Московский государственный технический университет имени Н.Э. Баумана (МГТУ им. Н.Э. Баумана)

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7

ПО ДИСЦИПЛИНЕ «БАЗОВЫЕ КОМПОНЕНТЫ ИНТЕРНЕТ ТЕХНОЛОГИЙ»

Выполнил(а): Белов Ф.А.

Проверил(а): Гапанюк Ю.Е.

Москва, 2017

**Формулировка задачи:**

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;
* ID записи об отделе.

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.

**Код**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace \_3sem\_7lab

{

class Worker

{

/// <summary>

/// Ключ

/// </summary>

public int id;

/// <summary>

/// Фамилия

/// </summary>

public string sername;

/// <summary>

/// Класс данных

/// </summary>

public int idDeratment;

public Worker(int i, string sn, int d)

{

this.id = i;

this.sername = sn;

this.idDeratment = d;

}

public override string ToString()

{

return "ID: " + this.id + "; Фамилия: " + this.sername + "; ID\_Отдела: " + this.idDeratment;

}

}

class Department

{

/// <summary>

/// Ключ

/// </summary>

public int id;

/// <summary>

/// Фамилия

/// </summary>

public string name;

public Department(int i, string n)

{

this.id = i;

this.name = n;

}

public override string ToString()

{

return "ID: " + this.id + "; Наименование отдела: " + this.name;

}

}

class DepartmentWorker

{

public int idWorker;

public int idDepartment;

public DepartmentWorker(int iW, int iD)

{

this.idWorker = iW;

this.idDepartment = iD;

}

}

class MainClass

{

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

{

new Worker(1, "Белов", 1),

new Worker(2, "Кучеренко", 2),

new Worker(3, "Тимаков", 2),

new Worker(4, "Лескина", 2),

new Worker(5, "Брысина", 3),

new Worker(6, "Тюлькина", 1),

new Worker(7, "Байбарин", 3)

};

static List<Department> departments = new List<Department>()

{

new Department(1, "Отдел продаж"),

new Department(2, "Экномический отдел"),

new Department(3, "Юридический отдел")

};

static List<DepartmentWorker> departmentWorkers = new List<DepartmentWorker>

{

new DepartmentWorker(1,1),

new DepartmentWorker(1,2),

new DepartmentWorker(1,3),

new DepartmentWorker(2,1),

new DepartmentWorker(3,1),

new DepartmentWorker(3,3),

new DepartmentWorker(4,3),

new DepartmentWorker(5,2),

new DepartmentWorker(6,1),

new DepartmentWorker(7,2),

new DepartmentWorker(7,3)

};

public static void Main(string[] args)

{

foreach (var d in departments)

{

var q1 = from x in workers

where (d.id == x.idDeratment)

select x;

Console.WriteLine(d);

foreach (var x in q1) Console.WriteLine(x);

}

/\*var q1 = from x in workers

orderby x.idDeratment, x.id

select x;

foreach (var x in q1) Console.WriteLine(x);\*/

Console.WriteLine("Все сотрудники, у которых фамилия начинается на Б:");

var q2 = from x in workers

where (x.sername.Substring(0, 1) == "Б")

select x;

foreach (var x in q2) Console.WriteLine(x);

Console.WriteLine("Количество сотрудников в каждом из отделов:");

foreach (var x in departments)

{

int num = workers.Count(y => y.idDeratment == x.id);

Console.WriteLine(x + ": " + num);

}

Console.WriteLine("Отделы, в которых у всех сотрудников фамилия начинается на Б:");

var q3 = from x in departments

where (workers.Count(y => y.sername.Substring(0, 1) == "Б" && y.idDeratment == x.id) == workers.Count(y => y.idDeratment == x.id))

select x;

foreach (var x in q3) Console.WriteLine(x);

Console.WriteLine("Отделы, в которых хотя бы у одного сотрудника фамилия начинается на Б:");

var q4 = from x in departments

where (workers.Count(y => y.sername.Substring(0, 1) == "Б" && y.idDeratment == x.id) > 0)

select x;

foreach (var x in q4) Console.WriteLine(x);

foreach(var x in departments)

{

var q5 = from y in departmentWorkers

where (y.idDepartment == x.id)

select y;

var q6 = from y in workers

from z in q5

where (z.idWorker == y.id)

select y;

Console.WriteLine(x);

foreach (var y in q6) Console.WriteLine(y);

}

foreach (var x in departments)

{

var q5 = from y in departmentWorkers

where (y.idDepartment == x.id)

select y;

Console.WriteLine(x + ": " + q5.Count());

}

}

}

}