Loops

- **while** loops
- **do** loops
- **break** and **continue**
- **for** loops

Introduction to Java

See Also

http://www.tutorialspoint.com/java/java_loop_control.htm

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/for.html

https://www.youtube.com/watch?v=efvZmFd1prA

The while Loop

```java
int a = 0;

System.out.println("Start");

while(a<5)
{
    System.out.println(a);
    a = a + 1;
}

System.out.println("Done");
```

```
Start
0
1
2
3
4
Done
```

The while Loop

```java
int a = 0;

System.out.println("Start");

while(a<5)
{
    a = a + 1;
    System.out.println(a);
}

System.out.println("Done");
```

```
Start
1
2
3
4
5
Done
```

The while Loop

```
int a = 3;

System.out.println("Start");

while(a<=5)
{
    a = a + 1;
    System.out.println(a);
}

System.out.println("Done");
```
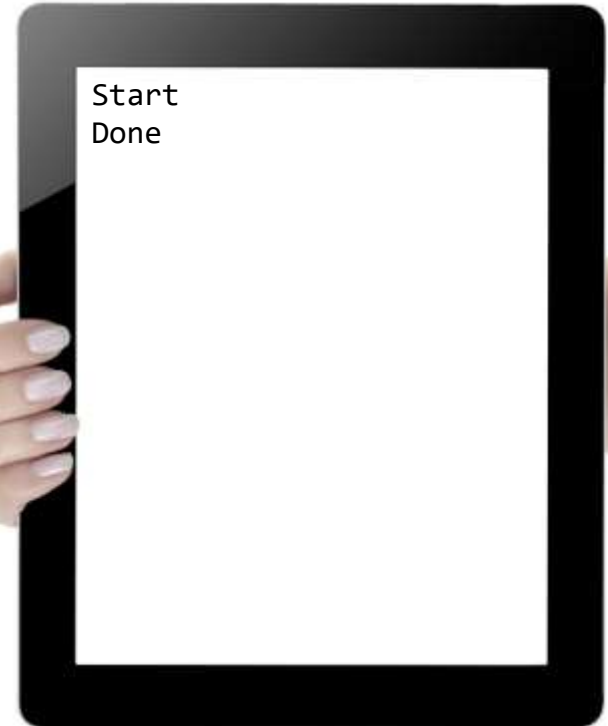
```
Start
4
5
6
Done
```

The while Loop

```java
int a = 10;

System.out.println("Start");

while(a<=5)
{
    a = a + 1;
    System.out.println(a);
}

System.out.println("Done");
```

```
Start
Done
```

The while Loop

```java
int a = 0;

System.out.println("Start");

while(a<=5)
{
    int b = a + 1;
    System.out.println(b);
}

System.out.println("Done");
```

```
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
```

The do Loop

```java
int a = 0;

System.out.println("Start");

do
{
    System.out.println(a);
    a = a + 1;
}
while(a<5);

System.out.println("Done");
```

```
Start
0
1
2
3
4
Done
```

While vs. Do

- The while loop may never execute
- The do loop executes at least once
- We usually mean "do" but end up using "while"
  - The "do" loop takes more typing
  - We are thinking about the expression before we start typing the body of the loop

```
10: getstatic      #16
13: iload_1
14: invokevirtual #30
17: iinc           1, 1
20: iload_1
21: iconst_5
22: if_icmplt      10
```
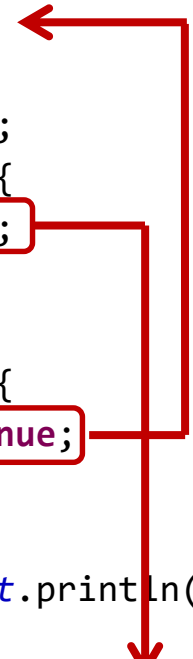
```
10: goto           23
13: getstatic      #16
16: iload_1
17: invokevirtual #30
20: iinc           1, 1
23: iload_1
24: iconst_5
25: if_icmplt      13
```

"break" and "continue"

```java
int a = 0;
int b = 10;
int c = 5;


while(true) {


    a = a + 1;
    if(a==b) {
        break;
    }


    if(a==c) {
        continue;
    }


    System.out.println(a);
}
```
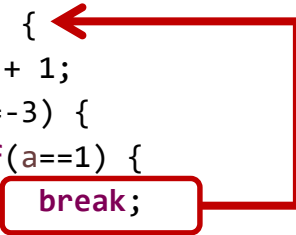
- break takes you out of the loop
- continue takes you to the expression (top or bottom)

```
1
2
3
4
6
7
8
9
```

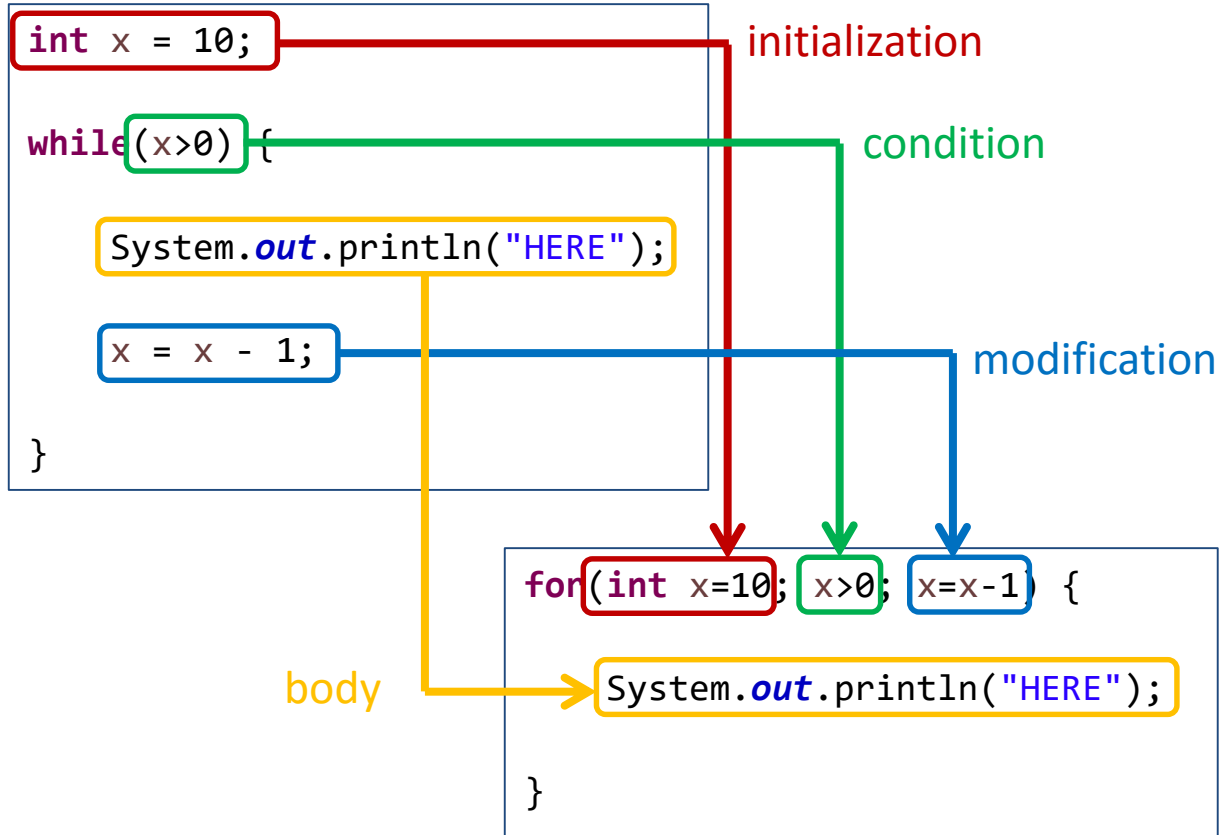# "break" and "continue"

- ONLY for loops (and switch)
- Will not break an if/else block
- Usually find them inside an "if"
- Look up the code to find the "while" or "do"

```
while(true) {
    if(a==2) {
        while(b<0) {
            b = b + 1;
            if(b==-3) {
                if(a==1) {
                    break;
                }
            }
        }
    }
}
```

"for" loops

```
int x = 10;

while (x>0) {

    System.out.println("HERE");

    x = x - 1;

}
```

initialization

condition

modification

body

```
for (int x=10; x>0; x=x-1) {

    System.out.println("HERE");

}
```

"for" defaults

- All three pieces are optional (the semicolons are not)
- The initialization defaults to "nothing"
- The modification defaults to "nothing"
- The condition defaults to "true"

```java
for(;;) {

    System.out.println("HERE");

}
```

```java
while(true) {

    System.out.println("HERE");

}
```

Multiple statements

- Use commas to separate multiple statements in initialization or modification

```
for(int a=2, b=3; a<5; a++,b++) {

    System.out.println("HERE");

}
```

```
int a=2;
int b=3;

while(a<5) {

    System.out.println("HERE");

    a++;
    b++
}
```

Tinkering

- Code up a loop to print even numbers from 2 to 20. Try out the modulo operator here: "a%2"

- What does a "do { } while(false);" loop do? What does a "break" inside the loop mean?

- Write a for loop to count x from 0 to 20. Put another for loop inside the first loop to count y from 0 to x. Print y inside the inner loop.