МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ИМ. Н. Э. БАУМАНА

Факультет "Информатика и системы управления"

### Кафедра ИУ5 “Системы обработки информации и управления”

Группа ИУ5-32Б

Отчёт по лабораторной работе №7

«Трек курса «Функциональное программирование»

Студент: Преподаватель:

Ваганов Даниил Дмитриевич Гапанюк Юрий Евгеньевич

Дата: \_\_\_.\_\_\_.\_\_\_\_\_\_ Дата: \_\_\_.\_\_\_.\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_ Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2020

**Условие 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:

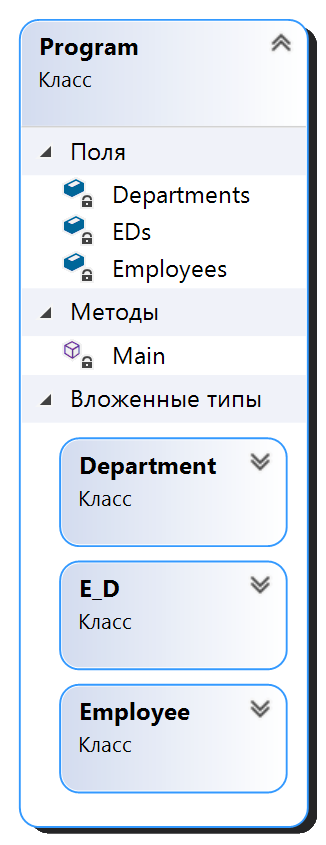
· ID записи о сотруднике;

· 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.

**Диаграмма классов:**

****

**Текст программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LAB7\_BKIT

{

class Program

{

public class Employee

{

public int id;

public string surname { get; set; }

public int dep\_id;

public Employee(int i, string s, int d)

{

this.id = i;

this.surname = s;

this.dep\_id = d;

}

public override string ToString()

{

return ($"ID = {this.id}, Surname = {this.surname} - \nDepartmentID = {this.dep\_id}");

}

}

public class Department

{

public int id;

public string name { get; set; }

public Department(int i, string n)

{

this.id = i;

this.name = n;

}

public override string ToString()

{

return ($"ID = {this.id}, Name = {this.name}");

}

}

public class E\_D

{

public int emp\_id { get; set; }

public int dep\_id { get; set; }

public E\_D(int employee\_id, int department\_id)

{

this.emp\_id = employee\_id;

this.dep\_id = department\_id;

}

}

static List<Employee> Employees = new List<Employee>()

{

new Employee(1, "Madsen", 1),

new Employee(2, "Bunker", 1),

new Employee(3, "Tarantino", 2),

new Employee(4, "Roth", 3),

new Employee(5, "Buscemi", 4),

new Employee(6, "Keitel", 4),

new Employee(7, "Penn", 1),

new Employee(8, "Tierney", 3),

new Employee(9, "Anderson", 3),

new Employee(10, "Alleman", 5),

new Employee(11, "Abu", 5),

new Employee(12, "Almond", 5)

};

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

{

new Department(1, "Stratton Oakmont"),

new Department(2, "L.F. Rothschild"),

new Department(3, "C.E. Unterberg"),

new Department(4, "Lehman Brothers"),

new Department(5, "AAA Company")

};

static List<E\_D> EDs = new List<E\_D>()

{

new E\_D(1, 3),

new E\_D(1, 1),

new E\_D(2, 2),

new E\_D(2, 5),

new E\_D(3, 1),

new E\_D(4, 2),

new E\_D(4, 4),

new E\_D(5, 4),

new E\_D(6, 2),

new E\_D(6, 3),

new E\_D(6, 5)

};

static void Main(string[] args)

{

Console.ForegroundColor = ConsoleColor.DarkGreen;

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

Console.ResetColor();

var r1 = from e in Employees

join d in Departments on e.dep\_id equals d.id

orderby d.name ascending

select new { ID = e.id, Surname = e.surname, Dep\_name = d.name };

foreach (var y in r1) Console.WriteLine($"ID = {y.ID}, Surname = {y.Surname} \t- \t({y.Dep\_name})");

Console.ForegroundColor = ConsoleColor.DarkGreen;

Console.WriteLine("\n---Вывод всех сотрудников, у которых фамилия начинается с буквы 'А'---");

Console.ResetColor();

var r2 = from e in Employees

join d in Departments on e.dep\_id equals d.id

where e.surname.StartsWith("A")

select new { ID = e.id, Surname = e.surname, Dep\_name = d.name };

foreach (var y in r2) Console.WriteLine($"ID = {y.ID}, Surname = {y.Surname} \t- \t({y.Dep\_name})");

Console.ForegroundColor = ConsoleColor.DarkGreen;

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

Console.ResetColor();

var r3 = (from e in Employees

join d in Departments on e.dep\_id equals d.id

select new

{

ID = d.id,

Name = d.name,

count = (from emp in Employees

where emp.dep\_id == d.id

select emp).Count()

}).Distinct();

foreach (var y in r3) Console.WriteLine($"ID = {y.ID}, Name = {y.Name}, Amount of Employyes = {y.count}");

Console.ForegroundColor = ConsoleColor.DarkGreen;

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

Console.ResetColor();

var r4 = from d in Departments

join e in Employees on d.id equals e.dep\_id into temp

select new { v1 = d.name, depGroup = temp };

foreach (var x in r4)

{

bool flag = true;

int k = 0;

foreach (var y in x.depGroup)

{ k++;

bool result = y.surname.StartsWith("A");

if (result != true)

flag = false;

}

if (flag == true && k > 0) Console.WriteLine(x.v1);

}

Console.ForegroundColor = ConsoleColor.DarkGreen;

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

Console.ResetColor();

foreach (var x in r4)

{ int k = 0;

foreach (var y in x.depGroup)

{

bool result = y.surname.StartsWith("A");

if (result == true)

k++;

}

if (k > 0) Console.WriteLine(x.v1);

}

Console.ForegroundColor = ConsoleColor.DarkGreen;

Console.WriteLine("\n---Имитация связи много-ко-многим---");

Console.ResetColor();

var lnk1 = from e in Employees

join l in EDs on e.id equals l.emp\_id into temp

from t1 in temp

join y in Departments on t1.dep\_id equals y.id into temp2

from t2 in temp2

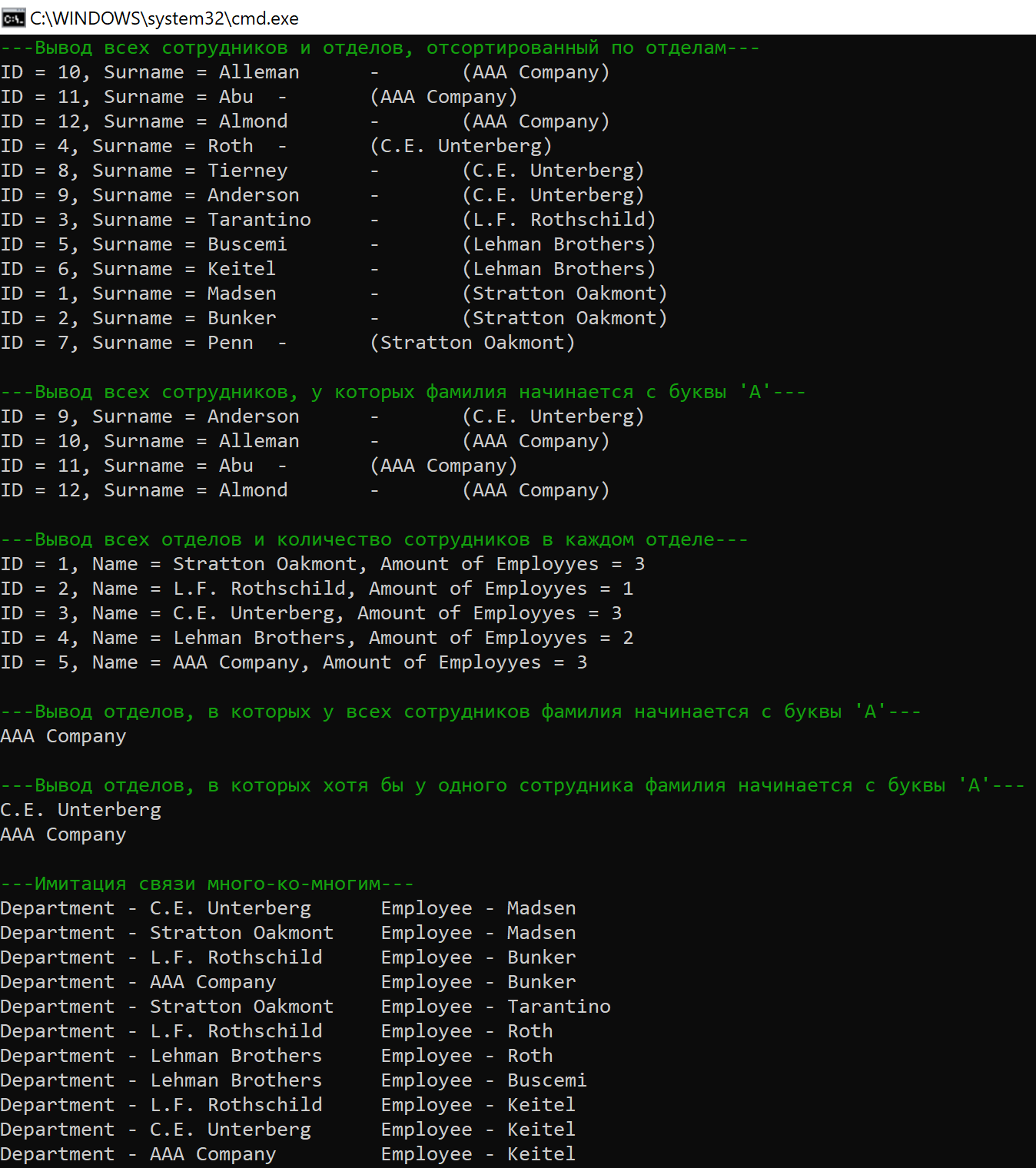
select new { dep\_name= t2.name, emp\_surname = e.surname };

foreach (var x in lnk1) Console.WriteLine($"Department - {x.dep\_name} \t Employee - {x.emp\_surname}");

}

}

}

**Анализ результатов:**