|  |  |
| --- | --- |
|  | **Министерство образования и науки 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 **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7**

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Москальцов Никита Олегович \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*фамилия, имя, отчество*

Группа: ИУ5 - 34

Название предмета**: Базовые компоненты интернет-технологий**

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** \_\_\_\_Москальцов Н.О.\_\_\_\_

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_**Гапанюк Ю.Е.**\_\_\_\_**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2017 г.*

**Задание**

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:

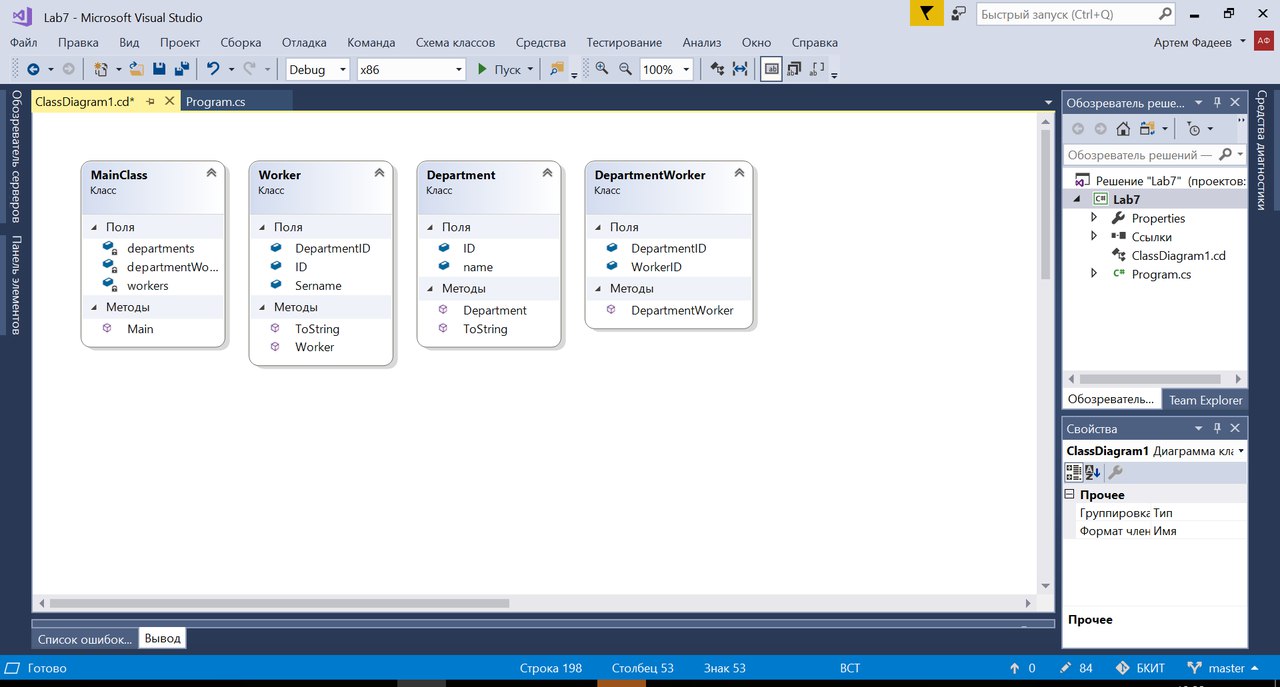
Выведите список всех отделов и список сотрудников в каждом отделе.

Выведите список всех отделов и количество сотрудников в каждом отделе.

**Код программы**

|  |
| --- |
| using System; |
|  | using System.Collections.Generic; |
|  | using System.Linq; |
|  | using System.Text; |
|  |  |
|  | namespace Lab7 |
|  | { |
|  | class Worker |
|  | { |
|  | // Ключ |
|  | public int ID; |
|  |  |
|  | // Фамилия |
|  | public string Sername; |
|  |  |
|  | // Номер отдела |
|  | public int DepartmentID; |
|  |  |
|  | //Конструктор сотрудника |
|  | public Worker(int i, string sn, int d) |
|  | { |
|  | this.ID = i; |
|  | this.Sername = sn; |
|  | this.DepartmentID = d; |
|  | } |
|  |  |
|  | //Перегрузка стандартного приведение к строке |
|  | public override string ToString() |
|  | { |
|  | return "ID: " + this.ID + "; Фамилия: " + this.Sername + |
|  | "; ID\_Отдела: " + this.DepartmentID; |
|  | } |
|  | } |
|  |  |
|  | class Department |
|  | { |
|  | //Номер отдела |
|  | public int ID; |
|  |  |
|  | // Навзание |
|  | public string name; |
|  |  |
|  | //Конструктор отдела |
|  | 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 WorkerID; |
|  |  |
|  | public int DepartmentID; |
|  |  |
|  | public DepartmentWorker(int iW, int iD) |
|  | { |
|  | this.WorkerID = iW; |
|  | this.DepartmentID = iD; |
|  | } |
|  | } |
|  |  |
|  |  |
|  | class MainClass |
|  | { |
|  | //Создаем класс со списками по сотрудникам, по отделам, и по сотруднико-отделам |
|  | static List<Worker> workers = new List<Worker>() |
|  | { |
|  | new Worker(1, "Махмудов", 1), |
|  | new Worker(2, "Петров", 2), |
|  | new Worker(3, "Кучеренко", 2), |
|  | new Worker(4, "Брысина", 3), |
|  | new Worker(5, "Арифулин", 2), |
|  | new Worker(6, "Прудниченков", 1), |
|  | new Worker(7, "Фадеев", 3) |
|  | }; |
|  |  |
|  | static List<Department> departments = new List<Department>() |
|  | { |
|  | new Department(1, "Отдел продаж"), |
|  | new Department(2, "Отдел закупок"), |
|  | new Department(3, "Отдел кадров") |
|  | }; |
|  |  |
|  | 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) |
|  | { |
|  | //Перебор по каждому элементу отделов |
|  | foreach (var d in departments) |
|  | { |
|  | //Перебор (укороченный синтаксис цикла) по каждому сотруднику |
|  | var q1 = from x in workers |
|  | where (d.ID == x.DepartmentID) |
|  | select x; |
|  | //Проверка на соответствие сотрудника его отделу (вывод типа: отдел - сотрудники) |
|  | Console.WriteLine(d); |
|  | foreach (var x in q1) Console.WriteLine(x); |
|  | } |
|  | Console.WriteLine("\n"); |
|  |  |
|  |  |
|  | //Сотрудники по первой букве фамилии |
|  | Console.WriteLine("Введите первую букву фамилии своих сотрудников: "); |
|  | string lit = Console.ReadLine(); |
|  | Console.WriteLine("Все сотрудники, у которых фамилия начинается на " + lit + ":"); |
|  | var q2 = from x in workers |
|  | where (x.Sername.Substring(0, 1) == lit) |
|  | select x; |
|  | foreach (var x in q2) Console.WriteLine(x); |
|  | if (q2.Count() == 0) |
|  | { |
|  | Console.WriteLine("Ни в одном отдел не отвечает данному требованию:("); |
|  | } |
|  | Console.WriteLine("\n"); |
|  |  |
|  |  |
|  | //Количество сотрудников по отделам (через лямбда-выражение) |
|  | Console.WriteLine("Количество сотрудников в каждом из отделов:"); |
|  | foreach (var x in departments) |
|  | { |
|  | int num = workers.Count(y => y.DepartmentID == x.ID); |
|  | Console.WriteLine(x + ": " + num); |
|  | } |
|  | Console.WriteLine("\n"); |
|  |  |
|  |  |
|  | //Печать отчетов по различным запросам по началу фамилий у сотрудников |
|  | Console.WriteLine("Отделы, в которых у всех сотрудников фамилия начинается на П:"); |
|  | var q3 = from x in departments |
|  | where (workers.Count(y => |
|  | y.Sername.Substring(0, 1) == "П" && y.DepartmentID == x.ID) == workers.Count(y => |
|  | y.DepartmentID == x.ID)) |
|  | select x; |
|  | foreach (var x in q3) Console.WriteLine(x); |
|  | if (q3.Count() == 0) |
|  | { |
|  | Console.WriteLine("Ни в одном отдел не отвечает данному требованию:("); |
|  | } |
|  | Console.WriteLine("\n"); |
|  |  |
|  |  |
|  | 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.DepartmentID == x.ID) > 0) |
|  | select x; |
|  | foreach (var x in q4) Console.WriteLine(x); |
|  | Console.WriteLine("\n"); |
|  |  |
|  |  |
|  | //Вывод на печать номера отдела - соответсвующий ему список сотрудников |
|  |  |
|  | //Перебор по отделам |
|  | foreach (var x in departments) |
|  | { |
|  | //Перебор по отделам-сотрудникам |
|  | var q5 = from y in departmentWorkers |
|  | //сравнение номера отдела сотрудника с текущим (по проходу) отделом |
|  | where (y.DepartmentID == x.ID) |
|  | //запоминание его в список й5 |
|  | select y; |
|  |  |
|  | //Перебор по списку сотрудников |
|  | var q6 = from y in workers |
|  | //Перебор по списку q5 |
|  | from z in q5 |
|  | //Сравнение номера сотрудника отдела с текущим работником |
|  | where (z.WorkerID == y.ID) |
|  | //Запоминание его в список й6 |
|  | select y; |
|  | Console.WriteLine(x); |
|  | foreach (var y in q6) Console.WriteLine(y); |
|  | } |
|  | Console.WriteLine("\n"); |
|  |  |
|  | Console.WriteLine("Вывод списка отделов (из промежуточной сущности)" + |
|  | " с количеством сотрудников в них"); |
|  | //Вывод списка количества сотрудников по всем отделам |
|  | //Перебор по отделам |
|  | foreach (var x in departments) |
|  | { |
|  | //Перебор по отделам работников |
|  | var q5 = from y in departmentWorkers |
|  | //Сравнение номера отдела работника с текущим отделом |
|  | where (y.DepartmentID == x.ID) |
|  | select y; |
|  | Console.WriteLine(x + ": " + q5.Count()); |
|  | } |
|  |  |
|  | Console.WriteLine("\n"); |
|  | Console.WriteLine("Нажмите на любую клавишу..."); |
|  | Console.Read(); |
|  | } |
|  | } |
|  | } |

**Диаграмма классов**



**Примеры работы**

