Review Sheet – Operators

Summarising the lessons from the Operators Tutorial:

What Are Operators

Arithmetic Operators allow us to do maths on variables:

Operator	Name	Description
+	Addition Operator	Adds two values together
-	Subtraction Operator	Subtracts one value from another
*****	Multiplication Operator	Multiplies two values together
/	Division Operator	Divides one value by another value
%	Modulo Operator	Gives the remainder after dividing two whole numbers, or integers
++	Increment Operator	Increases a whole number, or integer, by 1 value
--	Decrement Operator	Decreases a whole number, or integer, by 1 value

Assignment Operators give new values to variables, and can be combined with **Arithmetic Operators:**

Operator	Name	Example
=	Assignment Operator	**C = A + B** gives the value of **A + B** to **C**
+=	Addition And Assignment	**C += A** is the same as **C = C + A**
-=	Subtract And Assignment	C -= A is the same as C = C - A
*=	Multiply And Assignment	C *= A is the same as C = C * A
/=	Divide And Assignment	C /= A is the same as C = C / A
%=	Modulus and Assignment	C %= A is the same as C = C % A

Using Operators

```
int myVariable = 5;
myVariable = 10 + 50;
```

We can use arithmetic variables to do simple equations to create new values for a variable. The result above would be 50.

Assignment Operator and Reading Variables

```
int myVariable = 5;
myVariable = myVariable + 5;
```

Using variables on the left side of the = assignment operator means you will change them to the value on the right. Using variables on the right side of = operator means go and check what this variable is set to, and use that value. In the above, myVariable would be become 10, because it is 5 + 5.

Shorthand Arithmetic

```
int myVariable = 5;
myVariable += 5;
```

The short hand way of keeping a variables value the same, then adding another value to it.

Operators on Strings

You can't do maths on strings, but you can add two string together to make a longer one.

Decimals

```
int myVariable = 5;
myVariable /= 3;
```

Integers can't have decimal values. In the above example, the decimals would be chopped off giving you a value of 1.

```
float myVariable = 5;
myVariable /= 3;
```

If your variable is a float, however, it can have decimals. In the above example, you ould get a value of 1.666667.

```
float myVariable = 5;
myVariable = 5 / 3;
```

If you use only integers in an equation, the answer will always be an integer, even if assigning the value to a float. The above would result in 1.

```
float myVariable = 5;
myVariable = 5.0f / 3;
```

To make sure you get a decimal number, ensure that one of the values in your equation is a float – either declared explicitly by adding .0f, or by using a **float** variable in the equation.

Modulo

```
int myVariable = 5;
myVariable = myVariable % 3;
```

Modulo first divides the two values, then tells you what the remainder is. In the above example, the answer would be 2.

Equation	Answer
4 % 2	0
3 % 2	1
5%2	2
5%1	0

Using % 2 on any number is a good way to work out if it is even or odd. If the answer is 1, it's odd. If the answer is 0, it's even.