# Chapter 2 Coding Expressions and Statements

For efficiency, many of the examples in the rest of the book are snippets, but you can still add these statements inside of a **Main** method to compile and get a better feel for C# syntax. There will be plenty of complete programs too.

## Looping Statements

C# supports several loops, including **for**, **foreach**, **while**, and **do**. The code listings that follow perform similar logic.

double[] temperatures = { 72.3, 73.8, 75.1, 74.9 };   
for (int i = 0; i < temperatures.Length; i++)   
{

Console.WriteLine(i);

}

*Code Listing 30*

The **for** loop initializes **i** to **0**, makes sure **i** is less than the number of items in the **temperature** array, executes the **Console.WriteLine**, and then increments **i**. It continues executing until the condition (**i < temperatures.Length**) is false, and then moves on to the next statement in the program.

|  |
| --- |
| foreach (int temperature in temperatures)  {  Console.WriteLine(temperature);  } |

*Code Listing 31*

The **foreach** loop used in Code Listing 31 is simpler and will execute for each value in the **temperatures** array.

Next is an example of a **while** loop.

int tempCount = 0;

while (tempCount < temperatures.Length)

{

Console.WriteLine(tempCount); tempCount++;

}

*Code Listing 32*

The **while** loop evaluates the condition and executes if it’s true. Notice that I initialized **tempCount** to **0** and increment **tempCount** inside of the loop on each iteration.

Finally, the following example shows how to write a **do**-**while** loop.

|  |
| --- |
| int tempCount2 = 0; do  {  Console.WriteLine(tempCount2++);  }  while (tempCount2 <= temperatures.Length); |

*Code Listing 33*

A **do**-**while** loop is good for when you want to execute logic at least one time. This example increments **tempCount2** as a parameter to **Console.WriteLine**. Remember, the postfix operator changes the variable after evaluation.

## Summary

C# has a full set of operators and types that allow you to write a wide range of expressions and statements. With branching statements and loops, you can write logic of your choosing. All of the code in this chapter has been in the **Main** method, but clearly that’s inadequate and you’ll quickly grow out of that. The next chapter explores some new C# features to help organize code with methods and properties.