Лабораторная работа №6

«LINQ»

**Цели работы:**

1. Научиться работать с LINQ средствами языка C#.

**Задание№1**

Зарегистрируйтесь на сайте https://openweathermap.org/ для получения ключа (API key) к API от сервиса погоды.

Создайте структуру Weather, содержащую свойства Country(страна), Name(город или название местности), Temp(температура воздуха), Description(описание погоды).

Используя API, получите не менее 50 значений текущей погоды в разных точках мира.

Используйте запрос вида:

https://api.openweathermap.org/data/2.5/weather?lat={Широта}&lon={Долгота}&appid={API key}

, где:

Широта - дробная величина в диапазоне от -90 до 90.

Долгота - дробная величина в диапазоне от -180 до 180.

API key - ключ, полученный при регистрации на сайте https://openweathermap.org/.

Значения Широты и Долготы изменяйте случайным образом в заданных диапазонах, если для полученной координаты нет значения Country или Name, **следует сгенерировать новые координаты**.

На основе полученных данных создайте и заполните коллекцию структур Weather.

С помощью LINQ запросов к созданной коллекции, получите и выведите на консоль следующие данные:

1. Страну с максимальной и минимальной температурой.
2. Среднюю температуру в мире.
3. Количество стран в коллекции.
4. Первую найденную страну и название местности, в которых Description принимает значение: "clear sky","rain","few clouds"

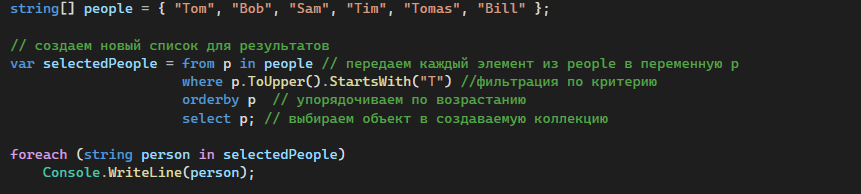
**Теоретические сведения**

**LINQ**

LINQ (Language-Integrated Query) представляет простой и удобный язык запросов к источнику данных. В качестве источника данных может выступать объект, реализующий интерфейс IEnumerable (например, стандартные коллекции, массивы), набор данных DataSet, документ XML. Но вне зависимости от типа источника LINQ позволяет применить ко всем один и тот же подход для выборки данных.

Существует несколько разновидностей LINQ:

* LINQ to Objects: применяется для работы с массивами и коллекциями
* LINQ to Entities: используется при обращении к базам данных через технологию Entity Framework
* LINQ to XML: применяется при работе с файлами XML
* LINQ to DataSet: применяется при работе с объектом DataSet
* Parallel LINQ (PLINQ): используется для выполнения параллельных запросов



Выражение from p in people проходит по всем элементам массива people и определяет каждый элемент как p. Используя переменную p мы можем проводить над ней разные операции.

Несмотря на то, что мы не указываем тип переменной p, выражения LINQ являются строго типизированными. То есть среда автоматически распознает, что набор people состоит из объектов string, поэтому переменная p будет рассматриваться в качестве строки.

Далее с помощью оператора where проводится фильтрация объектов, и если объект соответствует критерию (в данном случае начальная буква должна быть "T"), то этот объект передается дальше.

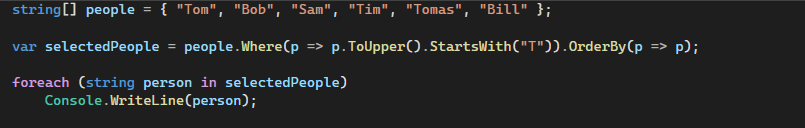
Оператор orderby упорядочивает по возрастанию, то есть сортирует выбранные объекты.

Оператор select передает выбранные значения в результирующую выборку, которая возвращается LINQ-выражением.

В данном случае результатом выражения LINQ является объект IEnumerable<T>. Нередко результирующая выборка определяется с помощью ключевого слова var, тогда компилятор на этапе компиляции сам выводит тип.

### Методы расширения LINQ

Кроме стандартного синтаксиса from .. in .. select для создания запроса LINQ мы можем применять специальные методы расширения, которые определены для интерфейса IEnumerable. Как правило, эти методы реализуют ту же функциональность, что и операторы LINQ типа where или orderby.



Запрос people.Where(p=>p.ToUpper().StartsWith("T")).OrderBy(p => p) будет аналогичен предыдущему. Он состоит из цепочки методов Where и OrderBy. В качестве аргумента эти методы принимают делегат или лямбда-выражение.

### Список используемых методов расширения LINQ

* Select: определяет проекцию выбранных значений
* Where: определяет фильтр выборки
* OrderBy: упорядочивает элементы по возрастанию
* OrderByDescending: упорядочивает элементы по убыванию
* ThenBy: задает дополнительные критерии для упорядочивания элементов возрастанию
* ThenByDescending: задает дополнительные критерии для упорядочивания элементов по убыванию
* Join: соединяет две коллекции по определенному признаку
* Aggregate: применяет к элементам последовательности агрегатную функцию, которая сводит их к одному объекту
* GroupBy: группирует элементы по ключу
* ToLookup: группирует элементы по ключу, при этом все элементы добавляются в словарь
* GroupJoin: выполняет одновременно соединение коллекций и группировку элементов по ключу
* Reverse: располагает элементы в обратном порядке
* All: определяет, все ли элементы коллекции удовлетворяют определенному условию
* Any: определяет, удовлетворяет хотя бы один элемент коллекции определенному условию
* Contains: определяет, содержит ли коллекция определенный элемент
* Distinct: удаляет дублирующиеся элементы из коллекции
* Except: возвращает разность двух коллекцию, то есть те элементы, которые создаются только в одной коллекции
* Union: объединяет две однородные коллекции
* Intersect: возвращает пересечение двух коллекций, то есть те элементы, которые встречаются в обоих коллекциях
* Count: подсчитывает количество элементов коллекции, которые удовлетворяют определенному условию
* Sum: подсчитывает сумму числовых значений в коллекции
* Average: подсчитывает среднее значение числовых значений в коллекции
* Min: находит минимальное значение
* Max: находит максимальное значение
* Take: выбирает определенное количество элементов
* Skip: пропускает определенное количество элементов
* TakeWhile: возвращает цепочку элементов последовательности, до тех пор, пока условие истинно
* SkipWhile: пропускает элементы в последовательности, пока они удовлетворяют заданному условию, и затем возвращает оставшиеся элементы
* Concat: объединяет две коллекции
* Zip: объединяет две коллекции в соответствии с определенным условием
* First: выбирает первый элемент коллекции
* FirstOrDefault: выбирает первый элемент коллекции или возвращает значение по умолчанию
* Single: выбирает единственный элемент коллекции, если коллекция содержит больше или меньше одного элемента, то генерируется исключение
* SingleOrDefault: выбирает единственный элемент коллекции. Если коллекция пуста, возвращает значение по умолчанию. Если в коллекции больше одного элемента, генерирует исключение
* ElementAt: выбирает элемент последовательности по определенному индексу
* ElementAtOrDefault: выбирает элемент коллекции по определенному индексу или возвращает значение по умолчанию, если индекс вне допустимого диапазона
* Last: выбирает последний элемент коллекции
* LastOrDefault: выбирает последний элемент коллекции или возвращает значение по умолчанию