namespace Usingiterators {

class Program {

static void Main()

{

// Using a simple iterator.

Listclass listclass1 = new Listclass();

foreach (int i in listclass1) {

System.Console.Write(i + " ");

}

// Output: 0 1 2 3 4 5 6 7 8 9

System.Console.WriteLine();

// Using a named iterator.

Listclass test = new Listclass();

foreach (int n in test.Sampleiterator(1, 10))

{ System.Console.Write(n + " "); }

// Output: 1 2 3 4 5 6 7 8 9 10

System.Console.WriteLine();

// Using multiple yield statements.

foreach (string element in new Testclass())

{ System.Console.Write(element); }

// Output: With an iterator, more than one value can be returned.

System.Console.WriteLine();

} }

class Listclass : System.Collections.Ienumerable

{

public System.Collections.Ienumerator Getenumerator()

{

for (int i = 0; i < 10; i++)

{ yield return i; }

}

// Implementing the enumerable pattern

public System.Collections.Ienumerable Sampleiterator(int start, int end)

{

for (int i = start; i <= end; i++)

{ yield return i; }

} }

class Testclass : System.Collections.Ienumerable

{

public System.Collections.Ienumerator Getenumerator()

{ yield return "With an iterator, ";

yield return "more than one ";

yield return "value can be returned";

yield return ".";

} } }