

Parameters, Return Values, & Math Methods

By: Megan Avery
Updated August 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
- Have AgeInformation.java and Receipt.java downloaded somewhere on your computer
 - If have completed other tutorials and practice problems will have them already
- *Suggested:* Watch previous Java tutorials

Parameters

Vocab Alert!

caller: the method calling a method

parameter: a value passed to a method by its caller

Purpose of Parameters

Parameters, like methods, are used to reduce redundancy in code. They are a way to customize what is being done inside a method so you don't have to write a new method for every version of the code.

Creating Method with a Parameter


can be any type

```
public static void methodName(type parameterName) {  
    // do stuff with the parameter  
    // example print it out  
    System.out.println("The parameter was: " + parameterName);  
}
```

have to match exactly

Calling a Method with a Parameter

```
public static void main(String[] args) {  
    // calling with with variable for parameter  
    type x = value;  
    methodName(x);  
    // calling with explicit value  
    methodName(value);  
}
```

 doesn't have to match parameter name

Multiple Parameters

Methods can have multiple parameters of different (or the same) types, just put them in separated by commas.

Example:

```
public static void methodName(type1 p1, type2 p2, type3 p3) {  
    // do stuff with parameters  
}
```


Scope

scope: the context within which a variable can be accessed

Variables can only be accessed within the pair of braces that they are defined in. So, variables defined in the main can only be accessed within methods if they are passed as parameters. Any variables defined within methods can only be used within those methods.

Common Errors

Calling a method that needs parameters without them

Calling a method with the wrong parameter type

Primitive Types as Parameters

value semantics: When primitive variables are passed as parameters, their values are copied.

- Modifying the parameter will not affect the variable passed in.

Method Overloading

method overloading: when 2 (or more) methods have the same name but different parameter lists.

- Either a different number of parameters or parameters of different types

Example:

```
public static void methodName(type parameterName) {  
    // do stuff with the parameter  
}  
  
public static void methodName(type1 p1, type2 p2, type3 p3) {  
    // do stuff with parameters  
}
```

Parameter Exercise

Edit the Age Information code from the Types, Variables, and Expressions tutorial to include a method that print following information with age as a parameter:

- all age information (calls the other methods)
- number of decades
- years into decade
- way to 100
- way to x method, where x is some given age

Return Values

Vocab Alert!

return: to send out a value as the result of a method

return type: the type that is being returned from a method

return value: the value that got returned from a method

Purpose of Return Values

Return values are used to give information back to the caller so that methods can be used for things like calculations instead of just printing.

Creating Method with Return Value

```
// return calculated variable
public static type methodName1() {
    type x;
    // code for calculation and assignment of x
    return x;
}

// return explicit value
public static type methodName2() {
    return value;
}
```

Calling a Method with a Return Value

```
public static void main(String[] args) {  
    type x = methodName1();  
    type y = methodName2();  
}
```

A Common Mistake

You have to store the return value from the method in a variable or it won't be able to be used later

Or, if you just want to print the return value you can use the method directly in a `println`

Can I Overload Methods with Return Values?

No, method overloading only works with parameters. If you try to create different methods with the same name, the same parameters, but different return values you will get a compile error

Return Value Exercise

Edit the Receipt code from the Types, Variables, and Expressions tutorial to include methods for:

- Printing out the receipt based on subtotal (will call other methods)
- Calculating the total with tax
- Calculating the total with tip

Math Methods

The Math Class

the Math class is a **static** class with methods in it that do many calculations for you.

static classes are used like `ClassName.methodName(params)`

Math Methods 1

Method Definition	Description
<code>Math.abs(value)</code>	absolute value
<code>Math.ceil(value)</code>	moves up to ceiling
<code>Math.floor(value)</code>	moves down to floor
<code>Math.log10(value)</code>	logarithm, base 10
<code>Math.max(value1, value2)</code>	larger of two values
<code>Math.min(value1, value2)</code>	smaller of two values
<code>Math.pow(base, exp)</code>	base to the exp power

Math Methods 2

Method Definition	Description
<code>Math.random()</code>	random double between 0 and 1
<code>Math.round(value)</code>	nearest whole number
<code>Math.sqrt(value)</code>	square root
<code>Math.sin(value)</code> <code>Math.cos(value)</code> <code>Math.tan(value)</code>	sine/cosine/tangent of an angle in radians
<code>Math.toDegrees(value)</code> <code>Math.toRadians(value)</code>	convert degrees to radians and back

Constants

Some classes have constants associated with them. Constants are variables that have set values and can be used to prevent “magic numbers” in your code.

Math class constants:

Constant	Value
Math.E	2.7182818...
Math.PI	3.1415926...

Practice for Later: Calculation Methods

- Calculates the average of 2 numbers
- Calculates the average of 3 numbers using method overloading
- Calculates the perimeter of a rectangle
- Calculates the area of a rectangle
- Calculates the circumference of a circle
- Calculates the area of a circle

When using the methods be sure to print out their values with the appropriate context

The End