CONDITION

true

false

LOOP_BODY
Loops

- Provide a way to run code multiple times
- DRY - Don't Repeat Yourself
- Two types of loops
  - For Loops
  - While Loops
Bart could have used a loop
For vs. While

For Loop

- **Fixed number of iterations**
- Simpler
  - Automatic iteration
- Use cases:
  - Use whenever possible, usually for iterating over a fixed size (e.g. counting to X)

While Loop

- **Unknown number of iterations**
- More complex
  - Manual iteration
- Use cases:
  - Run for an indeterminate amount of iteration
  - Reading files of undetermined size
For Loop

```java
for (int i=0; i<10; i++) {
    System.out.println("Count:" + i);
}
```

- i - short for index or iterator
- Declare the iterator
- Set the range for that iterator
  - Watch for off by one errors
- Set the value to iterator for
While Loop

```java
int i = 0; // initialize loop counter
while (i < 10) {
    System.out.println(i);
    i = i + 1; // increment loop counter
}
```

0) initialize loop counter
   (done in the for loop declaration in 'for' loops)
1) check if condition is still true
2) increment counter
   (done by the loop in 'for' loops)
Are you stuck in an infinite loop?

NO

YES
Infinite Loop

while (true) {
    System.out.println("Hello");
}
Breaking out of a Loop

```java
while (true) {
    System.out.println('inside')
    break;
}
```
int i = 0;
while (i < 100) {
    i = i + 1;
    if (i % 2 == 0) {
        continue;
    }
    System.out.println("Odd: "+ i);
}
Nested For Loops

// for the numbers from 1-9
// print that number n-times
// where n is the number from 1-9
// e.g. the number 3 would print “333”
for (int line=1; line<10; line++) {
    // outer loop
    for (int col=1; col<line; col++) {
        // inner loop
        System.out.print(line);
        // print char
    }
    System.out.println();
    // print newline
}
Stepping through an example

- Using the debugger
- Using print statements