Операции ThenBy и ThenByDescending

Вызовы ThenBy и ThenByDescending могут соединяться в цепочку, т.к. они принимают в качестве входной последовательности IOrderedEnumerable<T> и возвращают в качестве выходной последовательности тоже IOrderedEnumerable<T>.

Например, следующая последовательность вызовов не разрешена:

C#

```
inputSequence.OrderBy(s => s.LastName).OrderBy(s => s.FirstName)...
```

Вместо нее должна использоваться такая цепочка:

C#

```
inputSequence.OrderBy(s => s.LastName).ThenBy(s => s.FirstName)...
```

ThenBy

Операция ThenBy позволяет упорядочивать входную последовательность типа IOrderedEnumerable<T> на основе метода keySelector, который возвращает значение ключа. В результате выдается упорядоченная последовательность типа IOrderedEnumerable<T>.

В отличие от большинства операций отложенных запросов LINQ to Objects, операции ThenBy и ThenByDescending принимают другой тип входных последовательностей - **IOrderedEnumerable<T>**. Это значит, что сначала должна быть вызвана операция OrderBy или OrderByDescending для создания последовательности IOrderedEnumerable, на которой можно затем вызывать операции ThenBy и ThenByDescending.

Пройди тесты

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

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

Сортировка, выполняемая операцией ThenBy, является устойчивой. Другими словами, она сохраняет входной порядок элементов с эквивалентными ключами. Если два входных элемента поступили в операцию ThenBy в определенном порядке, и ключевое значение обоих элементов одинаково, то порядок тех же выходных элементов гарантированно сохранится. В отличие от OrderBy и OrderByDescending, операции ThenBy и ThenByDescending выполняют устойчивую сортировку.

Операция ThenBy имеет два прототипа, которые описаны ниже:

Первый прототип ThenBy

C#

```
public static IOrderedEnumerable<T> ThenBy<T, K> (
       this IOrderedEnumerable<T> source,
       Func<T, K> keySelector)
   where
       K : IComparable<K>;
```

В этом прототипе операции ThenBy упорядоченная входная последовательность типа IOrderedEnumerable<T> передается операции ThenBy наряду с делегатом метода keySelector. Метод keySelector принимает элемент типа T и возвращает поле внутри элемента, которое используется в качестве значения ключа с типом K для входного элемента. Типы T и K могут быть как одинаковыми, так и различными. Значение, возвращенное методом keySelector, должно реализовывать интерфейс IComparable. Операция ThenBy упорядочит входную последовательность по возрастанию на основе возвращенных ключей.

Второй прототип ThenBy

C#

```
public static IOrderedEnumerable<T> ThenBy<T, K>(
     this IOrderedEnumerable<T> source,
     Func<T, K> keySelector,
     IComparer<K> comparer);
```

Этот прототип подобен первому, за исключением того, что принимает объект-компаратор. Если используется эта версия операции ThenBy, то типу K не обязательно реализовывать интерфейс IComparable.

Пройди тесты

Ниже показан пример использования первого прототипа:

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

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

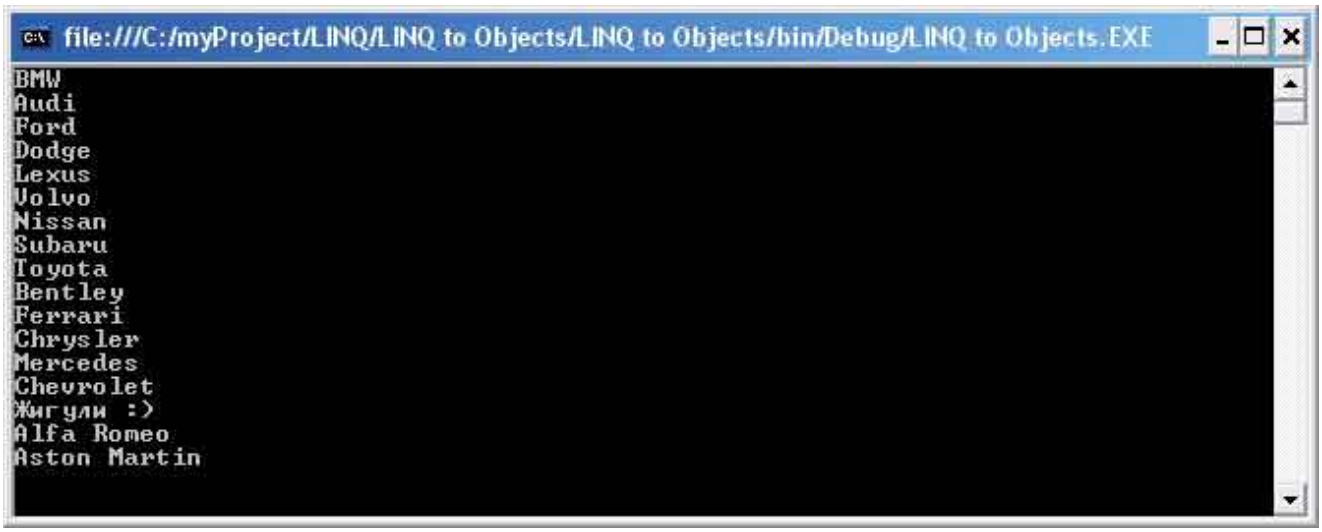
C#

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

IEnumerable<string> auto = cars.OrderBy(s => s.Length).ThenBy(s => s);

foreach (string str in auto)
        Console.WriteLine(str);
```

Этот код сначала упорядочивает элементы по их длине, в данном случае — длине названия автомобиля. Затем упорядочивает по самому элементу. В результате получается список названий, отсортированный по длине от меньшей к большей (по возрастанию), а затем — по имени в алфавитном порядке:



В примере применения второго прототипа ThenBy снова будет использоваться показанный в предыдущей статье объект-компаратор MyVowelToConsonantRatioComparer. Однако перед вызовом ThenBy сначала потребуется вызвать либо OrderBy, либо OrderByDescending. В данном примере будет вызван OrderBy, чтобы упорядочить список по количеству символов в имени. Таким образом, имена будут упорядочены по возрастанию количества букв в них, а затем — внутри каждой группы с одинаковой длиной — по соотношению гласных и согласных:

C#

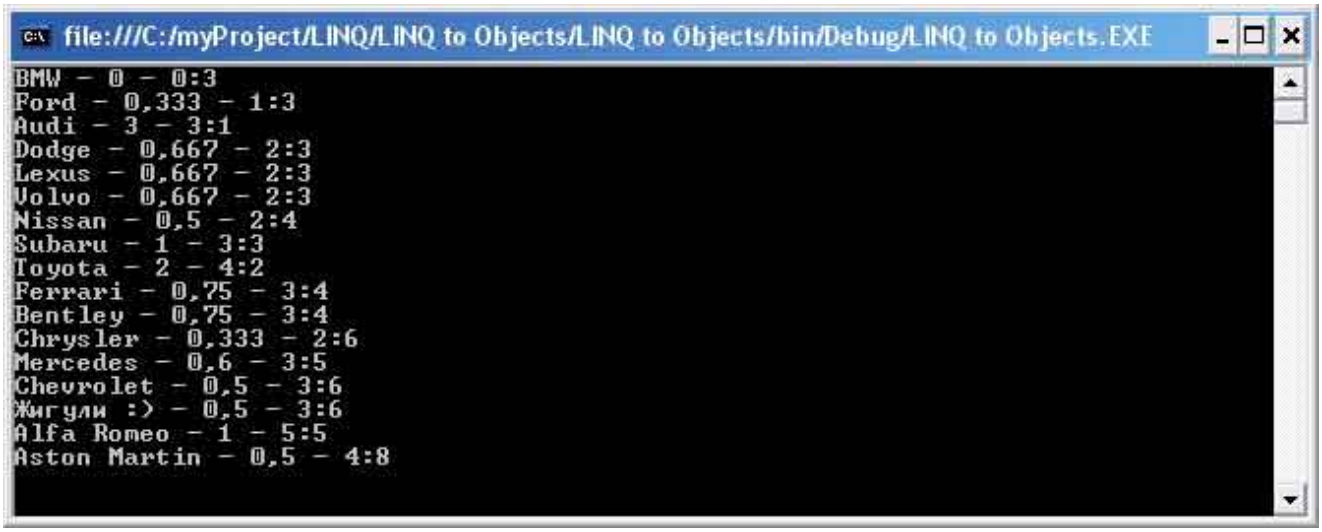
Пройди тесты

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

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

```
MyVowelToConsonantRatioComparer myComp = new MyVowelToConsonantRatioComparer();
            IEnumerable<string> auto = cars
                .OrderBy(s => s.Length)
                .ThenBy((s => s), myComp);

            foreach (string str in auto)
            {
                int vCount = 0;
                int cCount = 0;
                myComp.GetVowelConsonantCount(str, ref vCount, ref cCount);
                double dRatio = Math.Round(((double)vCount / (double)cCount), 3);
                Console.WriteLine(str + " - " + dRatio + " - " + vCount + ":" + cCount);
            }
```



Как и следовало ожидать, имена сначала упорядочены по длине, а затем — по соотношению гласных и согласных.

ThenByDescending

Эта операция прототипирована и ведет себя подобно операции ThenBy, за исключением того, что упорядочивает элементы по убыванию. Ниже показан пример использования операции ThenByDescending:

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

IEnumerable<string> auto = cars.OrderBy(s => s.Length).ThenByDescending(s => s);

foreach (string str in auto)
        Console.WriteLine(str);
```

В примере применения первого прототипа операции ThenByDescending используется тот же базовый подход, что и в примере вызова первого прототипа операции ThenBy, за исключением того, что вместо ThenBy будет вызываться ThenByDescending.

Этот код порождает вывод, где имена в пределах группы с одной и той же длиной сортируются по алфавиту в обратном порядке тому, который обеспечивает операция ThenBy:

Назад (2_5.php)	5	6	7	Вперед (2_7.php)

Пройди тесты 

Лучший чат для C# программистов (https://t.me/professorweb)

Professor Web (/)

Наш любимый хостинг (/)

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

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