100 XP

Exercise - Implement the foreach statement

12 minutes

Suppose you work for a manufacturing company. The company needs you to complete an inventory of your warehouse to determine the number of products that are ready to ship. In addition to the total number of finished products, you need to report the number of finished products stored in each individual bin in your warehouse, along with a running total. This running total will be used to create an audit trail so you can double-check your work and identify "shrinkage".

Looping through an array using foreach

The `foreach` statement provides a simple, clean way to iterate through the elements of an array. The `foreach` statement processes array elements in increasing index order, starting with index 0 and ending with index Length - 1. It uses a temporary variable to hold the value of the array element associated with the current iteration. Each iteration will run the code block that's located below the `foreach` declaration.

Here's a simple example:

c#

```
string[] names = { "Rowena", "Robin", "Bao" };
foreach (string name in names)
{
    Console.WriteLine(name);
}
```

Below the `foreach` keyword, the code block that contains the `Console.WriteLine(name);` will execute once for each element of the `names` array. As the .NET runtime loops through each element of the array, the value stored in the current element of the `names` array is assigned to the temporary variable `name` for easy access inside of the code block.

If you ran the code, you would see the following result.

Output

```
Rowena
Robin
Bao
```

Use the `foreach` statement to create a sum of all the items on hand in each bin of your warehouse.

Create and initialize an array of int

1. Ensure that you have an empty Program.cs file open in Visual Studio Code.

   If necessary, open Visual Studio Code, and then complete the following steps to prepare a Program.cs file in the Editor:

   a. On the **File** menu, select **Open Folder**.

   b. Use the Open Folder dialog to navigate to, and then open, the **CsharpProjects** folder.

   c. In the Visual Studio Code EXPLORER panel, select **Program.cs**.

   d. On the Visual Studio Code **Selection** menu, select **Select All**, and then press the Delete key.

2. To create an array of type `int` that stores the number of finished products in each bin, enter the following code:

   ```c#
   int[] inventory = { 200, 450, 700, 175, 250 };
   ```

Add a foreach statement to iterate through the array

1. To create a `foreach` statement that iterates through each element of the `inventory` array, enter the following code:

   ```c#
   foreach (int items in inventory)
   {
   ```

```
}
```

Notice that the `foreach` statement temporarily assigns the value of the current array element to an `int` variable named `items`.

2. Ensure that your code matches the following:

```c#
int[] inventory = { 200, 450, 700, 175, 250 };

foreach (int items in inventory)
{

}
```

Add a variable to sum the value of each element in the array

1. Position the cursor on the blank code line above the `foreach` statement.
2. To declare a new variable that represents the sum of all finished products in your warehouse, enter the following code:

```c#
int sum = 0;
```

Ensure that you declare the variable **outside** of the `foreach` statement.

3. Position the cursor inside the code block of the `foreach` statement.
4. To add the current value stored in `items` to the `sum` variable, enter the following code:

```c#
sum += items;
```

5. Ensure your code matches the following:

```c#
```

```
int[] inventory = { 200, 450, 700, 175, 250 };
int sum = 0;
foreach (int items in inventory)
{
    sum += items;
}
```

Display the final value of sum

1. Create a blank code line below the code block of the `foreach` statement.

2. To report the final sum of items in your inventory, enter the following code:

```
c#

Console.WriteLine($"We have {sum} items in inventory.");
```

3. Ensure that your code matches the following:

```
c#

int[] inventory = { 200, 450, 700, 175, 250 };
int sum = 0;
foreach (int items in inventory)
{
    sum += items;
}

Console.WriteLine($"We have {sum} items in inventory.");
```

4. On the Visual Studio Code **File** menu, click **Save**.

5. In the EXPLORER panel, to open a Terminal at your TestProject folder location, right-click **TestProject**, and then select **Open in Integrated Terminal**.

6. At the Terminal command prompt, type **dotnet run** and then press Enter.

```
Output

We have 1775 items in inventory.
```

Create a variable to hold the current bin number and display the running total

To fulfill the final requirement of your inventory reporting project, you'll need to create a variable that will hold the current iteration of the `foreach` statement so you can display the bin and the count of finished items in that bin, along with the running total of all items of bins accounted for so far.

1. Create a blank code line above the `foreach` statement.

2. To declare `int` variable named `bin` that's initialized to `0`, enter the following code:

   ```
   c#

   int bin = 0;
   ```

   You will use `bin` to store the number of the bin whose inventory is currently being processed.

3. Inside the `foreach` code block, to increment `bin` each time the code block is executed, enter the following code:

   ```
   c#

   bin++;
   ```

   Notice that you use the `++` operator to increment the value of the variable by 1. This is a shortcut for `bin = bin + 1`.

4. To report the bin number, the number of finished products in the bin, and the running total of finished products, enter the following code inside the `foreach` code block, after `bin++;`:

   ```
   c#

   Console.WriteLine($"Bin {bin} = {items} items (Running total: {sum})");
   ```

   This code will use your counter variable `bin`, the temporary `foreach` variable `items`, and your `sum` variable to report the current state of your inventory in a nicely formatted message.

5. Ensure that your code matches the following:

```c#
int[] inventory = { 200, 450, 700, 175, 250 };
int sum = 0;
int bin = 0;
foreach (int items in inventory)
{
    sum += items;
    bin++;
    Console.WriteLine($"Bin {bin} = {items} items (Running total: {sum})");
}
Console.WriteLine($"We have {sum} items in inventory.");
```

6. Save the changes to your Program.cs file, and then run the application.

   You should see the following output:

```Output
Bin 1 = 200 items (Running total: 200)
Bin 2 = 450 items (Running total: 650)
Bin 3 = 700 items (Running total: 1350)
Bin 4 = 175 items (Running total: 1525)
Bin 5 = 250 items (Running total: 1775)
We have 1775 items in inventory.
```

Recap

Here's a few things to remember about `foreach` statements and incrementing values that you learned in this unit:

- Use the `foreach` statement to iterate through each element in an array, executing the associated code block once for each element in the array.
- The `foreach` statement sets the value of the current element in the array to a temporary variable, which you can use in the body of the code block.
- Use the `++` increment operator to add 1 to the current value of a variable.

Next unit: Exercise - Complete a challenge activity for nested iteration and selection statements

Continue >