**Министерство науки и высшего образования Российской** **Федерации** **Федеральное государственное бюджетное образовательное учреждение высшего образования** **«Московский государственный технический университет** **имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Радиотехнический»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Домашнее задание

по дисциплине «Базовые компоненты интернет-технологий» на тему:

«Работа с LINQ to Objects»

Выполнил:

студент группы РТ5-31Б

В. И. Плешаков

Проверил:

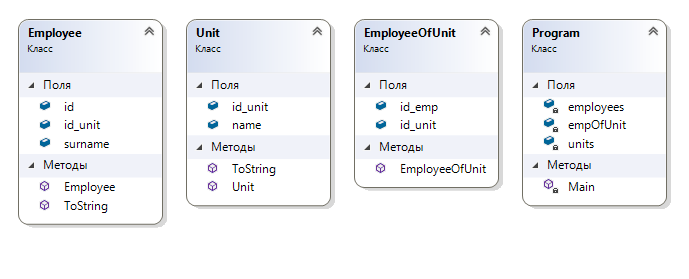
Ю. Е. Гапанюк

2020 г.

Задание:

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

Диаграмма классов:



Текст программы:

using System;

using System.Collections.Generic;

using System.Linq;

namespace DZ

{

/// <summary>

/// Класс Сотрудник

/// </summary>

public class Employee

{

/// <summary>

/// ID сотрудника

/// </summary>

public int id;

/// <summary>

/// Фамилия сотрудника

/// </summary>

public string surname;

/// <summary>

/// ID записи об отделе

/// </summary>

public int id\_unit;

/// <summary>

/// Конструктор класса

/// </summary>

public Employee(int i, string sn, int iu)

{

id = i;

surname = sn;

id\_unit = iu;

}

/// <summary>

/// Приведение к строке

/// </summary>

public override string ToString()

{

return "id = " + id.ToString() + "; surname = " + surname + "; id\_unit = " + id\_unit.ToString();

}

}

/// <summary>

/// Класс Отдел

/// </summary>

public class Unit

{

/// <summary>

/// ID записи об отделе

/// </summary>

public int id\_unit;

/// <summary>

/// Название отдела

/// </summary>

public string name;

/// <summary>

/// Конструктор класса

/// </summary>

public Unit(int i, string n)

{

id\_unit = i;

name = n;

}

/// <summary>

/// Приведение к строке

/// </summary>

public override string ToString()

{

return "id\_unit = " + id\_unit.ToString() + "; name = " + name;

}

}

/// <summary>

/// Класс Сотрудник отдела

/// </summary>

public class EmployeeOfUnit

{

/// <summary>

/// ID записи о сотруднике

/// </summary>

public int id\_emp;

/// <summary>

/// ID записи об отделе

/// </summary>

public int id\_unit;

/// <summary>

/// Конструктор класса

/// </summary>

public EmployeeOfUnit(int ie, int iu)

{

id\_emp = ie;

id\_unit = iu;

}

}

class Program

{

//Данные по сотрудникам

static List<Employee> employees = new List<Employee>()

{

new Employee(5, "Иванов", 1),

new Employee(6, "Аводков", 2),

new Employee(2, "Путин", 2),

new Employee(4, "Королев", 1),

new Employee(1, "Антонов", 3),

new Employee(3, "Афанасьев", 2)

};

//Данные по отделам

static List<Unit> units = new List<Unit>()

{

new Unit(1, "Продажи"),

new Unit(3, "Разработка"),

new Unit(2, "Маркетинг")

};

//Данные по сотрудникам отдела

static List<EmployeeOfUnit> empOfUnit = new List<EmployeeOfUnit>()

{

new EmployeeOfUnit(1,3),

new EmployeeOfUnit(1,2),

new EmployeeOfUnit(2,3),

new EmployeeOfUnit(2,1),

new EmployeeOfUnit(3,2),

new EmployeeOfUnit(4,1),

new EmployeeOfUnit(4,2),

new EmployeeOfUnit(5,1),

new EmployeeOfUnit(5,2),

new EmployeeOfUnit(5,3),

new EmployeeOfUnit(6,2),

new EmployeeOfUnit(6,1)

};

static void Main(string[] args)

{

Console.WriteLine("Сотрудники и отделы, отсортированные по отделам:");

var q1 = from x in employees

orderby x.id\_unit

select x;

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

var q2 = from x in units

orderby x.id\_unit

select x;

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

Console.WriteLine("\nСотрудники с фамилией на А:");

var q3 = from x in employees

where x.surname.ToUpper().StartsWith("А")

select x;

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

Console.WriteLine("\nКоличество сотрудников в каждом отделе:");

var q4 = from x in units

select x;

foreach (var x in q4)

Console.WriteLine(x + "\nКоличество сотрудников: " + employees.Count(y => x.id\_unit == y.id\_unit));

Console.WriteLine("\nОтделы, в которых у всех сотрудников фамилия начинается на А:");

var q5 = from x in units

join y in employees on x.id\_unit equals y.id\_unit into temp

where temp.All(p => p.surname.ToUpper().StartsWith("А"))

select x;

foreach (var x in q5) Console.WriteLine(x);

Console.WriteLine("\nОтделы, в которых хотя бы у одного сотрудника фамилия начинается на А:");

var q6 = from x in units

join y in employees on x.id\_unit equals y.id\_unit into temp

where temp.Any(p => p.surname.ToUpper().StartsWith("А"))

select x;

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

Console.WriteLine("\nОтделы и сотрудники в каждом отделе:");

var q7 = from x in units

join y1 in empOfUnit on x.id\_unit equals y1.id\_unit into temp

from t1 in temp

join y2 in employees on t1.id\_emp equals y2.id into temp2

from t2 in temp2

select new { id\_отдела = x.id\_unit, id\_сотрудника = t2.id };

foreach (var x in q7) Console.WriteLine(x);

Console.WriteLine("\nОтделы и количество сотрудников в каждом отделе:");

var q8 = from x in units

let temp1 = from eu in empOfUnit where eu.id\_unit == x.id\_unit select eu

from t1 in temp1

let temp2 = from y in employees

where y.id == t1.id\_emp

select y

select new { id\_отдела = x.id\_unit, количество\_сотрудников = temp1.Count() };

var q9 = q8.Distinct();

foreach (var x in q9) Console.WriteLine(x);

Console.ReadLine();

}

}

}

Пример выполнения программы:

