|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования 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 «Радиотехнический»**

**Кафедра «Системы обработки информации и управления»**

Домашнее задание

по дисциплине «Базовые компоненты интернет технологий»

Выполнил:

студент(ка) группы № РТ5-31Б

Пакало Александр Сергеевич

подпись, дата

Проверил:

преподаватель каф. ИУ5  
Гапанюк Юрий Евгеньевич

подпись, дата

2020 г.

Оглавление

[Описание задания. 3](#_Toc59478501)

[Диаграмма классов. 3](#_Toc59478502)

[Текст программы. 3](#_Toc59478503)

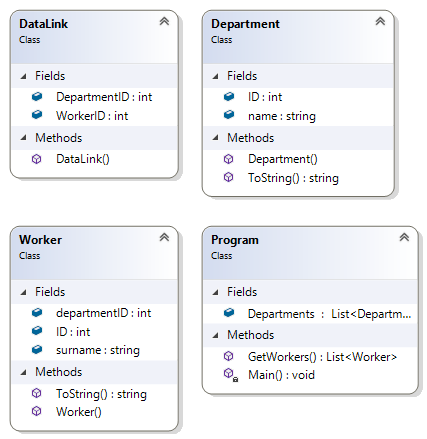
[Анализ результатов. 3](#_Toc59478504)

# Описание задания.

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.

## Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

namespace Lab\_7

{

// Department и Worker имеют связь 1 ко многим.

class Program

{

/// <summary>

/// Список отделов.

/// </summary>

public static List<Department> Departments = new List<Department>()

{

new Department(1, "Правительственный отдел"),

new Department(2, "Образовательный отдел"),

new Department(3, "Студенческий отдел"),

new Department(4, "Чилибасерский отдел")

};

/// <summary>

/// Метод, отвечающий за получение данных.

/// </summary>

/// <returns></returns>

public static List<Worker> GetWorkers()

{

/// <summary>

/// Данные (списки работников).

/// </summary>

var Workers = new List<Worker>()

{

new Worker(1, "Путин", 1),

new Worker(2, "Обама", 1),

new Worker(6, "Белодедов", 2),

new Worker(7, "Гапанюк", 2),

new Worker(3, "Трамп", 1),

new Worker(8, "Пакало", 3),

new Worker(9, "Грызин", 3),

new Worker(5, "Порошенко", 1),

new Worker(4, "Афанасьев", 2),

new Worker(10, "Семёнов", 2),

new Worker(11, "Демченков", 4),

new Worker(12, "Борисочкин", 3),

new Worker(13, "Незаметдинов", 4),

new Worker(14, "Ананасов", 4)

};

return Workers;

}

static void Main(string[] args)

{

// Инициализация.

var Workers = GetWorkers();

/// <summary>

/// База данных, образующая связь между Department и Worker.

/// </summary>

List<DataLink> dataBase = new List<DataLink>();

// Связываем работников с соответствующими отделами по ID.

foreach (Worker worker in Workers)

{

dataBase.Add(new DataLink(worker.departmentID, worker.ID));

}

var q1 = from w in Workers

orderby w.departmentID ascending

select w;

Console.WriteLine("Вывод списка всех сотрудников, отсортированный по отделам:");

foreach (var w in q1) Console.WriteLine(w);

var q2 = from w in Workers

where w.surname[0] == 'А'

select w;

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

foreach (var w in q2) Console.WriteLine(w);

var departmentsQ = from d in Departments

select d;

// Список связок отдел-количество работников.

var q3 = new List<Tuple<Department, int>>();

foreach (var d in departmentsQ)

{

// Делаем выборку работников каждого отдела для подсчёта их количества.

var tempQ = from dataLink in dataBase

where dataLink.DepartmentID == d.ID

select dataLink;

q3.Add(new Tuple<Department, int>(d, tempQ.Count()));

}

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

foreach (var d in q3) Console.WriteLine($"{d.Item1}, количество сотрудников: {d.Item2}");

/\*var q4 = from d in Departments

join w in Workers on d.ID equals w.departmentID // Inner join.

where w.

select d;\*/

var q4 = from w in Workers

group w by w.departmentID into d

where d.All(w => w.surname[0] == 'А')

select new { department = d.Key, workers = d };

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

foreach (var d in q4)

{

Console.WriteLine(d.department);

foreach (var w in d.workers)

{

Console.WriteLine(w);

}

}

var q5 = from d in Departments

join w in q2 on d.ID equals w.departmentID // Inner join. В q2 содержатся сотрудники с фамилией на 'А'.

select d;

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

foreach (var d in q5) Console.WriteLine(d);

// Добавление связей в базу данных для создания ситуации со связями много ко многим.

dataBase.Add(new DataLink(2, 14));

dataBase.Add(new DataLink(3, 13));

dataBase.Add(new DataLink(3, 12));

var q6 = from d1 in Departments

join link in dataBase on d1.ID equals link.DepartmentID into temp1

from t1 in temp1

join w in Workers on t1.WorkerID equals w.ID into temp2

from t2 in temp2

select new { department = d1, worker = t2};

Console.WriteLine("\nВывод списка сотрудников в каждом отделе при связи много ко многим:");

foreach (var link in q6) Console.WriteLine($"{link.worker}");

/\* var q7 = from link in q6

group link by link.department.ID into d

select new { department = d.Key, num = d.Count() };\*/

var q7 = from d1 in Departments

join link in dataBase on d1.ID equals link.DepartmentID into temp1

select new { department = d1, count = temp1.Count() };

Console.WriteLine("\nВывод списка всех отделов и количества сотрудников в каждом отделе при связи много ко многим:");

foreach (var d in q7) Console.WriteLine($"{d.department}, количество работников: {d.count}");

}

}

}

## DataLink.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab\_7

{

/// <summary>

/// Устанавливает одиночную связь между Worker и Department.

/// </summary>

class DataLink

{

public int DepartmentID;

public int WorkerID;

public DataLink(int DepartmentID, int WorkerID)

{

this.DepartmentID = DepartmentID;

this.WorkerID = WorkerID;

}

}

}

## Department.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab\_7

{

/// <summary>

/// Отдел.

/// </summary>

public class Department

{

public int ID;

public string name;

public Department(int ID, string name)

{

this.ID = ID;

this.name = name;

}

public override string ToString()

{

return $"Отдел с ID: {ID}, названием: {name}";

}

}

}

## Worker.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab\_7

{

/// <summary>

/// Работник.

/// </summary>

public class Worker

{

public int ID;

public string surname;

public int departmentID;

public Worker(int ID, string surname, int departmentID)

{

this.ID = ID;

this.surname = surname;

this.departmentID = departmentID;

}

public override string ToString()

{

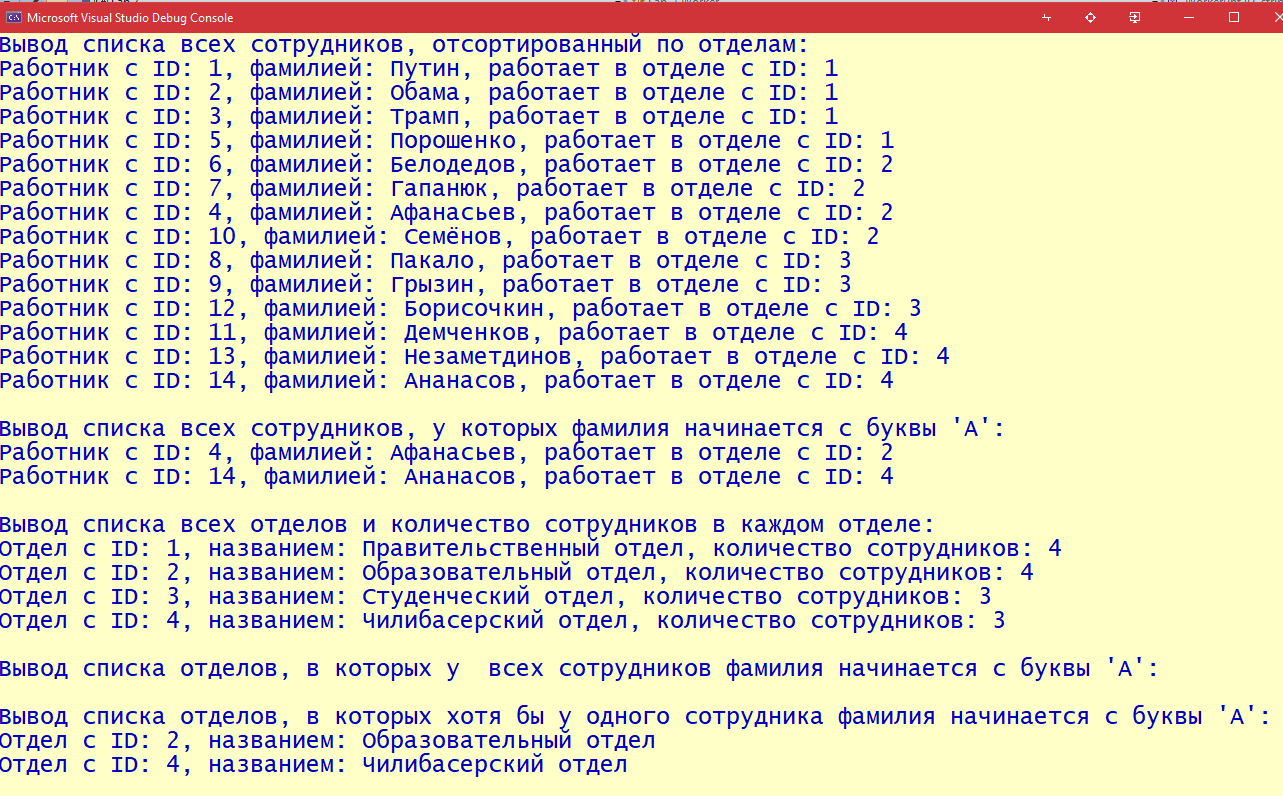
return $"Работник с ID: {ID}, фамилией: {surname}, работает в отделе с ID: {departmentID}";

}

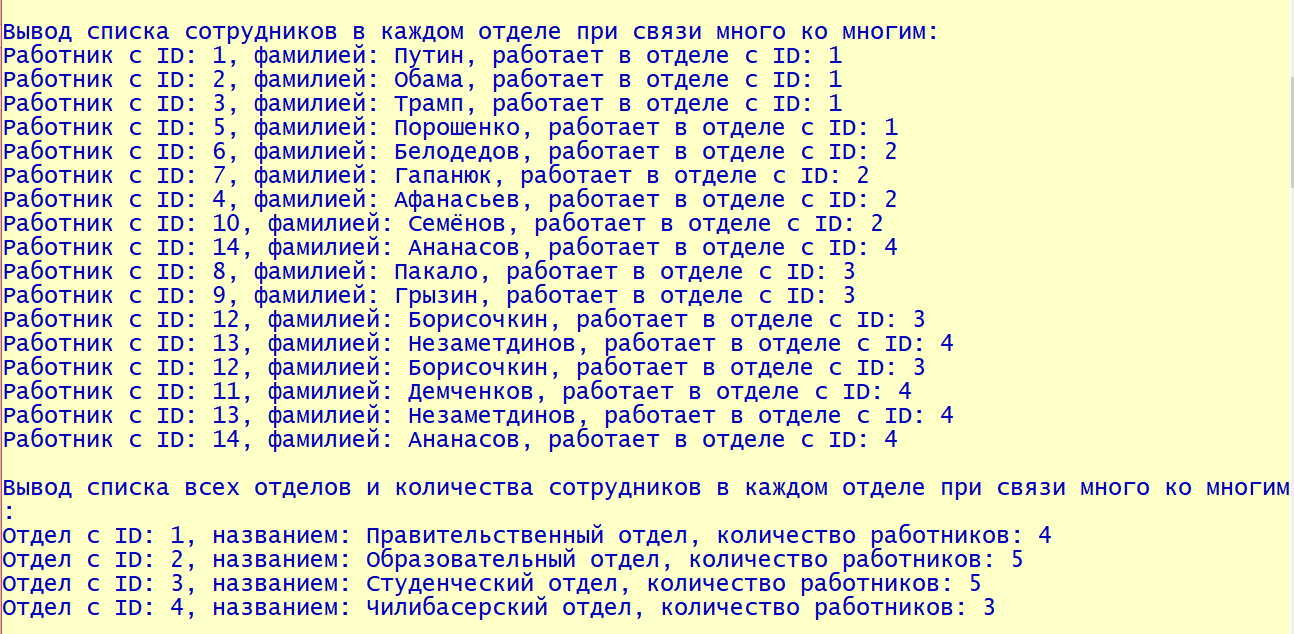
}

}

# Анализ результатов.



Скриншот 1 (демонстрация работы программы).



Скриншот 2 (демонстрация работы программы).