C++ Week 1 Operator Overloading

Operator Overloading

Exercise 1

Create a new C++ console project and within it define the class below. The value within the size data member should be used in the comparison of two Entity objects. Likewise the increment and decrement operators should affect the value of size.

```
class Entity{
public:
        int size;
        Entity(int s):size(s){}
};
```

Now overload the following operators
++(prefix)
--(prefix)
>
<
Test them out within main by creating two static instances of Entity and outputting message indicating the value of size and results of the comparison, increment & decrement operators.

Vectors

Exercise 2

Define the Weapon class below.

```
class Weapon{
public:
        int id;
        Weapon(){}
        Weapon(int id):id(id){}
};
```

Within main create a vector to hold weapons and within a for loop add 10 Weapon objects to the vector using the push_back member function, passing each new weapon a unique id in the constructor from 0..9.
Now define an iterator and use it to iterate over the vector printing the id values of the Weapon objects.

C++ Week 1 Operator Overloading

Exercise 3

Delete from the vector the Weapon with an id of 5 and display the revised list of Weapons.
Hint: In order for the find function to work you will need to overload the Weapon's equality operator (==). Which should return true when the ids are equal. The declaration for the operator overload is

```
bool operator==(const Weapon& rhd);
```

Recursion

A recursive function is one that invokes itself. These functions are often used to navigate through data structures like trees.

Exercise 4

Define a new class named IOObject. Within it define the single public member function below.

```
void DisplayIntLessThan(int n) {
        --n;
        if (n > 0) {
            DisplayIntLessThan(n);
        }
        cout << n << endl;
}
```

From within the main function instantiate an instance of IO and invoke DisplayIntLessThan passing 20 as an argument. Step through the code to predict the output before running the project.

C++ Week 1 Operator Overloading

Exercise 4

The IOObject member function below uses a loop to returns the factorial of a whole number passed as an argument. The factorial is the number calculated by multiplying all the numbers before and including the argument, together.
e.g. Factorial of 4 is 1 * 2 * 3 * 4 = 24

```
int factorial(int n) {
        int t, result;
        result = 1;
        for (t = 1; t <= n; t++) result *= t;
                return result;
}
```

Add it to your IOObject class and invoke it from main with the argument 4, output the value returned.

Exercise 5

Now write a new member function named factorialR that uses **recursion** to calculate the factorial. The algorithm for this method is below, where number is the argument passed to factorialR

If number equal to 1 return 1
return factorialR(number -1) * number