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 одинко-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.

using System;

using System.Collections.Generic;

using System.Linq;

// Класс Сотрудник

class Employee

{

public int ID { get; set; }

public string LastName { get; set; }

public int DepartmentID { get; set; }

}

// Класс Отдел

class Department

{

public int ID { get; set; }

public string Name { get; set; }

}

// Класс СотрудникиОтдела

class EmployeesInDepartment

{

public int EmployeeID { get; set; }

public int DepartmentID { get; set; }

}

class Program

{

static void Main()

{

// Создание коллекций объектов

List<Employee> employees = new List<Employee>

{

new Employee { ID = 1, LastName = "Ivanov", DepartmentID = 1 },

new Employee { ID = 2, LastName = "Alexeev", DepartmentID = 2 },

new Employee { ID = 3, LastName = "Abramov", DepartmentID = 1 },

new Employee { ID = 4, LastName = "Antonov", DepartmentID = 2 },

};

List<Department> departments = new List<Department>

{

new Department { ID = 1, Name = "IT Department" },

new Department { ID = 2, Name = "HR Department" },

};

List<EmployeesInDepartment> employeesInDepartments = new List<EmployeesInDepartment>

{

new EmployeesInDepartment { EmployeeID = 1, DepartmentID = 1 },

new EmployeesInDepartment { EmployeeID = 2, DepartmentID = 2 },

new EmployeesInDepartment { EmployeeID = 3, DepartmentID = 1 },

new EmployeesInDepartment { EmployeeID = 4, DepartmentID = 2 },

};

// Запросы с использованием LINQ to Objects

Console.WriteLine("Список всех сотрудников и отделов, отсортированный по отделам:");

var query1 = from emp in employees

join dep in departments on emp.DepartmentID equals dep.ID

orderby dep.Name

select new { Employee = emp, Department = dep };

foreach (var item in query1)

{

Console.WriteLine($"ID: {item.Employee.ID}, Фамилия: {item.Employee.LastName}, Отдел: {item.Department.Name}");

}

Console.WriteLine("\nСписок всех сотрудников, у которых фамилия начинается с буквы 'А':");

var query2 = from emp in employees

where emp.LastName.StartsWith("A", StringComparison.OrdinalIgnoreCase)

select emp;

foreach (var emp in query2)

{

Console.WriteLine($"ID: {emp.ID}, Фамилия: {emp.LastName}");

}

Console.WriteLine("\nСписок всех отделов и количество сотрудников в каждом отделе:");

var query3 = from dep in departments

join empDep in employeesInDepartments on dep.ID equals empDep.DepartmentID into empGroup

select new { Department = dep, EmployeeCount = empGroup.Count() };

foreach (var item in query3)

{

Console.WriteLine($"Отдел: {item.Department.Name}, Количество сотрудников: {item.EmployeeCount}");

}

Console.WriteLine("\nСписок отделов, в которых у всех сотрудников фамилия начинается с буквы 'А':");

var query4 = from dep in departments

where employees.All(emp => emp.DepartmentID == dep.ID && emp.LastName.StartsWith("A", StringComparison.OrdinalIgnoreCase))

select dep;

foreach (var dep in query4)

{

Console.WriteLine($"Отдел: {dep.Name}");

}

Console.WriteLine("\nСписок отделов, в которых хотя бы у одного сотрудника фамилия начинается с буквы 'А':");

var query5 = from dep in departments

where employees.Any(emp => emp.DepartmentID == dep.ID && emp.LastName.StartsWith("A", StringComparison.OrdinalIgnoreCase))

select dep;

foreach (var dep in query5)

{

Console.WriteLine($"Отдел: {dep.Name}");

}

// Многие ко многим

Console.WriteLine("\nСписок всех отделов и список сотрудников в каждом отделе (много-ко-многим):");

var query6 = from dep in departments

join empDep in employeesInDepartments on dep.ID equals empDep.DepartmentID into empGroup

select new { Department = dep, Employees = empGroup.Join(employees, ed => ed.EmployeeID, e => e.ID, (ed, e) => e) };

foreach (var item in query6)

{

Console.WriteLine($"Отдел: {item.Department.Name}");

foreach (var emp in item.Employees)

{

Console.WriteLine($" Сотрудник: {emp.LastName}");

}

}

Console.WriteLine("\nСписок всех отделов и количество сотрудников в каждом отделе (много-ко-многим):");

var query7 = from dep in departments

join empDep in employeesInDepartments on dep.ID equals empDep.DepartmentID into empGroup

select new { Department = dep, EmployeeCount = empGroup.Count() };

foreach (var item in query7)

{

Console.WriteLine($"Отдел: {item.Department.Name}, Количество сотрудников: {item.EmployeeCount}");

}

}

}