Операция SequenceEqual

Операция SequenceEqual определяет, эквивалентны ли две входные последовательности. Эта операция имеет два прототипа, описанные ниже:

Первый прототип SequenceEqual

C#

```
public static bool SequenceEqual<T>(
       this IEnumerable<T> first,
       IEnumerable<T> second);
```

Эта операция перечисляет каждую входную последовательность параллельно, сравнивая элементы с помощью метода System.Object.Equals. Если элементы эквивалентны, и последовательности содержат одинаковое количество элементов, операция возвращает true. Иначе она возвращает false.

Второй прототип SequenceEqual

Второй прототип операции работает так же, как и первый, за исключением того, что принимает объект IEqualityComparer<T>, который может быть использован для проверки эквивалентности:

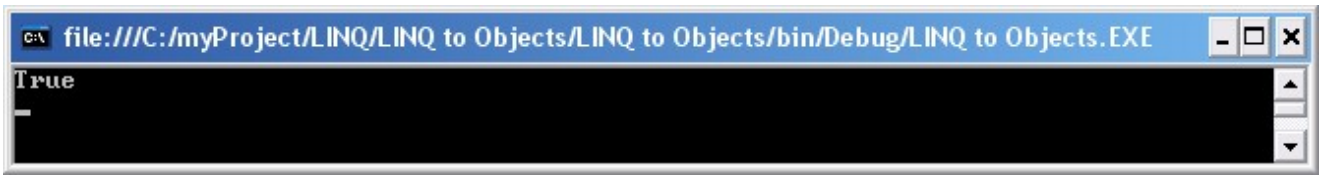
C#

```
public static bool SequenceEqual<T>(
        this IEnumerable<T> first,
        IEnumerable<T> second,
        IEqualityComparer<T> comparer);
```

Если любой из аргументов равен null, генерируется исключение ArgumentNullException. Пример приведен ниже:

```
string[] cars = { "Alfa Romeo", "Aston Martin", "Audi", "Nissan", "Chevrolet",  "Chrysler",
"Dodge", "BMW",
                            "Ferrari",  "Bentley", "Ford", "Lexus", "Mercedes", "Toyota",
"Volvo", "Subaru", "Жигули :)"};

bool eq = cars.SequenceEqual(cars.Take(cars.Count()));
Console.WriteLine(eq);
```

В приведенном коде с помощью операции Take были выбраны только первые N элементов из массива cars, и полученная выходная последовательность сравнивалась с исходным массивом cars. Итак, если в приведенном выше коде взять все элементы массива cars, указав количество элементов через cars.Count(), то будет получена вся выходная последовательность целиком. Как и следовало ожидать, вот результат:



Все работает, как и должно. Теперь возьмем все элементы кроме последнего, вычтя единицу из cars.Count(), как показано ниже:

C#

```
bool eq = cars.SequenceEqual(cars.Take(cars.Count() - 1));
Console.WriteLine(eq);
```

Теперь результат должен быть false, потому что две последовательности имеют разную длину. Во второй из них недостает последнего элемента.

В примере использования второго прототипа создаются два массива типа string, где каждый элемент представляет собой число в строковой форме. Элементы двух массивов будут такими, что при преобразовании в числа окажутся эквивалентными. В этом примере задействован общий класс MyStringifiedNumberComparer:

C#

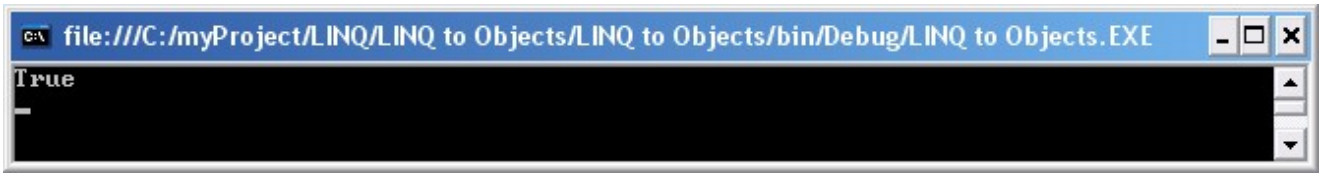
Пройди тесты

C# тест (легкий) (https://professorweb.ru/test/c-sharp-test.html)

↑ .NET тест (средний) (https://professorweb.ru/test/asp-test.html)

```
string[] strArr1 = { "0012", "130", "0000019", "4" };
string[] strArr2 = { "12", "0130", "019", "0004" };

bool eq = strArr1.SequenceEqual(strArr2,
          new MyStringifiedNumberComparer());
Console.WriteLine(eq);

...

public class MyStringifiedNumberComparer : IEqualityComparer<string>
  {
    public bool Equals(string x, string y)
    {
      return (Int32.Parse(x) == Int32.Parse(y));
    }

    public int GetHashCode(string obj)
    {
      return Int32.Parse(obj).ToString().GetHashCode();
    }
  }
```

Если преобразовать каждый элемент обоих массивов в целое число, а затем сравнить соответствующие числа, то эти два массива должны считаться эквивалентными. Это подтверждает результат:



Назад (3_3.php)	3	4	5	Вперед (3_5.php)