Lesson 14 - Loops and Logic II

Recall loops,

> ## Loop
>
> A `loop` in programming refers to a code block that executes some lines of code repeatedly, so think of 5 lines labelled `2`, `3`, `4`, `5`, `6`. Now we want to execute these 5 lines, 3 times each. so this is how the code will be executed:
>
> ```
> 1, 2, 3, 4, 5, 6, 2, 3, 4, 5, 6, 2, 3, 4, 5, 6, 7, 8, 9 ... and so on.
> ```
>
> So line 1, then 2-6, again 2-6, and one last time 2-6 then we move on to 7, 8 and more. Each repetition is called an `iteration`, so one iteration in this case would be `2`, `3`, `4`, `5`, `6`.
>
> Now lets take a look at the types of loops and how to implement them.

For loop

When we want to execute some code a fixed number of times, then we use a `for` loop. The general syntax of the `for` loop is as follows,

```
for (initialize; condition; increment)
{
        // coode to be executed repeatedly
}
```

There are 3 parts of a `for` loop.

1. Initialize the loop variable `i`
2. Set the looping condition
3. Set the increment or decrement condition

Example: basic for loop

```
for (int i = 0; i < 10; i++)
{
  Console.WriteLine(i);
}
```

Example: loop over an array

```
int[] myArray = { 1, 2, 3, 4 };
int sum = 0;

for (int i = 0; i < myArray.Length; i++)
        sum += myArray[i];

Console.WriteLine($"Sum: {sum}");
```

Example: for loop with decrement

```
for (int i = 10; i > 0; i--)
        Console.WriteLine(i);
```

Example: for loop with compound arithmetic operations

```
for (int i = 0; i < 20; i += 2)
        Console.WriteLine(i);
```

Example: You can skip init and increment

```
int i = 0;

for ( ; i < 10 ; )
        Console.WriteLine(i++);
```

For each loop

There is a special loop for iterating over a collection, called a `foreach` loop. The general syntax is as below.

```
foreach (var item in collection)
{
        // code to be executed for each item
}
```

Example

```
int[] myArray = { 1, 2, 3, 4 };

foreach (int num in myArray)
        Console.WriteLine(num);
```