Операции Min и Max

Min

Операция Min возвращает минимальное значение входной последовательности. Эта операция имеет четыре прототипа, которые описаны ниже:

Первый прототип Min

C#

```
public static Numeric Min (
        this IEnumerable<Numeric> source);
```

Тип Numeric должен быть одним из int, long, double или decimal, либо одним из их допускающих null эквивалентов: int?, long?, double? или decimal?.

Первый прототип операции Min возвращает элемент с минимальным числовым значением из входной последовательности source. Если тип элемента реализует интерфейс IComparable<T>, этот интерфейс применяется для сравнения элементов. Если же элемент не реализует интерфейс IComparable<T>, будет использован необобщенный интерфейс IComparable. Пустая последовательность, либо последовательность, состоящая только из значений null, вернет null.

Второй прототип Min

Второй прототип операции Min ведет себя подобно первому, за исключением того, что он предназначен для не числовых типов.

C#

```
public static T Min<T>(
    this IEnumerable<T> source);
```


Третий прототип Min

Третий прототип предназначен для типов Numeric и подобен первому, но с возможностью указания делегата метода selector, который позволяет сравнивать члены каждого элемента входной последовательности в процессе поиска минимального значения и возвращать это минимальное значение.

C#

```
public static T Min<T>(
       this IEnumerable<T> source,
       Func<T, Numeric> selector);
```

Четвертый прототип Min

Четвертый прототип предназначен для не числовых типов и подобен второму, но с возможностью указания делегата метода selector, который позволяет сравнивать члены каждого элемента входной последовательности в процессе поиска минимального значения и возвращать это минимальное значение.

C#

```
public static S Min<T, S>(
        this IEnumerable<T> source,
        Func<T, S> selector);
```

Если любой из аргументов равен null, генерируется исключение ArgumentNullException. Если последовательность source пуста для Numeric-версий прототипов и если тип T не допускает значений null, такой как int, long, double или decimal, генерируется исключение InvalidOperationException. Если типы допускают null, подобно int?, long?, double? или decimal?, то вместо этого операция возвращает null.

Ниже показан пример использования всех вышеперечисленных прототипов:

C#

Пройди тесты

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

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

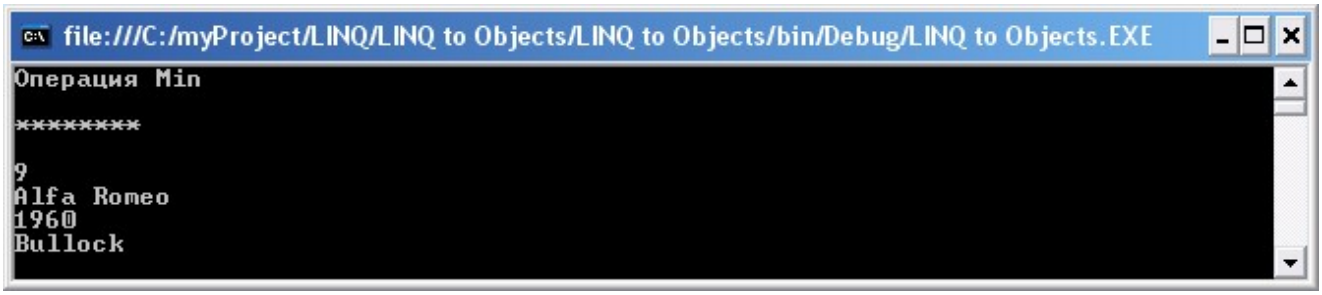
```
string[] cars = { "Alfa Romeo", "Aston Martin", "Audi", "Nissan", "Chevrolet",  "Chrysler",
"Dodge", "BMW",
                            "Ferrari",  "Bentley", "Ford", "Lexus", "Mercedes", "Toyota",
"Volvo", "Subaru", "Жигули :)"};

Console.WriteLine("Операция Min\n\n********\n");

int[] nums = { 34, 578, 200, 9, 72, 1399};

// Первый прототип
int min = nums.Min();
Console.WriteLine(min);

// Второй прототип
string auto = cars.Min();
Console.WriteLine(auto);

// Третий прототип
min = Actor.GetActors().Min(a => a.birthYear);
Console.WriteLine(min);

// Четвертый прототип
string actor = Actor.GetActors().Min(a => a.lastName);
Console.WriteLine(actor);

...

public class Actor
  {
    public int birthYear;
    public string firstName;
    public string lastName;

    public static Actor[] GetActors()
    {
      Actor[] actors = new Actor[] {
        new Actor { birthYear = 1964, firstName = "Keanu", lastName = "Reeves" },
        new Actor { birthYear = 1968, firstName = "Owen", lastName = "Wilson" },
        new Actor { birthYear = 1960, firstName = "James", lastName = "Spader" },
        new Actor { birthYear = 1964, firstName = "Sandra", lastName = "Bullock" },
      };

      return (actors);
    }
  }
```

В примере использования первого прототипа Min, объявляется массив целых чисел и затем возвращается минимальное из них. Это очень тривиальный пример, и его результат равен 9.

Пройди тесты

В примере использования второго прототипа, операция Min просто вызывается на стандартном массиве cars. Она должна вернуть элемент с минимальным значением в алфавитном порядке. В примере применения третьего прототипа операции Min

используется общий класс Actor для поиска самой ранней даты рождения актера посредством вызова Min на годах рождения. Ну и наконец, в примере четвертого прототипа снова задействован класс Actor - здесь извлекается фамилия актера, которая находится первой в алфавитном порядке. Результат показан ниже:



Max

Операция Max возвращает максимальное значение из входной последовательности. Эта операция имеет четыре прототипа, полностью идентичных прототипам операции Min. Ниже показан пример, аналогичный примеру с использованием операции Min (не забудьте добавить класс Actor):

C#

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

Console.WriteLine("Операция Max\n\n********\n");

int[] nums = { 34, 578, 200, 9, 72, 1399};

// Первый прототип
int Max = nums.Max();
Console.WriteLine(Max);

// Второй прототип
string auto = cars.Max();
Console.WriteLine(auto);

// Третий прототип
Max = Actor.GetActors().Max(a => a.birthYear);
Console.WriteLine(Max);

// Четвертый прототип
string actor = Actor.GetActors().Max(a => a.lastName);
Console.WriteLine(actor);
```

Назад (3_8.php)	8	9	10	Вперед (3_10.php)

Пройди тесты

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

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