Chapter 5 - Methods

Chapter 5 introduces the idea of methods, which allows programmers to write programs that are very complex in a way that is easier to understand, and in a way that usually requires fewer lines of code to be written. Methods are similar to the 'divide and conquer' approach, where to solve a complicated problem; the problem is broken down into smaller pieces and solved one step at a time. This allows for a shorter and more concise program which is usually preferred over a long and complicated code in the main method.

A method has the following structure:

```
private static int countA(int num) {
    Statements;
    return a;
} // countA method
```

In order to create a method, the knowledge of specific naming conventions for the method header is required. The following are four critical parts in writing a method header.

Name	Example	Purpose
Method Modifiers	public static	*public* sets if the method is publicly accessible to code outside of the class. *static* means the method belongs to the class and not a specific object
Return Type	int	The return value of the method when called. Can be many different data types. *void* returns nothing.
Method Name	main	A distinct name to call the method.
Parentheses	(int num)	Where arguments for the methods will be placed in. Arguments need to be the same as the data type specified in the parentheses.

In order to call a method in Java, the method name should be in the main method, and the arguments for the method should be placed inside the parentheses. ProgramOneCFive.java shows an example of calling a method in Java.

In order to pass arguments in the method, the argument needs to match the data type that has been specified in the parentheses. These arguments can then be accessed by the method and manipulated in calculations depending on the instructions. However, the value of the original argument will not be affected by the method. A method is not only limited to having one argument, but it is also capable of having multiple arguments in the paratheses. ProgramTwoCFive.java shows an example of a multi-argument method.

The return statement in a value-returning method is essential when writing methods. If the return type is *void*, then the method does not require a return statement. However, if the return type is a data type such as *int*, *boolean*, *double*, the method should return a value with the same data type as specified in the method header. ProgramThreeCFive.java highlights the difference between a *void* method and a value-returning method.