Further Data Manipulations in C

The C language has a few features available that may initially seem peculiar, but provide some simple and efficient ways to perform mathematical operations. To see some of these unique operations, we will first investigate a fundamental difference in the way that all languages handle operations.

Example

One of the most basic types of operations is to be able to take an existing value for a variable and change it to a new value. For example, if we were to make a number guessing program, we might use an integer called **numGuesses** that keeps track of the number of guesses that the player has taken. Each time a new guess is made, 1 is added to the previous value of **numGuesses**. The way this operates is as follows:

```
numGuesses = numGuesses +1;          //Add one to # guesses made
```

This type of operation would be illegal in your math class, but is essential in programming. In math class, saying that x = x + 1 would be simplified by cancelling an x on each side, leaving the result that 0 = 1, which math teachers generally don't like. In a computer program, however, the equation takes the original value of the variable, adds one, and then saves this as a new value. Operations are ALWAYS done from right to left. The C language also provides a couple of options on how this may be written, which sometimes causes it to look even more peculiar to beginning programmers. The statements below are all equivalent in their result.

```
numGuesses = numGuesses + 1;         //The easiest to read
numGuesses += 1;                     //Simpler; works for any addition
numGuesses++;                        //The increment operator.
```

The last of these examples, the increment operator, is very commonly found in C programs. Although it can only add a value of one to a variable's value, we will soon see a number of ways that this can be used effectively in programming. (Note: the increment operator can also be done as ++numGuesses…. what difference might this make?)

Accompanying the increment operator is the decrement operator, a double negative sign. We could use a command such as numGuesses—if our program was counting down the number of allowable guesses remaining before the user's guesses ran out.

Exercises:

1. In the table below, determine an equivalent way to represent each operation.

Original Equation	Alternative Form
x = x + 5	x+=5
time = time –1	
y = y –5	
value = 1.08*value	
size = size/10	
remaining = remaining %5	

2. In each example, determine the value of each variable at each step.

int x = 25;
 	
x = x –10;
 	
x *=7;
 	
x -=5;
	int x = 10, y = 12;
	
	x++;
	
	y--;
	
	x*=y;
		int x = 5, y =10;
		
		x = y + 10;
		
		y = x + 5;
		
		x = y;

3. Write a C program called check.c which will determine the accuracy in question 2 by printing the values for each variable after each operation. You should print initial, intermediate and final values for each variable.

4. Write a C program called manips.c . In this program you are to use three variables called **num1**, **num2** and **num3**. Create several different operations on each of these variables, with a print statement outputting their values after each operation. Have a neighbour assign initial values for each of the variables in your program and have that person predict the output of the program. When the person has completed their prediction, run your program and compare answers. Repeat as necessary with different values.