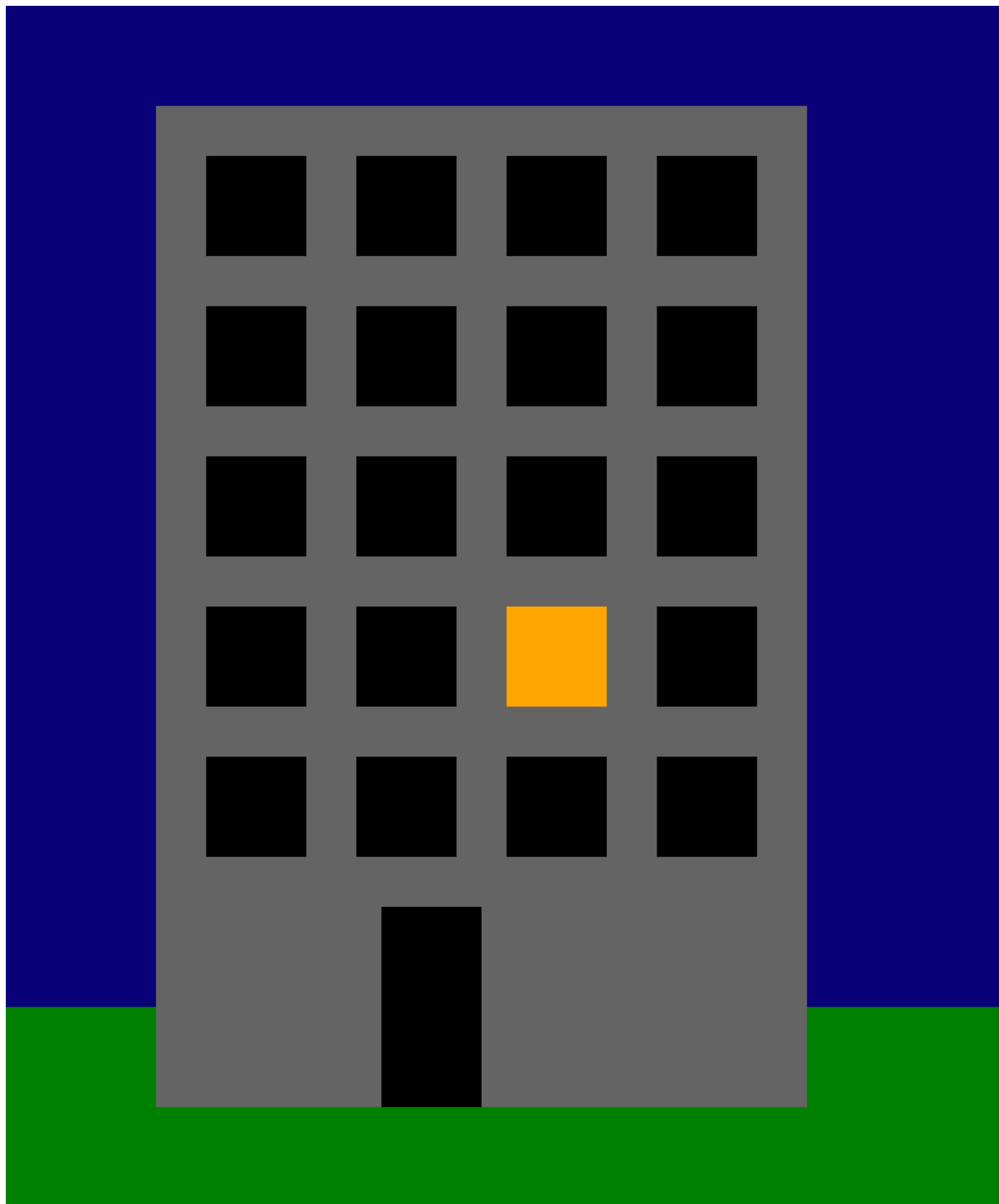
What's the intuition behind loops?

Repetition, goal-seeking

- If you're writing code and find that you're writing the same thing over and over again, a loop *might* be a way to simplify things
- Or if you want to run the same code repeatedly to zone in on a solution
- Basically, if you think about repetition at all, a loop is probably a good bet

Reuse & Rewrite Previous HW w/Loops

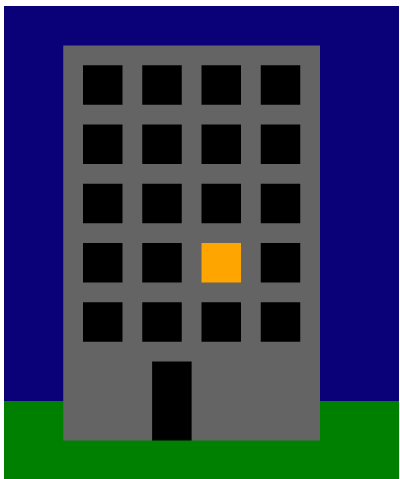
One of the three virtues of a good programmer is laziness



```
for (let floor = 0; floor < numFloors; floor++) {  
  for (let room = 0; room < numRooms; room++) {  
    renderWindow(floor, room);  
  }  
}
```

Reuse & Rewrite Previous HW w/Loops

One of the three virtues of a good programmer is laziness



- In general, you should feel free to re-use last week's homework as a basis for this week.
- Maybe don't do this *every* week, just to keep things fresh
- But there's definitely value in improving your own code over a longer time period.

Editor Hints

- Control Slash to comment & uncomment
- Stop your sketch to conserve battery usage (p5 kills your battery OMG)