

# Chapter 1: Numbers, Strings and Concatenation

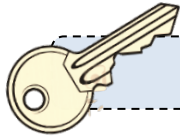
## Learning Outcomes:

- ✓ Identify the Java code structure
- ✓ Identify the importance and purpose of the 'main' method
- ✓ Implement strings and integers in a program
- ✓ Implement concatenation in a program



## Prerequisite Knowledge:

- ✓ Be able to save, compile and run a Java program in the chosen programming environment
- ✓ Be familiar with the maths operators used in Java to perform calculations (+ - \* /)



Concatenation

Class

Method

BODMAS

Keywords

## 1.1 The Theory: The Java Code Structure

Programming is a complex subject to learn, but there are simple theories that can be learnt in advance to better the programming experience. One such theory worth grasping beforehand, or at least having a quick overview of, is the structure of programming code.

### What is in a Java Source File?

A source file is a physical file that holds the source code for a program, or more specifically, holds a 'class'. Classes are amazing things, and play an integral part in object-orientated programming (OOP); however, before discussing OOP, 'first things first'. For now, view a class as a small piece of a larger application (although tiny applications may consist of a single class). For example, an email application may have a 'contacts' class and an 'inbox' class. Breaking larger applications down into smaller parts (classes) has numerous advantages, especially manageability. Source files in Java have the '.java' extension. The curly brackets mark the start and the end of a class.

```
public class OptimusPrime {  
    }  
    _____ Class name
```

### What is in a Java Class?

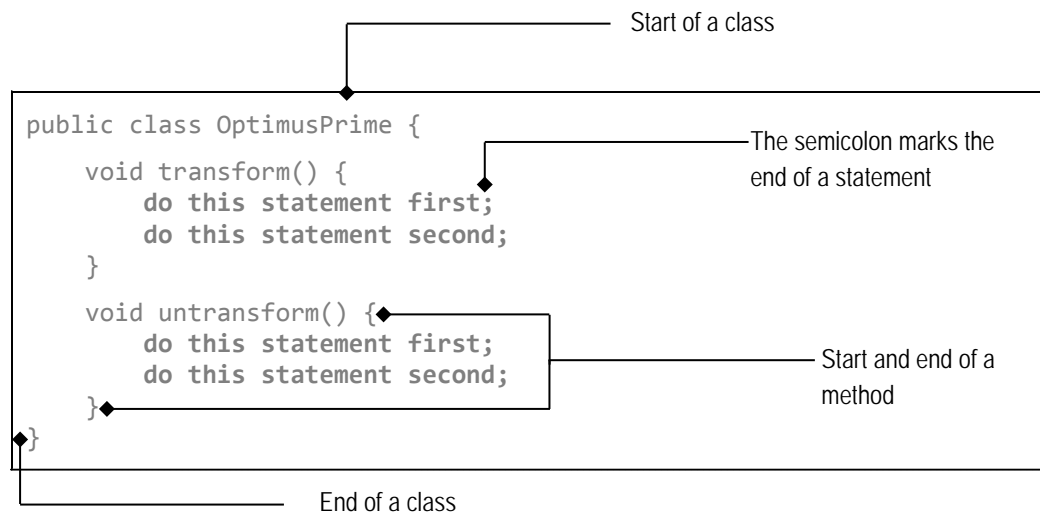
A class holds one or more methods; a method may be viewed as a segment of code which has a specific purpose. A good analogy may be to consider a class similar to a cookbook; a cookbook holds many recipes, the same as a class may hold many methods. Methods must be declared inside of a class, between the two curly brackets that state the start and end of the class. Methods also have curly brackets that mark the beginning and the end of themselves too; these are used to contain the method code.



```
public class OptimusPrime {  
    void transform() {  
    }  
    void untransform() {  
    }  
}  
    _____ Method name
```

## What is in a Java Method?

Recipes contain instructions of how to complete something. Methods hold instructions of how to complete something too, however, these instructions are referred to as statements. Statements are written between the curly brackets that mark the start and the end of a method. In Java, all statements must end with a semicolon (;), otherwise the compiler will throw an error. Programmers sometimes refer to methods as procedures, functions and/or subroutines too; don't get confused as they all refer to the same principle.



## The Java 'main' Method

Every Java program must have a 'main' method. The main method is the first method called by any program; in other words, the starting point of any application. The main method syntax is shown below:

```
public static void main(String[] args){  
}
```

Statements found between the curly brackets of a main method will begin to execute on the start-up of a program, every time. Make note that every Java application must have at least one class and must contain one main method; this is not one main method per class, but one main method per program! The main method will be examined more closely in later chapters.



### Programming Tips!

Indentation is useful to show precedence between classes, methods and statements. Not only is indentation good practice, but it is also extremely helpful when debugging lengthy code. In the example above, the methods are further indented than the class, and the statements are indented further still. Use the tab key, on the keyboard, to achieve this.



### The Big Problem

Java is case sensitive, so 'A' is considered different to 'a'. When typing out programs, be extra vigilant and check for upper-case and lower-case letters. For beginners, this is one of the most likely causes of errors in their code.

## 1.2 Practical: Printing Strings and Sequence

### My First Program

The `[System.out.println()]` method is used to print to a standard output, usually to the command-line. The content between the brackets `[("Hello Gizmo")]` is the content that is outputted to the command-line. The quotations are used as the value is 'string' (text), but it also indicates that there is no calculation to be performed. Removing the quotations from around text will cause an error.

**Important:** the class name and the source document name must be the same.

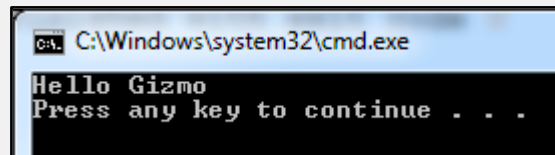
Try the code

```
public class MyFirstProgram{  
    public static void main(String[] args){  
        System.out.println("Hello Gizmo");  
    }  
}
```



### Step by step

1. Type out the above code
2. Save the program as `MyFirstProgram.java`
3. Compile the program
4. Run the `MyFirstProgram.class` file
5. Check the results of the program



```
C:\Windows\system32\cmd.exe  
Hello Gizmo  
Press any key to continue . . .
```

### Sequence

The Sequence of each statement is important; sometimes beginners forget this as their confidence grows and their applications become more complex. However, being vigilant of this will help to avoid frustrating errors. Remember, statement one will always be executed before statement two and so on.

### Activity 1.1

Based on the previous example, write an application that displays your top 5 favourite films!



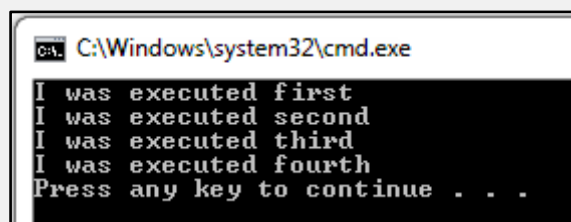
Try the code

```
public class MySecondProgram{  
    public static void main(String[] args){  
        System.out.println("I was executed first");  
        System.out.println("I was executed second");  
        System.out.println("I was executed third");  
        System.out.println("I was executed fourth");  
    }  
}
```



### Step by step

1. Type out the above code
2. Save the program as `MySecondProgram.java`
3. Compile the program (use the `javac` compiler)
4. Run the `MySecondProgram.class` file
5. Check the results of the program



```
C:\Windows\system32\cmd.exe  
I was executed first  
I was executed second  
I was executed third  
I was executed fourth  
Press any key to continue . . .
```

## 1.3 Practical: Numbers and Calculations

### Lucky Sevens

The `[System.out.println()]` method was used earlier to print string (text) values to the command-line; notice that the quotations were placed around the text. However, if a programmer wants a statement to perform a calculation with numbers, then the quotations are removed. For example, `7+7` will have a different output to `"7+7"`; numbers and/or text inside of quotations will be printed literally, as opposed to being calculated.

#### Activity 1.2

Finish the Lucky Sevens program, by adding divide (/) and subtract (-) statements, both with and without quotations.



Try the code

```
public class LuckySevens{  
    public static void main(String[] args){  
        System.out.println("7+7");  
        System.out.println(7+7);  
        System.out.println("7*7");  
        System.out.println(7*7);  
    }  
}
```



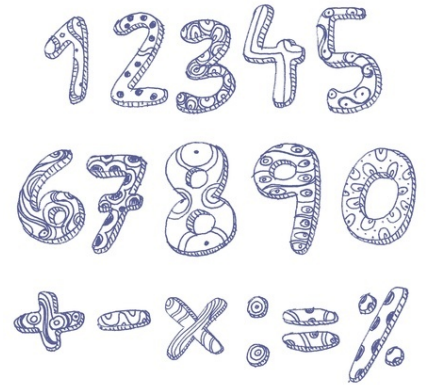
#### Step by step

1. Type out the above code
2. Save the program as `LuckySevens.java`
3. Compile the program
4. Run the `LuckySevens.class` file
5. Check the results of the program

```
C:\Windows\system32\cmd.exe  
7+7  
14  
7*7  
49  
Press any key to continue . . .
```

## Order of Operation

Calculations are not performed from left to right; they are performed in order of **BODMAS** (Brackets, Order, Division, Multiplication, Addition and Subtraction). For example, number operations in brackets will be calculated before divisions, and multiplications will be calculated before additions. In other words,  $7+4*2$  will not return the answer 22; it will return the answer 15. This is because multiplications are calculated before additions. If the programmer requires the answer 22, then this calculation  $(7+4)*2$  would be used instead.



Logical errors (errors that are logically correct, but unknowingly different to the programmers' intentions) are often hard to spot and in the past have caused many problems in software projects. For beginners, however, most logical problems are caused because they have simply forgotten about the order of operation (the BODMAS rules).

```
public class MyOperation{  
    public static void main(String[] args){  
        System.out.println(2+3*2);  
        System.out.println((2+3)*2);  
        System.out.println(1+2*3+7);  
        System.out.println((1+2)*(3+7));  
    }  
}
```

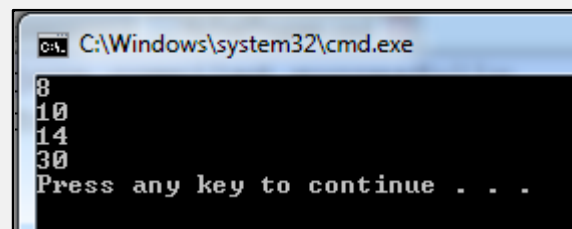


Try the code



### Step by step

1. Type out the above code
2. Save the program as MyOperation.java
3. Compile the program
4. Run the MyOperation.class file
5. Check the results of the program



### Activity 1.3



Experiment with different calculations in Java; write 10 medium to complex calculations on paper and the answers that you suspect will be printed to the command-line. Check the written-down calculations, and their answers, in a program; if they are incorrect examine them more closely to discover the reasons to why they were different.

## 1.4 Practical: Concatenation

### Joining It Up

Concatenation is a term used when joining parts together. In previous examples, 'parts' such as string (text) and numbers were printed to the command-line on individual lines. However, if a programmer wanted, for example, to print a sum and its answer on the same line, then concatenation is one solution that can be used. In Java, the addition symbol (+) is used to concatenate parts together.

#### Activity 1.4

Write a program that displays the first 10 sums of the 9 times table. The program should display the sum and the answer on the same line.

Try the code

```
public class MyJoins{  
    public static void main(String[] args){  
        System.out.println("7+7=" + (7+7));  
        System.out.println("7*7=" + 7*7);  
    }  
}
```



### Step by step

1. Type out the above code
2. Save the program as MyJoins.java
3. Compile the program
4. Run the MyJoins.class file
5. Check the results of the program

```
C:\Windows\system32\cmd.exe  
7+7=14  
7*7=49  
Press any key to continue . . .
```



### Programming Tips!

Whitespace (the space between code) does not have any impact on Java; for example, `7+7` and `7 + 7` would result in the same output. However, the space character does appear if it is string and inside of quotations; for example, `"7+7="` would appear differently in the command-line to `"7 + 7 = "`.



### Programming Tips!

The plus (+) symbol is used in Java for both addition operations and concatenation. This can sometimes cause a problem, for example, if a program concatenates two numbers as opposed to adding them together. However, this can be easily resolved by placing brackets around an addition operation to force the program to add, instead of concatenating. For example;

```
System.out.println("7+7=" + (7+7));
```