

# While Loops

By: Megan Avery

Updated October 2018



# Before we begin

- Make sure you have Java & Java JDK downloaded on your computer
  - can run **java -version** in terminal to check
- Make sure you have IntelliJ downloaded on your computer
- *Suggested:* Watch previous Java tutorials

# Vocab Reminder

**loop variable:** the variable that keeps track of the loop counter

**increment:** add to a variable

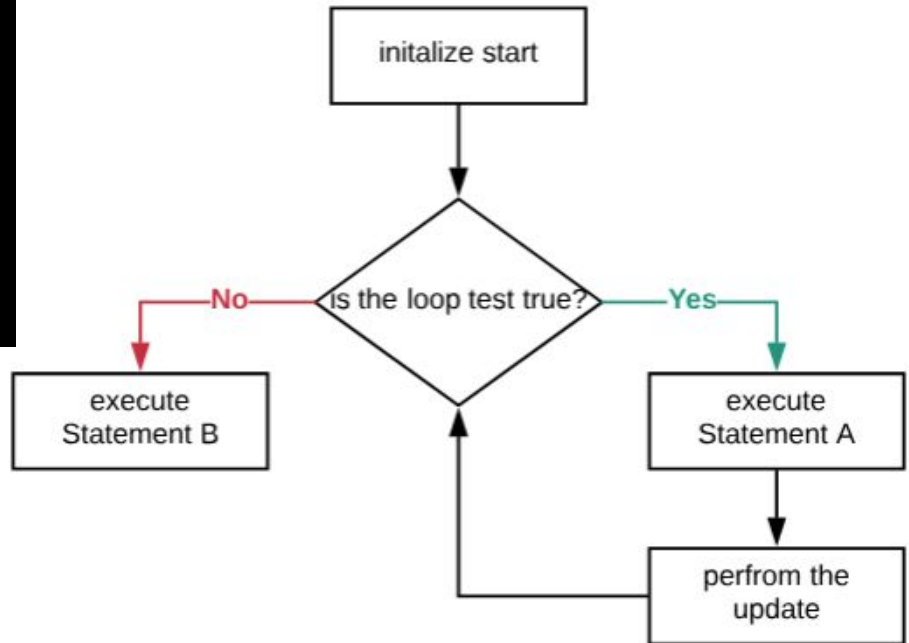
**decrement:** subtract from a variable

# What is a while loop?

A **while loop** is a **control structure** that lets you repeat a piece of code as long as the loop test remains true.

# Code and Diagram of While Loop

```
int i = 0;  
while (i < 5) {  
    System.out.println("*");  
    ++i;  
}
```



# Common Issues

- Loops that never run because the loop test is false from the beginning

```
int i = 0;
while (i > 5) {
    System.out.println("*");
    ++i;
}
```

- Loops that never run because there is a semi colon after the first line

```
int i = 5;
while (i > 0); {
    System.out.println("*");
    --i;
}
```

- Loops that run forever because the loop condition is always true
  - Common way this happens, forget the update in the while loop

# “Break”ing a While Loop

Putting a **break;** inside of a while loop, just like with a for loop, will force the loop to stop executing. This can be good if you are using your loop to find the first occurrence of something or if some condition (besides the loop test) should cause your loop to stop.

# Returning Inside a While Loop

Returning inside of a while loop will return from the method you are inside of as well, just like with a for loop. This means anything in your method after of the loop won't happen if the return inside of your loop gets executed.



# When to use a while loop?

If you find yourself writing a piece of code where you are repeating an action and that action should be repeated as long as some test is true.

## **Examples:**

- Continuing to play a game of rock, paper, scissors as long as player 1 is winning
- Continuing to play hangman until either the player loses or wins

# Loop Tracing

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i;
}
```

i

output

**\*\* keep track of loop variable (i) and output \*\***

## Loop Tracing

```
int i = 0; ←  
while (i < 5) {  
    System.out.println("*");  
    ++i;  
}
```


i  
0

output

**\*\* initialize i to 0 \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i;
}
```



i  
0

output  
\*

**\*\* i is less than 5, print out an asterisk \*\***

## Loop Tracing

```
int i = 0;  
while (i < 5) {  
    System.out.println("*");  
    ++i; ←  
}
```


i  
01

output  
\*

**\*\* increment i by 1, i is now 1 \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i;
}
```



i  
01

output  
\* \*

**\*\* i is less than 5, print out an asterisk \*\***

## Loop Tracing

```
int i = 0;  
while (i < 5) {  
    System.out.println("*");  
    ++i; ←  
}
```

i  
0 → 2


output  
\* \*

**\*\* increment i by 1, i is now 2 \*\***



## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i;
}
```



i  
0 1 2

output  
\* \* \*

**\*\* i is less than 5, print out an asterisk \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i; ←
}
```


i  
~~0~~123

output  
\* \* \*

**\*\* increment i by 1, i is now 3 \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i;
}
```



i  
~~0~~123

output  
\*\*\*\*

**\*\* i is less than 5, print out an asterisk \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i; ←
}
```


i  
~~0~~1234

output  
\*\*\*\*

**\*\* increment i by 1, i is now 4 \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i;
}
```



i  
~~0~~1234

output  
\*\*\*\*\*

**\*\* i is less than 5, print out an asterisk \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i; ←
}
```


i  
~~0~~12345

output  
\*\*\*\*\*

**\*\* increment i by 1, i is now 5 \*\***

## Loop Tracing

```
int i = 0;
while (i < 5) {
    System.out.println("*");
    ++i;
}
```



i  
~~0~~12345

output  
\*\*\*\*\*

**\*\* i is NOT less than 5, stop executing the loop \*\***

# Exercises



# How Many Times Does This Loop Run?

**Time Limit:** 5 minutes

```
int i = 1;
while (i <= 6) {
    System.out.println("*");
    ++i;
}
```

# While Loop Exercise 1

**Time Limit:** 5 minutes

Write a method that prints a NASA like countdown to a rocket ship launching. Make the starting number for the countdown a parameter to the method.

*Example output:*

3...

2...

1...

Lift off!

# While Loop Exercise 2

**Time Limit:** 7 minutes

Write a method that, given a base number and an answer (as whole numbers), returns the power that the base was raised to to get the answer.

Example: given 2 and 8 you would return 3 since  $2^3 = 8$

**Hint:** divide by the base until you get to 1

# While Loop Exercise 3

**Time Limit:** 5 minutes

Write a method that given a whole number divides the number by 2 and prints “even” as long as the number is even AND is not 0.

Examples:

- 6 would print “even” once since 6 is even but 3 is not
- 9 would never print “even” since it is odd
- 16 would print “even” 4 times since 16, 8, 4, and 2 are even

# While Loop Exercise 4

**Time Limit:** 5 minutes

Write a method to print out the following shape:

```
+-----+  
 \       /  
 /       \  
 \       /  
 /       \  
+-----+
```

**\*\* the height of this  
figure is 2 \*\***

Make the height of the figure parameters to your method

# Practice for Later

Write a method that takes in a base 10 whole number and converts it to binary, the method should return the binary number as a String.

**Reminder:** To convert a number to binary divide original number by 2 repeatedly, along the way keep track of the remainder of the division and prepend it to the binary number you build up along the way.

Continue to divide by 2 until the answer is 0.

The End