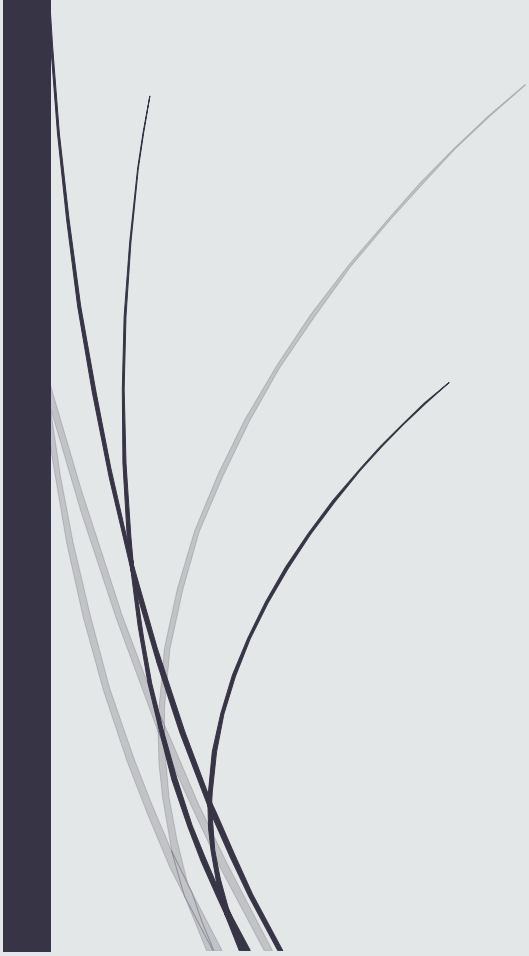
Java Lesson 5

Java From Scratch

Java Math

Java Booleans



Content

Java Math

The Java Math class has many methods that allows you to perform mathematical tasks on numbers.

Math.max(x,y)

The Math.max(*x*,*y*) method can be used to find the highest value of *x* and *y*:

Example

```
Math.max(5, 10);
```

Math.min(x,y)

The Math.min(*x*,*y*) method can be used to find the lowest value of *x* and *y*:

Example

```
Math.min(5, 10);
```

Math.sqrt(x)

The Math.sqrt(*x*) method returns the square root of *x*:

Example

```
Math.sqrt(64);
```

Math.abs(x)

The Math.abs(*x*) method returns the absolute (positive) value of *x*:

Example

```
Math.abs(-4.7);
```

Random Numbers

Math.random() returns a random number between 0.0 (inclusive), and 1.0 (exclusive):

Example

```
Math.random();
```

To get more control over the random number, for example, if you only want a random number between 0 and 100, you can use the following formula:

Example

```
int randomNum = (int)(Math.random() * 101);  // 0 to 100
```

Exercise:

Use the correct method to find the **highest value** of x and y.

```
int x = 5;
int y = 10;
Math.____(x, y);
```

Java Booleans

Very often, in programming, you will need a data type that can only have one of two values, like:

- YES / NO
- ON / OFF
- TRUE / FALSE

For this, Java has a boolean data type, which can store true or false values.

Boolean Values

A boolean type is declared with the boolean keyword and can only take the values true or false:

Example

```
boolean isJavaFun = true;
boolean isFishTasty = false;
System.out.println(isJavaFun);     // Outputs true
System.out.println(isFishTasty);   // Outputs false
```

However, it is more common to return boolean values from boolean expressions, for conditional testing (see below).

Boolean Expression

A Boolean expression returns a boolean value: true or false.

This is useful to build logic, and find answers.

For example, you can use a comparison operator, such as the **greater than** (>) operator, to find out if an expression (or a variable) is true or false:

Example

```
int x = 10;
int y = 9;
System.out.println(x > y); // returns true, because 10 is higher than 9
```

Or even easier:

Example

```
System.out.println(10 > 9); // returns true, because 10 is higher than 9
```

In the examples below, we use the **equal to** (==) operator to evaluate an expression:

Example

```
int x = 10;

System.out.println(x == 10); // returns true, because the value of x is equal to 10
```

Example

```
System.out.println(10 == 15); // returns false, because 10 is not equal to 15
```

Real Life Example

Let's think of a "real life example" where we need to find out if a person is old enough to vote.

In the example below, we use the >= comparison operator to find out if the age (25) is **greater than** OR **equal to** the voting age limit, which is set to 18:

Example

```
int myAge = 25;
int votingAge = 18;
System.out.println(myAge >= votingAge);
```

Cool, right? An even better approach (since we are on a roll now), would be to wrap the code above in an if...else statement, so we can perform different actions depending on the result:

Example

Output "Old enough to vote!" if myAge is **greater than or equal to** 18. Otherwise output "Not old enough to vote.":

```
int myAge = 25;
int votingAge = 18;

if (myAge >= votingAge) {
  System.out.println("Old enough to vote!");
} else {
  System.out.println("Not old enough to vote.");
}
```

Booleans are the basis for all Java comparisons and conditions.

Exercise:

Fill in the missing parts to print the values true and false:

```
____ isJavaFun = true;
____ isFishTasty = false;
System.out.println(isJavaFun);
System.out.println(isFishTasty);
```

Our Java Lessons

Java From Scratch

- Lesson 1 PDF (Java Getting Started)
- Lesson 2 PDF (Java Output, Comments, and Variables)
- Lesson 3 PDF (Java Data Types and Casting)
- Lesson 4 PDF (Java Operators and Strings)
- Lesson 5 PDF (Java Math and Booleans)
- Lesson 6 PDF (JAVA IF ELSE AND SWITCH)
- Lesson 7 PDF (Java While Loop and For Loop)
- Lesson 8 PDF (Java Arrays)