#### Московский государственный технический университет имени Н.Э. Баумана

## Лабораторная работа по дисциплине БКИТ №3

Выполнил: Байбарин Р.Г., ИУ5-34, 18.10.2017

### I. Описание задания

Разработать программу, реализующую работу с LINQ to Objects. В качестве примера используйте проект «SimpleLINQ» из примера «Введение в LINQ».

1. Программа должна быть разработана в виде консольного приложения на языке C#.

2. Создайте класс «Сотрудник», содержащий поля:

* ID записи о сотруднике;
* Фамилия сотрудника;

3. Создайте класс «Отдел», содержащий поля:

* ID записи об отделе;
* Наименование отдела.

4. Предполагая, что «Отдел» и «Сотрудник» связаны соотношением один-ко-многим разработайте следующие запросы:

* Выведите список всех сотрудников и отделов, отсортированный по отделам.
* Выведите список всех сотрудников, у которых фамилия начинается с буквы «А».
* Выведите список всех отделов и количество сотрудников в каждом отделе.
* Выведите список отделов, в которых у всех сотрудников фамилия начинается с буквы «А».
* Выведите список отделов, в которых хотя бы у одного сотрудника фамилия начинается с буквы «А».

5. Создайте класс «Сотрудники отдела», содержащий поля:

* ID записи о сотруднике;
* ID записи об отделе.

6. Предполагая, что «Отдел» и «Сотрудник» связаны соотношением много-ко-многим с использованием класса «Сотрудники отдела» разработайте следующие запросы:

* Выведите список всех отделов и список сотрудников в каждом отделе.
* Выведите список всех отделов и количество сотрудников в каждом отделе.

### 2. Код программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab7

{

class Program

{

static List<office.Worker> workers = new List<office.Worker>()

{

new office.Worker(1, "Лескина "),

new office.Worker(2, "Брысина "),

new office.Worker(3, "Байбарин "),

new office.Worker(4, "Гаврилюк "),

new office.Worker(5, "Баскакова "),

new office.Worker(6, "Кондрашева "),

new office.Worker(7, "Тимаков "),

new office.Worker(8, "Авдеев "),

new office.Worker(9, "Александрова "),

};

static List<office.Department> rooms = new List<office.Department>()

{

new office.Department(1, "Отдел финансов "),

new office.Department(2, "Общий отдел "),

new office.Department(3, "Отдел кадров "),

new office.Department(4, "IT отдел ")

};

static List<office.Relations> rel = new List<office.Relations>()

{

new office.Relations(1, 1),

new office.Relations(2, 4),

new office.Relations(3, 2),

new office.Relations(4, 3),

new office.Relations(5, 3),

new office.Relations(6, 4),

new office.Relations(7, 1),

new office.Relations(8, 2),

new office.Relations(9, 2),

new office.Relations(6, 3),

new office.Relations(7, 2),

new office.Relations(8, 4),

new office.Relations(9, 1)

};

static void Main(string[] args)

{

Console.WriteLine("Перечисление всех сотрудников\n" +

"ID|Фамилия|");

var i1 = from x in workers select x;

foreach (var x in i1) Console.WriteLine(x);

Console.WriteLine("\nПеречисление всех отделов\n" +

"ID|Название отдела");

var i2 = from x in rooms select x;

foreach (var x in i2) Console.WriteLine(x);

Console.WriteLine("\nМежду сотрудниками и отделами существует свзь М:М\n" +

"Количество сотрудников в каждом отделе\n" +

"ID|Название отдела|Количество сотрудников");

foreach (var x in rooms)

{

var i8 = from y in rel

where (y.id\_department == x.id\_department)

select y;

Console.WriteLine(x + " " + i8.Count()+"|");

}

Console.WriteLine("\nСотрудники каждого отдела");

foreach (var x in rooms)

{

//Перебор по связям отдел-сотрудник

var i6 = from y in rel

where (y.id\_department == x.id\_department)

select y;

//Перебор по списку сотрудников

var i7 = from y in workers

from z in i6

where (z.id\_worker == y.id\_worker)

select y;

Console.WriteLine(x.id\_department+"-------"+x.title);

foreach (var y in i7) Console.WriteLine(y);

}

Console.WriteLine("\nСотрудники, состоящие в 2-х и более отделах");

foreach (var x in workers)

{

var i11 = from y in rel

where (x.id\_worker == y.id\_worker)

select y;

if (i11.Count() > 1)

Console.WriteLine(x);

}

Console.ReadKey();

}

}

}

namespace office

{

class Worker

{

public int id\_worker;

public string surname;

public Worker() { }

public Worker(int a, string s)

{

this.id\_worker = a;

this.surname = s;

}

public override string ToString()

{

StringBuilder b = new StringBuilder();

return this.id\_worker.ToString() + "| " + this.surname + b.Append(Convert.ToChar(32), 15 - this.surname.Length)+"|";

}

}

class Department

{

public int id\_department;

public string title;

public Department() { }

public Department(int a, string s)

{

this.id\_department = a;

this.title = s;

}

public override string ToString()

{

StringBuilder b = new StringBuilder();

return this.id\_department.ToString() + "| " + this.title.ToString() + b.Append(Convert.ToChar(32), 15 - this.title.Length) + "|";

}

}

class Relations

{

public int id\_worker;

public int id\_department;

public Relations(int a, int b)

{

this.id\_worker = a;

this.id\_department = b;

}

public override string ToString()

{

return this.id\_worker.ToString() + "| " + this.id\_department + "|";

}

}

}

### 3. Примеры работы программы



