**Московский государственный технический университет им. Н.Э. Баумана**

**Базовые компоненты интернет технологий**

Отчет по лабораторной работе №7

Гаранин Антон Викторович

ИУ5Ц-51Б

25 декабря 2018г.

Москва

2018

**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:**

DepartmentEmployees.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LR7

{

class DepartmentEmployees

{

public int EmployeeID;

public int DepartmentID;

public DepartmentEmployees(int employeeId, int departmentId)

{

EmployeeID = employeeId;

DepartmentID = departmentId;

}

public override string ToString()

{

return string.Format("{{ DepartmentEmployees EmployeeID: {0}, DepartmentID: {1} }}", EmployeeID, DepartmentID);

}

}

}

Employee.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LR7

{

public class Employee

{

public int ID;

public string Surname;

public int DepartmentID;

public Employee(int id, string surname, int departmentId)

{

ID = id;

Surname = surname;

DepartmentID = departmentId;

}

public override string ToString()

{

return string.Format("{{ Employee ID: {0}, Surname: {1}, DepartmentID: {2} }}", ID, Surname, DepartmentID);

}

}

}

Department.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LR7

{

public class Department

{

public int ID;

public string Name;

public Department(int id, string name)

{

ID = id;

Name = name;

}

public override string ToString()

{

return string.Format("{{ Department ID: {0}, Name: {1} }}", ID, Name);

}

}

}

Program.cs

using System;

using System.Collections.Generic;

using System.ComponentModel.Design;

using System.Linq;

namespace LR7

{

class Program

{

static void Main(string[] args)

{

var departments = new List<Department>

{

new Department(1, "Бухгалтерия"),

new Department(2, "Маркетинговый отдел"),

new Department(3, "Финансовый отдел"),

new Department(4, "Отдел управления поставками"),

new Department(5, "Отдел технической поддержки"),

new Department(6, "Отдел разработки программного обеспечения"),

new Department(7, "Отдел тестирования")

};

var employees = new List<Employee>

{

new Employee(1, "Тураев", 6),

new Employee(2, "Гаранин", 6),

new Employee(3, "Ролдугин", 7),

new Employee(4, "Климанов", 5),

new Employee(5, "Жильцов", 1),

new Employee(6, "Фонканц", 3),

new Employee(7, "Болгова", 1),

new Employee(8, "Журавлёва", 1),

new Employee(9, "Евдокимов", 4),

new Employee(10, "Нефёдова", 3),

new Employee(11, "Щепицына", 7),

new Employee(12, "Спицын", 5),

new Employee(13, "Ерохин", 2),

new Employee(14, "Никольский", 2)

};

Console.WriteLine("Выведите список всех сотрудников и отделов, отсортированный по отделам");

var list1 =

from employee in employees

join department in departments on employee.DepartmentID equals department.ID

orderby department.Name

select new

{

employee.Surname,

DepartmentName = department.Name

};

foreach (var item in list1)

{

Console.WriteLine(item);

}

Console.WriteLine();

Console.WriteLine("Выведите список всех сотрудников, у которых фамилия начинается с буквы «А»");

var list2 =

from employee in employees

where employee.Surname[0] == 'А'

select employee;

foreach (var item in list2)

{

Console.WriteLine(item);

}

Console.WriteLine();

Console.WriteLine("Выведите список всех отделов и количество сотрудников в каждом отделе");

var list3 =

from department in departments

join employee in employees on department.ID equals employee.DepartmentID into employeesOnDepartment

select new

{

DepartmentID = department.ID,

DepartmentName = department.Name,

CountOfEmployees = employeesOnDepartment.Count()

};

foreach (var item in list3)

{

Console.WriteLine(item);

}

Console.WriteLine();

Console.WriteLine("Выведите список отделов, в которых у всех сотрудников фамилия начинается с буквы «А»");

var list4 =

from department in departments

join employee in employees on department.ID equals employee.DepartmentID into employeesOnDepartment

where employeesOnDepartment.All(employee => employee.Surname[0] == 'А')

select department;

foreach (var item in list4)

{

Console.WriteLine(item);

}

Console.WriteLine();

Console.WriteLine("Выведите список отделов, " +

"в которых хотя бы у одного сотрудника фамилия начинается с буквы «А»");

var list5 =

from department in departments

join employee in employees on department.ID equals employee.DepartmentID into employeesOnDepartment

where employeesOnDepartment.Any(employee => employee.Surname[0] == 'А')

select department;

foreach (var item in list5)

{

Console.WriteLine(item);

}

Console.WriteLine();

var departmentEmployees = new List<DepartmentEmployees>

{

new DepartmentEmployees(5, 3),

new DepartmentEmployees(5, 1),

new DepartmentEmployees(5, 3),

new DepartmentEmployees(7, 5),

new DepartmentEmployees(3, 2),

new DepartmentEmployees(6, 7),

new DepartmentEmployees(6, 7),

new DepartmentEmployees(6, 7),

new DepartmentEmployees(7, 5),

new DepartmentEmployees(7, 2),

new DepartmentEmployees(5, 1),

new DepartmentEmployees(4, 6),

new DepartmentEmployees(4, 5),

new DepartmentEmployees(1, 4),

new DepartmentEmployees(2, 3),

new DepartmentEmployees(3, 3),

new DepartmentEmployees(11, 2),

new DepartmentEmployees(14, 5),

new DepartmentEmployees(12, 4),

new DepartmentEmployees(4, 7),

new DepartmentEmployees(3, 5),

new DepartmentEmployees(9, 2),

new DepartmentEmployees(9, 4),

new DepartmentEmployees(10, 6),

new DepartmentEmployees(12, 4),

new DepartmentEmployees(7, 2),

new DepartmentEmployees(1, 1),

new DepartmentEmployees(2, 1),

new DepartmentEmployees(12, 5)

};

Console.WriteLine("Выведите список всех отделов и список сотрудников в каждом отделе");

var list6 =

from departmentEmployee in departmentEmployees

group departmentEmployees by departmentEmployee.DepartmentID into deps

select new

{

Department = departments.Single(dep => dep.ID == deps.Key),

Employees = string.Join(", ", employees.FindAll(employee => employee.DepartmentID == deps.Key))

};

foreach (var item in list6)

{

Console.WriteLine(item);

}

Console.WriteLine();

Console.WriteLine("Выведите список всех отделов и количество сотрудников в каждом отделе");

var list7 =

from departmentEmployee in departmentEmployees

group departmentEmployees by departmentEmployee.DepartmentID into deps

select new

{

Department = departments.Single(dep => dep.ID == deps.Key),

EmployeesCount = employees.FindAll(employee => employee.DepartmentID == deps.Key).Count

};

foreach (var item in list7)

{

Console.WriteLine(item);

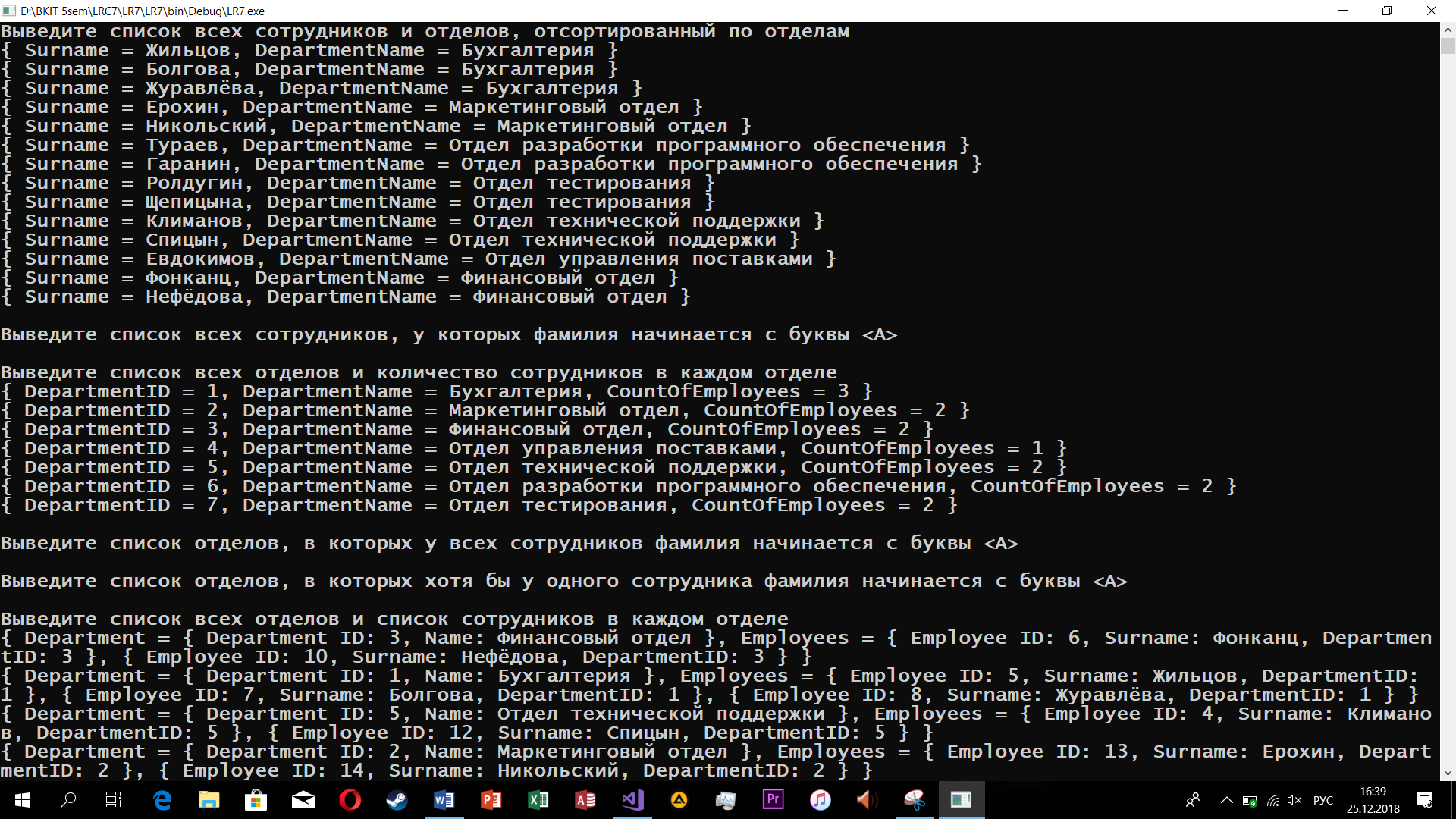
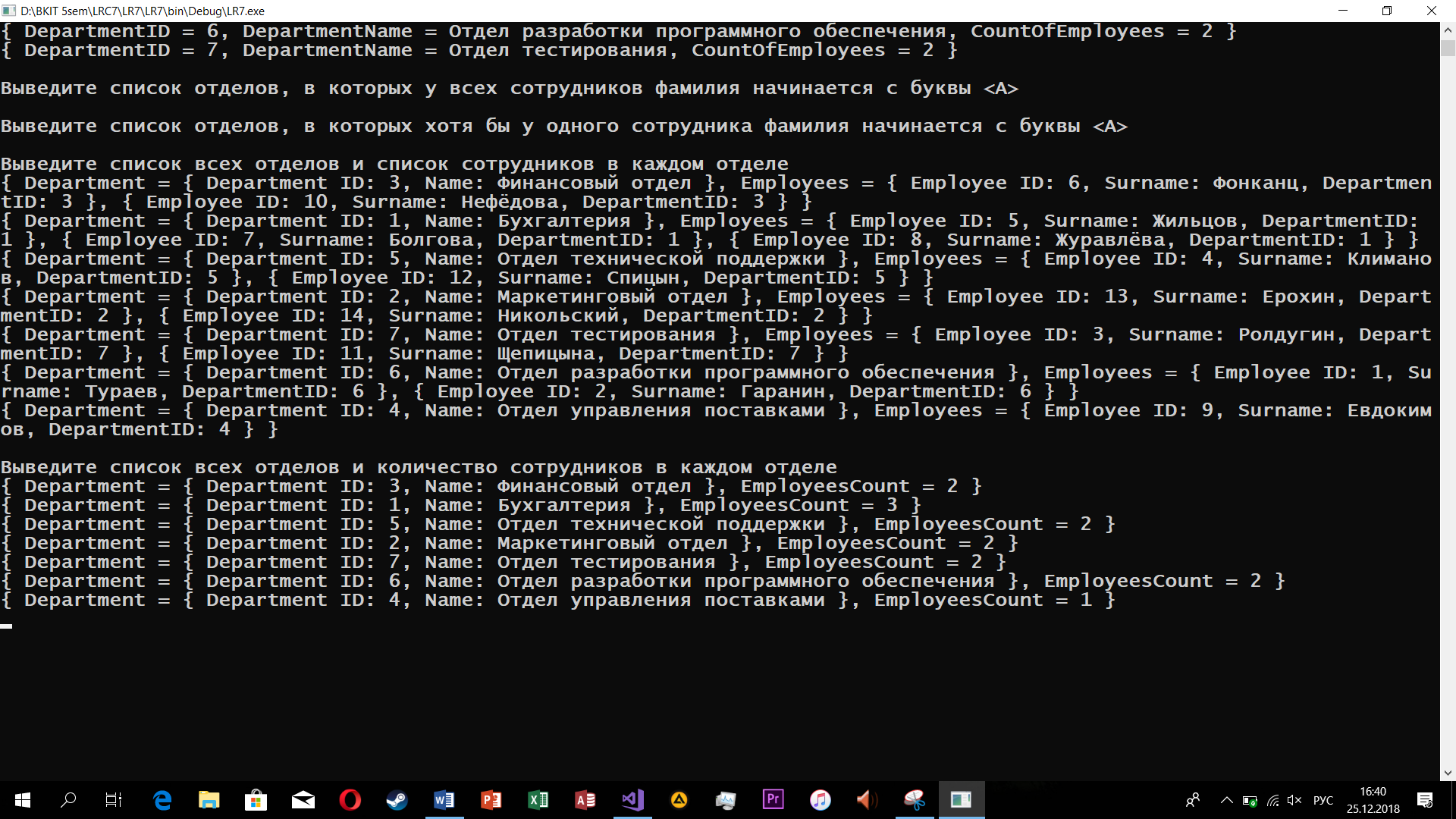
}

Console.ReadLine();

}

}

}

ТЕСТ: