**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №7

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-35б |  | преподаватель каф. ИУ5 |
| Хижняков Вадим |  |  |
| Подпись и дата: |  | Подпись и дата: |

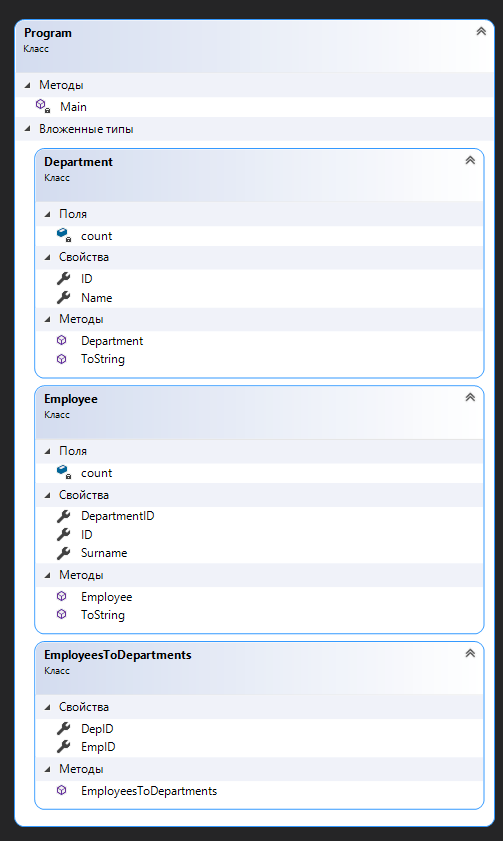
Москва, 2020 г.

# 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.

# Диаграмма классов



# Текст программы

using System;

using System.Collections.Generic;

using System.Linq;

namespace Sem3Lab7\_LINQ

{

class Program

{

class Employee

{

private static ulong count = 0;

public ulong ID { get; private set; }

public string Surname { get; set; }

public ulong DepartmentID { get; set; }

public Employee(string surname, Department department)

{

Surname = surname;

DepartmentID = department.ID;

ID = ++count;

}

public override string ToString()

{

return $"{Surname}";

}

}

class Department

{

private static ulong count = 0;

public ulong ID { get; private set; }

public string Name { get; set; }

public Department(string name)

{

Name = name;

ID = ++count;

}

public override string ToString()

{

return $"{Name}";

}

}

class EmployeesToDepartments

{

public ulong EmpID { get; set; }

public ulong DepID { get; set; }

public EmployeesToDepartments(ulong empID, ulong depID)

{

EmpID = empID;

DepID = depID;

}

}

static void Main(string[] args)

{

var SalesDepartment = new Department("Отдел продаж");

var HumanResourcesDepartment = new Department("Отдел кадров");

var AccountingDepartment = new Department("Бухгалтерия");

var ITDepartment = new Department("Отдел разработки");

var LegalDepartment = new Department("Юридический отдел");

var departments = new List<Department>(new Department[] { SalesDepartment, HumanResourcesDepartment, AccountingDepartment, ITDepartment, LegalDepartment });

var e1 = new Employee("AKhizhnyakov", ITDepartment);

var e2 = new Employee("Zhuk", ITDepartment);

var e3 = new Employee("Boyarko", AccountingDepartment);

var e4 = new Employee("Sedisheva", LegalDepartment);

var e5 = new Employee("Pokalo", ITDepartment);

var e6 = new Employee("Mamedova", SalesDepartment);

var e7 = new Employee("Astafyev", HumanResourcesDepartment);

var e8 = new Employee("Prohorov", ITDepartment);

var e9 = new Employee("Smyslov", ITDepartment);

var e10 = new Employee("Amrakhov", HumanResourcesDepartment);

var employees = new List<Employee>(new Employee[] { e1, e2, e3, e4, e5, e6, e7, e8, e9, e10 });

var res = employees.OrderBy(i => i.DepartmentID);

Console.WriteLine("\t-Cписок всех сотрудников и отделов, отсортированный по отделам-");

foreach (var item in res)

Console.WriteLine(departments.Where(i => i.ID == item.DepartmentID).First().ToString() + " - " + item.ToString());

Console.WriteLine("-----------------------------------------------------------------");

Console.WriteLine("\t-Cписок всех сотрудников, у которых фамилия начинается с буквы \"A\"-");

var res2 = employees.Where(i => i.Surname.StartsWith('A'));

foreach (var item in res2)

Console.WriteLine(item.ToString());

Console.WriteLine("-----------------------------------------------------------------");

Console.WriteLine("\t-Cписок всех отделов и количество сотрудников в каждом отделе-");

foreach (var item in departments)

Console.WriteLine(item.ToString() + ". Сотрудников: " + employees.Where(i => i.DepartmentID == item.ID).Count());

Console.WriteLine("-----------------------------------------------------------------");

Console.WriteLine("\t-Cписок всех сотрудников и отделов, отсортированный по отделам V.2-");

var res3 = departments.GroupJoin(

employees,

d => d.ID,

e => e.DepartmentID,

(dep, emps) => new

{

Name = dep.Name,

Employees = emps.Select(emp => emp.Surname)

});

foreach (var dep in res3)

{

Console.WriteLine("\t\*" + dep.Name + "\*");

foreach (var emp in dep.Employees)

{

Console.WriteLine(emp);

}

}

Console.WriteLine("-----------------------------------------------------------------");

Console.WriteLine("\t-Cписок отделов, в которых у всех сотрудников фамилия начинается с буквы \"А\"-");

var res4 = res3.Where(i => i.Employees.All(e => e.StartsWith('A')));

foreach (var dep in res4)

{

Console.WriteLine("\t\*" + dep.Name + "\*");

foreach (var emp in dep.Employees)

{

Console.WriteLine(emp);

}

}

Console.WriteLine("-----------------------------------------------------------------");

Console.WriteLine("\t-Cписок отделов, в которых хотя бы у одного сотрудника фамилия начинается с буквы \"А\"-");

var res5 = res3.Where(i => i.Employees.Any(e => e.StartsWith('A')));

foreach (var dep in res5)

{

Console.WriteLine("\t\*" + dep.Name + "\*");

foreach (var emp in dep.Employees)

{

Console.WriteLine(emp);

}

}

var EmpToDep = new List<EmployeesToDepartments>() {

new EmployeesToDepartments(e1.ID, ITDepartment.ID),

new EmployeesToDepartments(e1.ID, HumanResourcesDepartment.ID),

new EmployeesToDepartments(e2.ID, ITDepartment.ID),

new EmployeesToDepartments(e3.ID, AccountingDepartment.ID),

new EmployeesToDepartments(e3.ID, SalesDepartment.ID),

new EmployeesToDepartments(e4.ID, LegalDepartment.ID),

new EmployeesToDepartments(e5.ID, ITDepartment.ID),

new EmployeesToDepartments(e6.ID, SalesDepartment.ID),

new EmployeesToDepartments(e7.ID, HumanResourcesDepartment.ID),

new EmployeesToDepartments(e8.ID, ITDepartment.ID),

new EmployeesToDepartments(e8.ID, HumanResourcesDepartment.ID),

new EmployeesToDepartments(e9.ID, ITDepartment.ID),

new EmployeesToDepartments(e10.ID, HumanResourcesDepartment.ID),

new EmployeesToDepartments(e10.ID, LegalDepartment.ID)

};

var res6 = EmpToDep.Join(

employees,

etd => etd.EmpID,

emp => emp.ID,

(etd, emp) => new { DepID = etd.DepID, Surname = emp.Surname }

);

var res7 = departments.GroupJoin(

res6,

d => d.ID,

r6 => r6.DepID,

(dep, rs6) =>

new

{

DepName = dep.Name,

Employees = rs6.Select(i => i.Surname)

});

Console.WriteLine("===============================MANY TO MANY===============================");

Console.WriteLine("\t-Cписок всех отделов и список сотрудников в каждом отделе-");

foreach (var dep in res7)

{

Console.WriteLine("\t\t\* " + dep.DepName + " \*");

foreach (var emp in dep.Employees)

{

Console.WriteLine(emp);

}

}

Console.WriteLine("-----------------------------------------------------------");

Console.WriteLine("\t-Cписок всех отделов и количество сотрудников в каждом отделе-");

foreach (var dep in res7)

{

Console.WriteLine(dep.DepName + ": " + dep.Employees.Count());

}

}

}

}

# Результаты

