# Лабораторная работа № 8. LINQ

## *1. Цель работы*

Научиться использовать делегаты и события.

## *2. Задание*

В лабораторной работе требуется создать обобщенный класс CollectionType<T>. Определить в классе конструкторы, деструктор, методы добавления и удаления элементов, другие необходимые методы и, если требуется, перегруженные операции. Определить индексаторы и свойства. CollectionType можно реализовать на основе стандартных коллекций (List, Stack, Array и т.д.).

Возьмите созданный тип (класс) из предыдущих лабораторных работ и реализуйте в нем интерфейс IComparable<T>. Используйте данный класс в качестве параметра вашего обобщенного класса. Создайте несколько коллекций. Выполните сохранение в файл, сортировку, LINQ-запросы в соответствии с вариантом.

Выполните несколько сложных LINQToObject запросов (минимум 5) к коллекции объектов, используя одновременно более трех операций (пример: where + select + orderBy, first + any + min).

Создайте обобщённую стандартную коллекцию из пространства имен System.Collections указанную в варианте со строками и выполните ввод-вывод, сохранение в файл, поиск строк, содержащих определенное значение, подсчет количества строк длины n, сортировку в возрастающем и убывающем порядке.

## *3. Базовые сведения*

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.

**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
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 содерятся 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 первый элемент коллекции или возвращает значение по умолчанию
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

## 4. Варианты заданий

Варианты:

1, 8 создать массив объектов CollectionType, запросы – найти коллекции размера n; найти максимальную и минимальную коллекцию в массиве по количеству элементов. Обобщенная коллекция – LinkedList<T>

2, 9 создать массив объектов CollectionType, запросы – найти коллекции с отрицательными элементами (выбрать любое поле объекта), найти максимальную и минимальную коллекцию в массиве, содержащую указанный элемент. Обобщенная коллекция – Dictionary<T>.

3, 10 создать массив объектов CollectionType, запросы – найти количество коллекций равных заданному размеру, найти максимальную и минимальную коллекцию в массиве. Обобщенная коллекция – List<T>

4, 11 создать массив объектов CollectionType, запросы – найти количество коллекций, содержащих только 2 элемента, найти максимальную и минимальную коллекцию в массиве по заданному значению поля объекта (можно выбрать любое поле). Обобщенная коллекция – List<T>

5, 12 создать массив объектов CollectionType, запросы – найти количество коллекций, содержащих указанный элемент, найти максимальную коллекцию, содержащую указанный элемент. Обобщенная коллекция – Dictionary<T>.

6, 13 создать массив объектов CollectionType, запросы – найти количество коллекций, содержащих заданное значение (выбрать любое поле объекта), найти максимальную и минимальную коллекцию в массиве. Обобщенная коллекция – LinkedList<T>

7, 14 создать массив объектов CollectionType, запросы – найти количество коллекций, сумма которых больше указанного значения (для суммирования выбрать любое поле объекта), найти максимальную и минимальную коллекцию в массиве. Обобщенная коллекция – ArrayList<T>

## 5. Требования к оформлению отчета:

- титульный лист;
- название;
- цель работы;
- лабораторное задание;
- описание метода решения задачи;
- листинг (текст программы);
- пояснительный текст к программе;
- результаты работы программы;
- выводы.

## 6. Контрольные вопросы

1) Каким образом в языке C# используется обобщения?
2) Что делает ключевое слово «where» при определении класса?

3) Для какой цели класс может реализовывать интерфейс IComparable? Что описывает данный интерфейс?

4) Объясните назначение интерфейса IEnumerable. Какие методы придется реализовать для того, чтобы воспользоваться данным интерфейсом?

5) Что такое «Итератор». Какой интерфейс описывает свойства и поведение объекта-итератора? Объясните принцип работы итераторов в языке C#. Поясните принцип работы индексатора.

6) Для чего используется язык интегрированных запросов (Language Integrated Query)?

7) Приведите пример отложенных запросов и тех, что выполняются сразу, в языке LINQ.

8) В чем преимущества отложенных запросов?

9) Каким образом LINQ использует лямбда-выражения?

## *7. Список рекомендованной литературы*


1. Васильев А. C#. Объектно-ориентированное программирование: Учебный курс. – СПб.: Питер, 2012. – 320 с.: ил.
2. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного программирования. Паттерны проектирования. – СПб: Питер, 2009. – 366 с.: ил.
3. Герберт Шилдт. C# 3.0. Полное руководство. - Изд. Вильямс, 2010.
4. Нейгел К., Ивьен Б. и др. C# 2008 и платформа NET 3.5 для профессионалов. – Изд. Диалектика, 2008.
5. Трей Нэш. C# 2010. Ускоренный курс для профессионалов. - Изд. Вильямс, 2010.
6. Троелсен Э. Язык программирования C# 2008 и платформа .NET 3.5- Изд. Вильямс, 2010.
7. Стиллмен Э., Грин Дж. Изучаем C# [пер. с англ. И. Рузмайкина]. - 2-е изд. - Москва: Питер, 2012. – 694 с. : ил.