Equivalence of C++ Looping Structures

This lesson explains how we can convert a for loop into a while loop.

We'll cover the following

- Example of For loop
- Converting For Loop to While Loop

Example of For loop

An example of `for` loop is given below:

```
#include <iostream>
using namespace std;
int main()
{
    int i;
    for (i=0 ; i<10 ; i++) {
        i = i*2;
        cout << "Value of i is: "<<i<<endl;

    }
    cout << "Final value of i is: " << i << endl;
    return 0;
}
```

Converting For Loop to While Loop

This can easily be reformatted as (do recognize the extra enclosing brackets, and the two extra semicolons after the expressions in order to turn them into statements):

```
#include <iostream>
using namespace std;
int main()
{
    int i=0 ;
    while (i<10) {
        i = i*2;
        cout << "Value of i is: "<<i<<endl;
        i++;
    }
    cout << "Final value of i is: "<< i <<endl;

    return 0;
}
```

A `for` loop is more often used by C++ programmers due to its conciseness as well as its separation of the looping logic (often using a loop control variable like **“int i”** or another simple iterator) from the loop’s content.

A `while` loop is often preferred if the *initial* statement or update statement requires more complex code than fits neatly into the `for` construct. However, the two are fully *equivalent*. Therefore, it is ultimately a coding style decision, not a technical decision whether to use one or the other.

In the next lesson, we’ll discuss infinite loops and how they arise.