Practical 4: Expressions

Expressions are made of *operands* and *operators*. Operands can be combinations of variables, *literals* (numbers, characters or strings of characters for example) or *method calls*. Operators (as their name implies) operate on operands. These are symbols such as: +, −, /, * etc. The function of some operators can change depending on their operands. For example, you have already seen that the + operator does two completely different things depending on whether its operands are numbers or strings.

Expressions evaluate to a single value.

Task 1

The result of numeric expressions (expressions involving numeric variables and numbers) has a data type just like variables and literals in Java.

In this task you are supposed to write your own simple programs to find out the resulting type of numeric expressions that involve operands (variables and/or numbers) of various types. Complete the following table with your findings:

Numeric expressions containing data type(s)	Evaluate to data type
`byte`	
`short`	
`int`	
`byte, short`	
`float`	
`int, float`	
`float, double`	

For example, to find out the data type of the return value of an expression involving only operands of type `byte` (first row of the table above) you could start considering the following simple program:

```
public class Application
{
    public static void main(String[] args)
    {
        byte a = 5;
        byte b = 8;

        byte c = a + b;
    }
}
```

...which will not compile because the result of the expression `a + b` above is not of type `byte`. Have a look at the compiler error message to find out what you should change the data type of variable `c` to.

Do you understand what you are doing and why?

Ask a demonstrator to look at your completed table before you proceed to the next task.

Task 2

Remember that unless you specify otherwise (with the L and F suffixes) the whole numbers that you use as literals in your program code are of type `int` and the real numbers are of type `double`.

Consider the program below. **Try to predict what the output will be before running the above code. Can you explain why you do not get what you expect?**

```
public class Program
{
    public static void main(String[] args)
    {
        int a = 2;
        int b = 4;
        float c = 2.0F;
        float d = 4.0F;

        System.out.println("a/b = " + (a/b));
        System.out.println("c/d = " + (c/d));
        System.out.println("a/d = " + (a/d));
        System.out.println("c/b = " + (c/b));
    }
}
```

Task 3

A great thing about the Java computer language is that there are many *libraries* of classes you can use that have been either created by the Java developers themselves or by the very large community that uses the language.

One of the standard and most often used classes is the `Math` class. It used to be the case that you had to *import* the `Math` class at the start of your program (before the class definition) if you wanted to use it with the following statement:

```
import java.lang.Math;
```

However, in the more recent versions of Java the Math class is part of the default `java.lang` library and it does not need to be imported explicitly.

The following program is an example using the `Math` class. The program makes use of the `abs` method that is defined in the Math class. The `abs` method is used to find the *absolute value* of a number i.e. the number without its sign.

```
import java.lang.Math;  // Not necessary but your program will still work.

public class Program
{
    public static void main(String[] args)
    {
        int a = -2;
        float b = 4.2F;

        System.out.println("The absolute value of a is " + Math.abs(a));
        System.out.println("The absolute value of b is " + Math.abs(b));
    }
}
```

Notice in the example above that the expression:

```
"The absolute value of a is " + Math.abs(a)
```

is one involving a String operand (`"The absolute value of a is "`), the `+` operator (which acts as a *concatenator* above) and a method call (`Math.abs(a)`). The above expression evaluates to a single value (the String `"The absolute value of a is 2"`) which is passed as an *argument* to the `println` *method*.

Ask a demonstrator to explain if you do not fully understand the program.

Task 4

The following website:

https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html

is the official class definition of the Java `Math` class. Please study it carefully.

Write small programs to demonstrate the function of the following `Math` class methods:

`floor`
`max`
`min`
`random`
`round`
`sqrt`