Scalable Data Infrastructures - Code Exercise 06

Overview

This code exercise will give you a chance to work on your ability to work with Lists within a loop.

Instructions

It's always best to start with a problem analysis. While one is not required to be submitted for this code exercise, it would benefit you greatly to begin by examining the requirements and creating a bulleted list of the things you'll need to complete the code. You can then begin by creating a new solution with the following naming format: *LastName_FirstName_CE06*.

The first thing you'll need to do is create two Lists. Keep in mind that in order to use Lists in your project, you'll need to add the directive *using System.Collections.Generic* to the top of your code. Each List can be anything you like: a List of car makes or models, a List of movie or book titles, etc. Be creative! The Lists do not have to be related in anyway, but you must have two Lists with at least five (5) elements, and each List must be a different length. So, if your first List is 5 elements, your second List would need to be more than 5 elements.

Next, create a custom method that will accept the List as an argument. The code within the custom method should have a loop that will cycle through the List and output each element of the List. Remember that your output should be meaningful to the user, and not simply outputting each element without some kind of context. This custom method will not return any values, so make sure this is indicated in the method signature.

In the Main method, create an output that tells the user what the application is going to do. Then invoke the custom method by sending it the first List from the Main method. Then, inform the user that another List will be output, and then invoke the custom method again by sending the second List. In the end, the Main method should contain two Lists of different lengths and two different method calls with each method call including a different List. Thus, the method will output the elements of the first List with the first method call and will output the second List with the second method call.

Things to Consider

Look to your problem analysis to verify that you've included all the requirements of the problem.
If you ask your instructor or a lab specialist for assistance on the code, he/she is going to ask to see your problem analysis first. If you didn't do that, we are going to tell you to complete that task before we will look at your code.

Make sure you're including comments at the top of the code to include your name, class and term, and the assignment name and that you have meaningful comments for each line of code in the project.

Finally, remember that you must compress the entire project folder for submission. Submitting only the Program.cs file or the .sln file will result in a 0 for the activity.

Rubric: Code Exercise 6

Minimum Project Requirements

These requirements must be satisfied before any points are awarded. Failing to meet these requirements will result in a zero (0) grade.

1. Project must run when instructor compiles it.
2. The submission must be submitted in the proper format as defined in the FSO activity.
3. You will lose 5 points if the project does not follow the naming convention described in the activity's documentation.

Topic	%	Excellent (100%)	Acceptable (80%)	Good (50%)	Fair (25%)	Poor (0%)
Coding						
Comments	**5**	Comments exist at the top of the code to include name, class and term, assignment, and date, and each line of the code is properly commented.	Missing the initial comments with name, class and term, assignment, and date, but the rest of the code is commented properly.	Missing up to four line comments, but some comments present.	Missing more than four comments	No comments in the code.
Syntax	**10**	There are no syntax errors, including correct line and **formatting according to the style taught**.	There are no syntax errors, but the code does not follow the style taught.	Project code contains minor syntax errors but is easily fixed.	Project code contains more major syntax errors but are easily fixed.	Project code does not run.
Lists	**25**	Two Lists are created with descriptive names, at least 5 elements, and a different number of elements in each.	Two Lists are created but both contain the same number of elements.		Only one List exists in the project.	**No Lists exist in the submitted project. Zero (0) for the entire exercise.**
Custom Method	**20**	A custom method is created and includes all the requirements of the method signature, including a descriptive, PascalCase, active-verb name with correct return and parameter data types.	A custom method is created, but the name is not descriptive or not an active verb or not in PascalCase.	A custom method is created but indicates an incorrect return type in the method signature.	A custom method is created but is missing a parameter to receive data in the method signature.	No custom method exists in the submitted project.
Method Call	**15**	The custom method is properly invoked and the data is sent as an argument.	The custom method is invoked, but the List is not sent as an argument.			The custom method is not invoked or does not exist.
Loop	**15**	A proper loop is included in the custom method and uses the length of the List as the interation limit.	The wrong loop type is used to loop through the List.	The correct loop is used, but the value in the comparison does not use the number of List elements to control the number of times the loop runs.	The correct loop is used, but the code is not encapsulated in the custom method.	No loop is used to cycle through a List.
Output	**10**	Output is meaningful and done in the Main method.	Output is done in the Main method, but the text is not meaningful to the user.			The output is not done in the Main method.