**Лабораторная работа № 7 по курсу**

**“Базовые компоненты интернет-технологий”**

*Лазарев Станислав Алексеевич*

*РТ5-31*

*МГТУ им. Баумана*

**Описание задания лабораторной работы.**

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

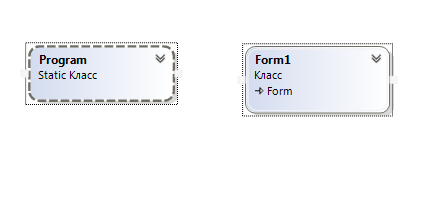
* + 1. Программа должна быть разработана в виде консольного приложения на языке C#.
  + 2. Создайте класс «Сотрудник», содержащий поля:
  +  ID записи о сотруднике;
  +  Фамилия сотрудника;
  +  ID записи об отделе.
  + 3. Создайте класс «Отдел», содержащий поля:
  +  ID записи об отделе;
  +  Наименование 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 программы:**

using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Text.RegularExpressions;  
  
namespace lab7  
{  
    class Program  
    {  
        public class Worker  
        {  
            public int id;  
            public string name;  
            public int unit\_id;  
  
            public Worker(int i, string n, int u\_id)  
            {  
                this.id = i;  
                this.name = n;  
                this.unit\_id = u\_id;  
            }  
            public override string ToString()  
            {  
                return "(id=" + this.id.ToString() + "; name=" + this.name + "; place id=" + this.unit\_id + ")";  
            }  
        }  
        public class Unit  
        {  
            public int id;  
            public string name;  
  
            public Unit(int i, string n)  
            {  
                this.id = i;  
                this.name = n;  
            }  
            public override string ToString()  
            {  
                return "(id=" + this.id.ToString() + "; name=" + this.name + ")";  
            }  
        }  
        public class WorkersOfUnit  
        {  
            public int worker;  
            public int unit;  
            public WorkersOfUnit(int i1, int i2)  
            {  
                this.worker = i1;  
                this.unit = i2;  
            }  
        }  
        static List<Worker> worker = new List<Worker>(){  
            new Worker(1,"Oleg",1),  
            new Worker(2,"Kolya",2),  
            new Worker(3,"BOrya",3),  
            new Worker(4,"Pasha",2),  
            new Worker(5,"Savveliy",2),  
            new Worker(6,"Rokesh",4),  
            new Worker(7,"Alin",6),  
            new Worker(8,"Alex",6),  
            new Worker(9,"Hackerman",3),  
            new Worker(10,"Inga",4),  
            new Worker(11,"Gorshok",2),  
            new Worker(12,"Ira",4),  
            new Worker(13,"GUDBOI",4),  
            new Worker(14,"Kolbasa",5),  
            new Worker(15,"Bono",3),  
            new Worker(16,"Mono",5),  
        };  
        static List<Unit> unit = new List<Unit>(){  
            new Unit(1,"CEO"),  
            new Unit(2,"SMM"),  
            new Unit(3,"Dev"),  
            new Unit(4,"Support"),  
            new Unit(5,"PR"),  
            new Unit(6,"HR"),  
        };  
        static List<WorkersOfUnit> wou = new List<WorkersOfUnit>()  
        {  
            new WorkersOfUnit(1,1),  
            new WorkersOfUnit(2,2),  
            new WorkersOfUnit(3,3),  
            new WorkersOfUnit(4,4),  
            new WorkersOfUnit(5,5),  
            new WorkersOfUnit(6,6),  
        };  
  
        static void Main(string[] args)  
        {  
            Console.Title = "База сотрудников";  
            Console.WriteLine("Список всех сотрудников и отделов, отсортированный по отделам:\n");  
            var q1 = from x in unit  
                     join y in worker on x.id equals y.unit\_id into temp  
                     select new { id = x.id, name = x.name, d2Group = temp };  
            foreach (var x in q1)  
            {  
                Console.WriteLine(x.id + " " + x.name);  
                foreach (var y in x.d2Group)  
                    Console.WriteLine("   " + y);  
            }  
            Regex regex = new Regex("A");  
            Console.WriteLine("\nCписок всех сотрудников, у которых фамилия начинается на букву A \n");  
            var q2 = from x in worker  
                     where regex.IsMatch(x.name)  
                     select x;  
            foreach (var x in q2) Console.WriteLine(x);  
            Console.WriteLine("\nВыведите список всех отделов и количество сотрудников в каждом отделе. \n");  
            var q3 = from x in unit  
                     select new { uid = x.id, uname = x.name, ucount = worker.Count(z => z.unit\_id == x.id) };  
            foreach (var x in q3)  
                Console.WriteLine(x);  
            Console.WriteLine("\nВыведите список отделов, в которых у всех сотрудников фамилия начинается с буквы «A» \n");  
            var q4 = worker.GroupBy(x => x.unit\_id);  
            foreach (var x in q4.Where(z => z.All(p => regex.IsMatch(p.name))))  
                Console.WriteLine("{0}", x.Key);   
            Console.WriteLine("Отделы, в которых хотя бы у одного сотрудника фамилия начинается с буквы 'A'");  
            foreach (var x in q4.Where(z => z.Any(p => regex.IsMatch(p.name))))  
                Console.WriteLine("{0}", x.Key);  
            Console.WriteLine("\nСписок всех отделов и список сотрудников в каждом отделе. \n");  
            var wou1 = from x in worker  
                       join l in wou on x.id equals l.worker into temp  
                       from t1 in temp  
                       join y in unit on t1.unit equals y.id into temp2  
                       from t2 in temp2  
                        select new {unit = t2.name, name = x.name};  
            foreach (var x in wou1) Console.WriteLine(x);  
            Console.WriteLine("\nСписок всех отделов и количество сотрудников в каждом отделе. \n");  
            var wou2 = from x in worker  
                       join l in wou on x.id equals l.worker into temp  
                       from t1 in temp  
                       join y in unit on t1.unit equals y.id into temp2  
                       from t2 in temp2  
                                         select new { unit = t2.name, ucount = worker.Count(z => z.unit\_id == x.id) };  
            foreach (var x in wou2) Console.WriteLine(x);  
            Console.WriteLine("\nPress any key to exit");  
            Console.ReadKey();  
        }  
    }  
}

}

**Диаграмма классов:**



**Пример консольного вывода:**

