Lab 6 Interfaces, LINQ, and Extensions

CS321L Winter 2023, Professor Christopher Diggins

Implement the following LINQ methods as extension methods.

1. IEnumerable<T2> Select<T1,T2>(this IEnumerable<T1> self, Func<T1, T2> f);
2. IEnumerable<T> Where<T>(this IEnumerable<T> self, Func<T, bool> f);
3. TAcc Aggregate<TAcc, T>(this IEnumerable<T> self, TAcc acc, Func<TAcc, T, TAcc> f);
4. IEnumerable<T> Skip<T>(this IEnumerable<T> self, int count);
5. IEnumerable<T> Take<T>(this IEnumerable<T> self, int count);
6. bool Any<T>(this IEnumerable<T> self, Func<T, bool> f);
7. T First<T>(this IEnumerable<T> self, Func<T, bool> f);
8. T Last<T>(this IEnumerable<T> self, Func<T, bool> f);
9. Int Count<T>(this IEnumerable<T> self, Func<T, bool> f);
10. IEnumerable<T> Concat<T>(this IEnumerable<T> self, IEnumerable<T> other);

Write a unit test for each method. Use as an input to each test an array of integers from 1 to 10 inclusive (e.g., new int[] { 1,2,3,4,5,6,7,8,9,10 }) For the unit tests can convert the results of IEnumerable into arrays using your “ToArray()” implementation and use the “Assert.AreEqual” function

Submission

- Provide a screen shot of the tests in .PNG format
- Provide the code as a single “.cs” file.
- Nothing else please

Grade 15 points

- 1 point per working implementation + passing unit test.
- 5 points for following C# coding conventions - https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/coding-conventions

Notes:

- There was an error Concat() should have been Concat<T>(). It is now fixedx.
- The “default” keywords can be a useful way to initialize values that are unknown. For example “T value = default;”
- Don’t worry about warnings related to whether or not your value is nullable.
- You can also remove the “<Nullable>enable</Nullable>” line in the project file to remove warnings and errors around usage of nullable types.
- If “First<T>()” or “Last<T>()” does not find a result, you should throw an exception: “throw new Exception(“not found”)”
- If you choose to test that exception is thrown you could use “Assert.Throws<Exception>(() => youtestcode)”;