foreach Loops

<https://csci-1301.github.io/about#authors>

November 2, 2023 (01:03:32 PM)

Table of Contents

This lab serves multiple goals:

* To introduce you to foreach loops,
* To introduce you to their use cases,
* To compare for loops and foreach loops by converting between them, and
* To illustrate how foreach can be useful in conjunction with classes.

# Practicing foreach Loops

## Warm-up

Create a new project, and replace the content of the Main method with the following code:

int[] primes = {2, 3, 5, 7, 11, 13, 17, 19};
for(int i = 0; i < primes.Length; i++)
{
 Console.WriteLine(primes[i]);
}

Execute the code. You should see the elements of the array *primes* (the prime numbers less than 20) in the console.

Next rewrite the code using a **foreach** statement, then answer the following questions:

1. Identify two differences between the for and foreach versions.
2. Which one is easier to understand?
3. Which one needs fewer variables?

Answers:

The code simply becomes:

int[] primes = {2, 3, 5, 7, 11, 13, 17, 19};
foreach(int val in primes)
{
 Console.WriteLine(val);
}

* The differences are the keyword (obviously!), the fact that foreach does not need indices nor to use the Length property, and the absence of an update or condition in the header.
* This is a matter of taste, but foreach generally seems more intuitive.
* Both use one additional variable (i in the for case, val in the foreach case).

## Converting from for to foreach (1/2)

Can you rewrite the following code with a foreach statement? Why?

double[] numbers = {1.2, 4.3, 5.7, 11, -3.13, 1.7};

for(int i = 0; i < numbers.Length; i++)
{
 numbers[i] = numbers[i] \* 1.1;
 Console.WriteLine(numbers[i]);
}

## Converting from for to foreach (2/2)

Can you rewrite the following code with a foreach statement? Why?

double[] numbers = {1.2, 4.3, 5.7, 11, -3.13, 1.7};

for(int i = 0; i < numbers.Length - 1; i++)
{
 Console.WriteLine((numbers[i] + numbers[i+1]) / 2);
}

## Conversion between for and foreach

1. Can you think of any loops that can be implemented with *foreach* but not with *for*? If so, write an example.
2. Can you think of any loops that can be implemented with *for* but not with *foreach*? If so, write an example.

# Mixing foreach With Classes

Download the [Library project](labs/Foreach/Library.zip), extract it, and open it with your IDE.

Observe the program and its two classes:

* The Book class represents a single book.
* Program creates an array of 10 books.

Next modify the code in Program.cs to perform the following steps:

1. Write a foreach loop that displays all the books.
2. Add statements where you ask the user to enter a year, then modify the foreach loop to display only books published on or after the year the user entered.
3. Write a for loop implementation that performs the same task of displaying books published on or after the year user entered.

Which one do you prefer to implement the above search? Explain your answer.