

Лабораторная работа №5

Создание LINQ-запросов в C# для выборки данных из коллекции. Технология LINQ to Object.

Запустим **Visual Studio**, закажем новый проект шаблона **Windows Forms Application C#**. Первая задача заключается в том, чтобы создать список сотрудников предприятия и из этого списка выбрать некоторых сотрудников по какому-либо признаку, например, по стажу. При создании списка объявлен класс *Сотрудник*, который содержит три свойства: **ФИО**, **Возраст** и **Стаж**. В начале решения заполняем список сотрудников, а затем строим LINQ-запрос для заполнения списка сотрудников со стажем больше 10 лет. Этот список выводим в текстовое поле **textBox1**, используя цикл **foreach**.

Затем из панели элементов перенесем в форму текстовое поле. Далее через щелчок правой кнопкой мыши перейдем к вкладке программного кода. Пошагово пропишем код:

1. Создать класс *Сотрудник* с полями: ФИО, Возраст, Стаж работы
2. Объявить и заполнить коллекцию (список) *Сотрудники*

```
List<тип_данных> Сотрудники = new List<тип_данных> {  
    new Сотрудник {ФИО="Карапузова Ирина", Возраст=27, Стаж=10}  
    . . .    };
```

В качестве типа данных необходимо указать имя класса, созданного в п.1.

3. Создать **LINQ-запрос** для вывода ФИО и возраста сотрудников с стажем больше 10 лет

```
var Результат = from Сотрудник in Сотрудники  
                where условие (Сотрудник.Стаж>10)  
                orderby имя_поля //сортировка по полю  
                select Сотрудник;
```

4. Вывести результат запроса в **TextBox**, используя цикл **foreach**

```

        foreach (var x in Результат)
        {
            textBox1.Text = textBox1.Text + string.Format("{0} -   возраст " + "- {1}\r\n",
                x.ФИО, x.Возраст);
        }
    }
}

```

Решим еще одну задачу, когда в списке необходимо использовать еще вложенный (внутренний) список. Требуется создать список студентов факультета, содержащий фамилию студента и список полученных им текущих оценок, т. е. список оценок должен быть "вложен" в список студентов. Из списка студентов необходимо выбрать тех, кто имеет в списке своих оценок хотя бы одну двойку. Для решения этой задачи вначале объявляем новый класс *Студент*, который имеет в качестве свойств класса фамилию студента и список (типа **List**) оценок. Затем строим **LINQ-запрос**, где, поскольку нам необходимо искать элемент списка внутри "вложенного" списка, мы используем предложение **From** два раза.

1. Объявляем класс *Студент*, содержащий фамилию студента и список полученных им оценок. Для хранения оценок используем СПИСОК: `List<int>` Оценки

2. Создадим список *Студенты* и заполним его значениями

```

List<тип> Студенты = new List<тип>
{
    new Студент {Фамилия="Иванов", Оценки= new List<int> {5, 4, 4, 5}}
    . . . . }

```

В качестве типа данных необходимо указать класса *Студент*

3. Создадим **LINQ-запрос** для выбора студентов, имеющих двойки. Для доступа к внутреннему списку предложение **from** используется еще раз

```

var СписокДвоечников = from Студент in Студенты
                        from Оценка in Студент.Оценки
                        where Оценка <= 2

```

```
orderby Студент.Фамилия  
select new { Студент.Фамилия, Оценка };
```

4. Вывести полученный результат в текстовое поле

Задачи для самостоятельного выполнения (блок 2):

Внимание! Данные необходимо хранить в списках (коллекции). Добавьте возможность добавления информации в коллекции.

1. Даны сведения о счетах клиентов в банке: ФИО клиента, номер счета (пример: 40562**810**45718745, 40562**840**45718745, 40562**978**45718745), сумма остатка. Вывести общую сумму одного клиента по всем счетам в указанной валюте. Валюта определяется по выделенным в образце цифрам (810 – рубли, 978 – евро, 840 – доллары) в номере счета. Пользователь должен иметь возможность выбора валюты и ввода фамилии клиента.

Руб. - 195000

Eur - 1000

USD – 500

2. Исходная последовательность содержит сведения о клиентах фитнес-центра. Каждый элемент последовательности включает следующие целочисленные поля:

<Код клиента> <Год> <Номер месяца>

<Продолжительность занятий (в часах)>

Найти элемент последовательности с минимальной продолжительностью занятий. Вывести эту продолжительность, а также соответствующие ей год и номер месяца (в указанном порядке на той же строке). Если имеется несколько элементов с минимальной продолжительностью, то вывести данные того из них, который является последним в исходной последовательности.

3. Исходная последовательность содержит сведения об абитуриентах.

Каждый элемент последовательности включает следующие поля:

<Номер школы> <Год поступления> <Фамилия>

Для каждого года, присутствующего в исходных данных, вывести число различных школ, которые окончили абитуриенты, поступившие в этом году (вначале указывать число школ, затем год). Сведения о каждом годе выводить на новой строке и упорядочивать по возрастанию числа школ, а для совпадающих чисел — по возрастанию номера года.

4. Исходная последовательность содержит сведения о задолжниках по оплате коммунальных услуг, живущих в 144-квартирном 9-этажном доме. Каждый элемент последовательности включает следующие поля:

<Задолженность> <Фамилия> <Номер квартиры>

Задолженность указывается в виде дробного числа (целая часть — рубли, дробная часть — копейки). В каждом подъезде на каждом этаже располагаются по 4 квартиры. Для каждого из 4 подъездов дома найти трех жильцов с наибольшей задолженностью и вывести сведения о них: задолженность (выводится с двумя дробными знаками), номер подъезда, номер квартиры, фамилия жильца. Считать, что в наборе исходных данных все задолженности имеют различные значения. Сведения о каждом задолжнике выводить на отдельной строке и упорядочивать по убыванию размера задолженности (номер подъезда при сортировке не учитывать). Если в каком-либо подъезде число задолжников меньше трех, то включить в полученный набор всех задолжников этого подъезда.

5. Исходная последовательность содержит сведения об автозаправочных станциях (АЗС). Каждый элемент последовательности включает следующие поля:

<Компания> <Марка бензина> <Цена 1 литра (в копейках)> <Улица>

Названия компаний и улиц не содержат пробелов. В качестве марки бензина указываются числа 92, 95 или 98. Каждая компания имеет не более одной АЗС на каждой улице; цены на разных АЗС одной и той же компании могут различаться. Для каждой марки бензина, присутствующей в исходных данных, определить минимальную и максимальную цену литра бензина этой марки (вначале выводить марку, затем цены в указанном порядке). Сведения о каждой марке выводить на новой строке и упорядочивать по убыванию значения марки.