Лабораторная работа 10 по С#

LINQ.

студент, заочное, 40331, Василевский Евгений Валерьевич

**Задание 1**

Задание 1. LINQ с массивом. Фильтрация Создайте массив из 30 элементов типа int. Заполните его случайными числами. Выберите диапазон случайных чисел, оптимальный для решаемой задачи. Выполните задание, используя сначала запрос LINQ, а затем метод расширения.

Console.WriteLine("Лабораторная работа 10. Задание 1. Вариант 10");

int[] numbers= new int[30];

Random random = new Random();

for (int i = 0; i < numbers.Length; i++)

{

numbers[i] = random.Next(-20,80);

Console.Write(numbers[i]+" ");

}

Console.WriteLine();

//var result1 = from n in numbers

// where n.ToString().Contains('3')

// select n;

//foreach (var result in result1)

//{

// Console.Write(result + " ");

//}

//Console.WriteLine();

//foreach (var result in numbers)

//{

// Console.Write(result+" ");

//}

//Console.WriteLine();

//var result2 = numbers.Where(n => n.ToString().Contains('3'));

//foreach (var res in result2)

//{

// Console.Write(res + " ");

//}

//Console.WriteLine();

//var result3= numbers.TakeWhile(n=>n!=0);

//foreach (var res in result3) { Console.Write(res + " "); }

//Вариант 1 Создать новый массив из всех чисел в диапазоне от 10 до 20. Определить,

//есть ли в массиве хотя бы одно число, равное 0.

var result = numbers.Where(x => x >= 10 && x <= 20);

foreach (int i in result)

{

Console.Write(i + " ");

}

Console.WriteLine();

Console.WriteLine(numbers.Any(x => x == 0) ? "В массиве есть 0" : "В массиве нет 0");

//Вариант 2 Создать новый массив из всех отрицательных чисел. Определить, все ли

//числа в массиве отрицательные.

//var result = numbers.Where(x => x <0);

//foreach (int i in result)

//{

// Console.Write(i + " ");

//}

//Console.WriteLine();

//Console.WriteLine(result.All(x => x < 0) ? "В массиве все отрицательные" : "В массиве есть положительные");

//Вариант 3 Создать новый массив из всех чисел, кратных 3. Вывести с начала

//массива все числа до первого, кратного 3.

//var result = numbers.Where(x => x %3==0);

//foreach (int i in result)

//{

// Console.Write(i+" ");

//}

//Console.WriteLine();

//var result1 = numbers.TakeWhile(x => x % 3 != 0);

//foreach (int i in result1)

//{

// Console.Write(i+" ");

//}

//Console.WriteLine();

//Вариант 4 Создать новый массив из всех чисел, заканчивающихся на 7. Вывести

//массив, начиная с первого отрицательного числа.

//var result = numbers.Where(x => x % 10 == 7);

//foreach (int i in result)

//{

// Console.Write(i+" ");

//}

//Console.WriteLine();

//var result1 = numbers.SkipWhile(x=>x>0).TakeWhile(x => x!=null);

//foreach (int i in result1)

//{

// Console.Write(i+" ");

//}

//Вариант 5 Создать новый массив из всех четных положительных чисел. Определить,

//есть ли в массиве хотя бы одно нечетное отрицательное число.

//var result = numbers.Where(x => x % 2 == 0 && x>=0);

//foreach (int i in result)

//{

// Console.Write(i + " ");

//}

//Console.WriteLine();

//Console.WriteLine(numbers.Any(x=> x<0 && x%2!=0)? "Есть нечетное отрицательное число": "Нет нечетных отрицательных чисел");

//Вариант 6 Создать новый массив из всех чисел, в которых есть цифра 7.

//Определить, все ли числа в массиве четные.

//var result = from l in numbers

// where l.ToString().Contains('7')

// select l;

//foreach (var l in result)

//{

// Console.Write(l+" ");

//}

//Console.WriteLine();

//Console.WriteLine(numbers.All(x => x % 2 == 0) ? "Все числа четные" : "есть нечетные числа");

//Вариант 7 Создать новый массив из всех чисел больше 10. Вывести первую группу

//положительных чисел в массиве.

//var result = numbers.Where(x =>x > 10);

//foreach (int i in result)

//{

// Console.Write(i + " ");

//}

//Console.WriteLine();

//var result1 = numbers.TakeWhile(x=>x>=0);

//foreach (int i in result1)

//{

// Console.Write(i+" ");

//}

//Вариант 8 Создать новый массив из всех чисел в диапазоне от -5 до 5. Вывести

//первую группу отрицательных чисел в массиве.

//var result = numbers.Where(x => x >= -5 && x<=5);

//foreach (int i in result)

//{

// Console.Write(i + " ");

//}

//Console.WriteLine();

//var result1 = numbers.SkipWhile(x=> x>=0).TakeWhile(x => x < 0);

//foreach (int i in result1)

//{

// Console.Write(i + " ");

//}

//Вариант 9 Создать новый массив из всех чисел, кратных 7. Определить, есть ли в

//массиве хотя бы одно число, кратное 5.

//var result = numbers.Where(x => x % 7 == 0);

//foreach (int i in result)

//{

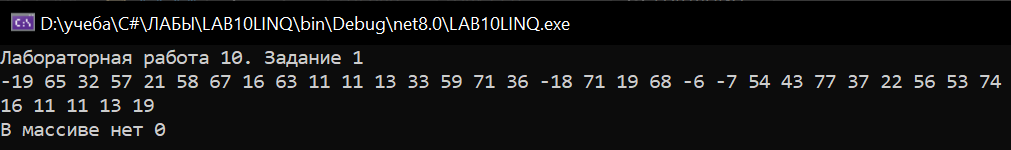
// Console.Write(i + " ");

//}

//Console.WriteLine();

//Console.WriteLine(numbers.Any(x => x % 5 == 0) ? "Есть число кратное 5" : "Нет чисел кратных 5");

Console.ReadKey();



**Задание 2**

Задание 2. LINQ с массивом. Группировка и агрегатные операции Создайте массив из 30 элементов типа int. Заполните его случайными числами. Выберите диапазон случайных чисел, оптимальный для решаемой задачи. Выполните задание, используя запрос LINQ или метод расширения.

Console.WriteLine("Лабораторная работа 10. Задание 2");

int[] numbers = new int[30];

Random random = new Random();

for (int i = 0; i < numbers.Length; i++)

{

numbers[i] = random.Next(-20, 800);

Console.Write(numbers[i] + " ");

}

Console.WriteLine();

//var result=numbers.Where(n => n>=10 && n<=20).Sum();

//Console.WriteLine($"sum = {result}");

//var result2 = numbers.GroupBy(n => n >= 0 ? "Положительные" : "Отрицательные");

//foreach (var group in result2)

//{

// var min = group.Min();

// var max = group.Max();

// var sum = min+max;

// Console.WriteLine($"{group.Key}: min = {min}, max = {max}, sum = {sum}");

// foreach(var item in group)

// {

// Console.Write(item + " ");

// }

// Console.WriteLine();

//}

//Вариант 1 Определить минимум среди четных положительных чисел.

//Сгруппировать отрицательные и положительные числа, определить их

//количество и среднее.

//var result =numbers.Where(x=> x > 0 && x%2==0).Min();

//Console.WriteLine(result);

//var result1 = numbers.GroupBy(n=> n>=0 ? "Положительные":"Отрицательные");

//foreach (var x in result1)

//{

// Console.Write(x.Key+ ": ");

// Console.WriteLine($"Количество: {x.Count()}, Среднее: {x.Average():0.00}");

//}

//Вариант 2 Определить максимум среди всех отрицательных чисел. Сгруппировать

//четные и нечетные числа, определить максимум и минимум среди них.

//var result = numbers.Where(x=> x<0).Max();

//Console.WriteLine(result);

//var result1 = numbers.GroupBy(x => x % 2 == 0 ? "Четные" : "Нечетные");

//foreach (var x in result1)

//{

// Console.Write(x.Key+": ");

// Console.WriteLine($"Min: {x.Min()}, Max: {x.Max()}");

//}

//Вариант 3 Найти максимальное число, кратное 7. Сгруппировать четные и нечетные

//числа, определить максимум и минимум среди них.

//var result= numbers.Where(x=> x%7==0).Max();

//Console.WriteLine(result);

//var result1=numbers.GroupBy(x=> x%2== 0 ? "Четные" : "Нечетные");

//foreach (var x in result1)

//{

// Console.Write(x.Key + ": ");

// Console.WriteLine($"Min: {x.Min()}, Max: {x.Max()}");

//}

//Вариант 4 Найти среднее значение среди отрицательных чисел. Сгруппировать в

//массиве числа по десяткам: от 0 до 9, от 10 до 19 и т.д. Найти их к-во в

//каждой группе.

//var result=numbers.Where(x=> x<0).Average();

//Console.WriteLine($"{ result: 0.00}");

//var result1 = from l in numbers

// group l by Math.Abs(l) / 10;

//foreach (var x in result1)

//{

// Console.WriteLine(x.Key+": ");

// Console.WriteLine($"Количство: {x.Count()}");

//}

//Вариант 5 Определить количество чисел, кратных 3. Сгруппировать в массиве числа

//по последней цифре числа. Найти их к-во в каждой группе.

//var result = numbers.Where(x => x%3== 0).Count();

//Console.WriteLine($"{result}");

//var result1 = from l in numbers

// group l by Math.Abs(l) % 10;

//foreach (var i in result1)

//{

// Console.Write(i.Key+": ");

// Console.WriteLine(i.Count());

//}

//Вариант 6 Вычислить сумму всех отрицательных чисел. Сгруппировать числа в 2

//группы: которые делятся на 3 и остальные. Найти сумму для каждой

//группы.

//var result=numbers.Where(x=> x<0).Sum();

//Console.WriteLine(result);

//var result1 = numbers.GroupBy(x => x % 3 == 0 ? "Делятся на 3" : "Не делятся на три");

//foreach (var x in result1)

//{

// Console.Write(x.Key + ": ");

// Console.WriteLine($"Sum: {x.Sum()}");

//}

//Вариант 7 Определить количество чисел в диапазоне от -5 до 5. Сгруппировать в

//массиве числа по сотням: от 0 до 99, от 100 до 199 и т.д. Найти их к-во в

//каждой группе.

//var result =numbers.Where(x => x >= -5&& x<=5).Count();

//Console.WriteLine(result);

//var result1 = from l in numbers

// group l by Math.Abs(l) / 100;

//foreach (var x in result1)

//{

// Console.Write(x.Key + ": ");

// Console.WriteLine($"Количство: {x.Count()}");

//}

//Вариант 8 Определить количество четных положительных чисел. Сгруппировать в

//массиве числа по предпоследней цифре числа. Найти их к-во в каждой

//группе.

//var result = numbers.Where(x => x%2==0 && x > 0).Count();

//Console.WriteLine(result);

//var result1 = from l in numbers

// group l by Math.Abs(l)%100 / 10;

//foreach (var x in result1)

//{

// Console.Write(x.Key + ": ");

// Console.WriteLine($"Количство: {x.Count()}");

//}

//Вариант 9 Вычислить произведение минимального и максимального чисел.

//Сгруппировать четные и нечетные числа, определить среднее значение

//для каждой группы.

Console.WriteLine(numbers.Min()\*numbers.Max());

var result1 = numbers.GroupBy(x => x % 2 == 0 ? "Четные" : "Нечетные");

foreach (var x in result1)

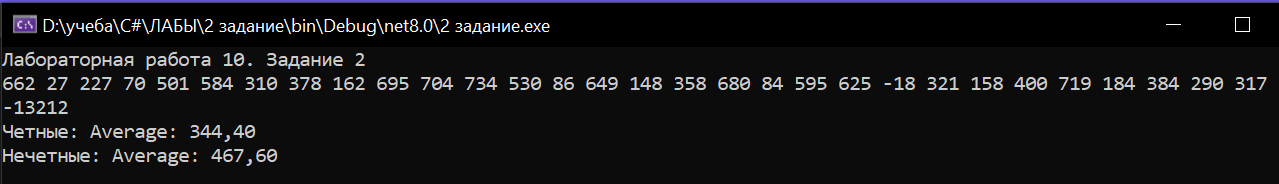
{

Console.Write(x.Key + ": ");

Console.WriteLine($"Average: {x.Average():0.00}");

}

Console.ReadKey();



**Задание 3**

Console.WriteLine("Лабораторная работа 10. Задание 3");

int[] numbers1 = new int[30];

Random random = new Random();

for (int i = 0; i < numbers1.Length; i++)

{

numbers1[i] = random.Next(-10, 10);

Console.Write(numbers1[i] + " ");

}

Console.WriteLine();

int[] numbers2 = new int[30];

for (int i = 0; i < numbers2.Length; i++)

{

numbers2[i] = random.Next(-20, 20);

Console.Write(numbers2[i] + " ");

}

Console.WriteLine();

//var result1=numbers1.TakeWhile(n => n > 0);

//var result2=numbers2.TakeWhile(n => n < 0);

//var result3=result1.Concat(result2);

//foreach (var result4 in result3)

//{

// Console.Write(result4+ " ");

//}

//Console.WriteLine();

//Console.WriteLine();

//result1 = numbers1.Where(n => n > 0).Take(3);

//result2 = numbers2.Where(n => n < 0).Take(3);

//result3 = result1.Concat(result2);

//foreach (var result4 in result3)

//{

// Console.Write(result4 + " ");

//}

//Console.WriteLine();

//Вариант 1 2 массива. Создайте новый массив – пересечение четных чисел из

//первого и отрицательных из второго.

//var result = numbers1.Where(x => x % 2 == 0);

//var result1=from l in numbers2

// where l<0

// select l;

//var res=result.Intersect(result1);

//foreach (var x in res)

// Console.Write(x+" ");

//Вариант 2 2 массива. Создайте новый массив – объединение положительных чисел

//из первого и отрицательных из второго.

//var result = numbers1.Where(x => x >0);

//var result1 = from l in numbers2

// where l < 0

// select l;

//var res = result.Union(result1);

//foreach (var x in res)

// Console.Write(x + " ");

//Вариант 3 2 массива. Создайте новый массив – разность положительных чисел из

//первого и положительных из второго.

//var result = numbers1.Where(x => x > 0);

//var result1 = from l in numbers2

// where l > 0

// select l;

//var res = result.Except(result1);

//foreach (var x in res)

// Console.Write(x + " ");

//Вариант 4 2 массива. Создайте новый массив – конкатенацию нечетных чисел из

//первого и нечетных из второго.

//var result = numbers1.Where(x => x %2!= 0);

//var result1 = from l in numbers2

// where l %2!= 0

// select l;

//var res = result.Concat(result1);

//foreach (var x in res)

// Console.Write(x + " ");

//Вариант 5 1 массив.Отсортировать все нечетные числа по убыванию, а четные - по

//возрастанию.

//var result = numbers1.Where(x => x % 2 != 0).OrderDescending();

//var result1 = from l in numbers1

// where l % 2 == 0

// orderby l

// select l;

//var res = result.Concat(result1);

//foreach (var x in res)

// Console.Write(x + " ");

//Вариант 6 2 массива. Создайте новый массив – пересечение положительных чисел

//из первого и нечетных из второго.

//var result = numbers1.Where(x => x > 0);

//var result1 = from l in numbers2

// where l %2!= 0

// select l;

//var res = result.Intersect(result1);

//foreach (var x in res)

// Console.Write(x + " ");

//Вариант 7 2 массива. Создайте новый массив – поэлементная сумма массивов.

//var result=numbers1.Zip(numbers2,(a,b)=>a+b);

//foreach (var g in result)

//{

// Console.Write(g+" ");

//}

//Вариант 8 2 массива. Создайте новый массив – конкатенацию отрицательны чисел

//из первого и положительных из второго.

//var result = numbers1.Where(x => x < 0);

//var result1 = from l in numbers2

// where l > 0

// select l;

//var res = result.Concat(result1);

//foreach (var x in res)

// Console.Write(x + " ");

//Вариант 9 2 массива. Создайте новый массив – разность четных чисел из первого и

//положительных из второго.

var result = numbers1.Where(x => x %2== 0);

var result1 = from l in numbers2

where l > 0

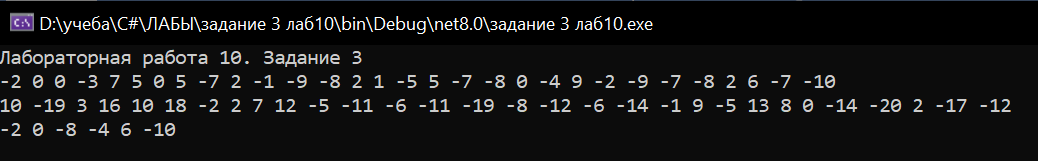
select l;

var res = result.Except(result1);

foreach (var x in res)

Console.Write(x + " ");

Console.ReadKey();



Задание 4

Задание 4. LINQ с коллекцией Создайте коллекцию объектов класса Person. Используемые поля – имя, дата рождения, должность, оклад, компания (string). Напишите метод, который принимает необходимые аргументы и возвращает новую коллекцию согласно варианту (класс новой коллекции содержит только указанные поля).

Console.WriteLine("Лабораторная работа 10. Задание 4");

List<Person> people = new List<Person>()

{

new Person("Иван Иванов", new DateTime(1990,5,15), "Разработчик",60000,"Google"),

new Person("Мария Петрова", new DateTime(1985,8,10), "Менеджер",80000,"Xerox"),

new Person("Сергей Смирнов", new DateTime(1992,3,10), "Аналитик",50000,"Google"),

new Person("Анна Кузнецова", new DateTime(1988,12,5), "Дизайнер",70000,"Xerox"),

new Person("Петр Петров", new DateTime(1950,3,5), "Директор",20000,"Google"),

new Person("Ольга Сидорова", new DateTime(1985,4,20), "Шофер",45000,"Xerox"),

new Person("Кузьма Плоткин", new DateTime(1992,11,10), "Грузчик",34000,"Google"),

new Person("Игорь Турецкий", new DateTime(1950,2,5), "Дизайнер",90000,"Xerox")

};

Console.WriteLine("Список сотрудников");

foreach (Person person in people)

{

Console.WriteLine(person);

}

Console.WriteLine();

//int oklad = 50000;

//string company = "Google";

//var people1 = people.Where(p => p.Company == company&& p.Salary<=oklad).Select(p => $"Имя: {p.Name}, Должность: {p.Position}, Оклад: {p.Salary}, Компания: {p.Company}");

//foreach(var person in people1)

//{

// Console.WriteLine(person);

//}

//Вариант 1 Список сотрудников на определенной должности с окладом, выше

//указанного.Имя сотрудника, должность, оклад, компания.

//int oklad = 80000;

//string dol = "Дизайнер";

//var result = from p in people

// where p.Salary > oklad && p.Position == dol

// select new

// {

// Name = p.Name,

// Position = p.Position,

// Salary = p.Salary,

// Company = p.Company

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.Position} {person.Salary} {person.Company}");

//}

//Вариант 2 Список сотрудников из определенной компании, не старше определенной

//даты.Имя сотрудника, дата рождения, компания.

//string com = "Google";

//DateTime date = Convert.ToDateTime("15.06.1991 00:00:00");

//var result = from p in people

// where p.Company == com && p.BirthDate.Year-date.Year>0

// select new

// {

// Name = p.Name,

// BirthDay= p.BirthDate,

// Company = p.Company

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.BirthDay} {person.Company}");

//}

//Вариант 3 Список сотрудников, родившихся в определенный месяц с окладом, не

//выше указанного. Имя сотрудника, дата рождения, оклад, компания.

//int oklad = 50000;

//int Month = 3;

//var result = from p in people

// where p.Salary <= oklad && p.BirthDate.Month - Month== 0

// select new

// {

// Name = p.Name,

// BirthDay = p.BirthDate,

// Salary = p.Salary,

// Company = p.Company

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.BirthDay} {person.Salary} {person.Company}");

//}

//Вариант 4 Список сотрудников, чье имя начинается с определенной буквы. Имя

//сотрудника, дата рождения, компания.

//char a = 'И';

//var result = from p in people

// where p.Name.StartsWith(a)

// select new

// {

// Name = p.Name,

// BirthDay = p.BirthDate,

// Company = p.Company

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.BirthDay} {person.Company}");

//}

//Вариант 5 Список сотрудников пенсионного возраста на определенной должности.

//Имя сотрудника, дата рождения, компания, должность.

//string dol= "Дизайнер";

//var result = from p in people

// where p.Position ==dol && DateTime.Now.Year-p.BirthDate.Year>63

// select new

// {

// Name = p.Name,

// BirthDay = p.BirthDate,

// Position=p.Position,

// Company = p.Company

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.BirthDay} {person.Position} {person.Company}");

//}

//Вариант 6 Список сотрудников, родившихся в определенный день месяца с

//окладом, не ниже указанного. Имя сотрудника, дата рождения, оклад,

//компания.

//int oklad = 50000;

//int day = 10;

//var result = from p in people

// where p.Salary >= oklad && p.BirthDate.Day== day

// select new

// {

// Name = p.Name,

// BirthDay = p.BirthDate,

// Salary = p.Salary,

// Company = p.Company

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.BirthDay} {person.Salary} {person.Company}");

//}

//Вариант 7 Список должностей с окладом без повторений. Содержит должность,

//оклад.

//var result = from p in people

// select new

// {

// Position=p.Position,

// Salary = p.Salary,

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Position} {person.Salary}");

//}

//Вариант 8 Список сотрудников моложе определенного возраста с окладом, не ниже

//указанного.Имя сотрудника, дата рождения, компания, должность.

//int oklad = 40000;

//int voz = 39;

//var result = from p in people

// where p.Salary>=oklad && DateTime.Now.Year - p.BirthDate.Year < voz

// select new

// {

// Name = p.Name,

// BirthDay = p.BirthDate,

// Position = p.Position,

// Company = p.Company

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.BirthDay} {person.Position} {person.Company}");

//}

//Вариант 9 Список сотрудников, родившихся в определенный год для указанной

//компании.Имя сотрудника, дата рождения, компания.

string com = "Xerox";

int Year = 1985;

var result = from p in people

where p.Company == com && p.BirthDate.Year == Year

select new

{

Name = p.Name,

BirthDay = p.BirthDate,

Company = p.Company

};

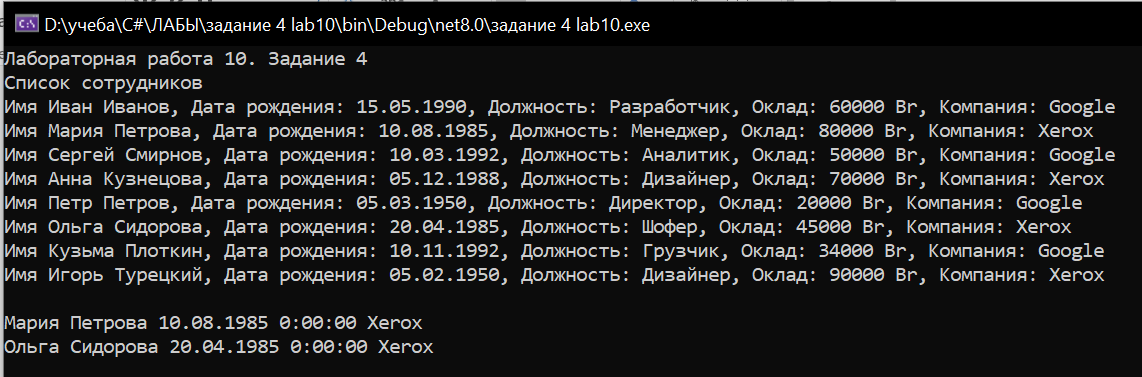
foreach (var person in result)

{

Console.WriteLine($"{person.Name} {person.BirthDay} {person.Company}");

}

Console.ReadKey();



Задание 5

Задание 5. LINQ с вложенной коллекцией Создайте коллекцию объектов класса Person. Используемые поля – имя, год рождения, должность, оклад, компания (Company). Класс Company содержит название компании и год основания. Получите новую коллекцию, согласно варианту.

Console.WriteLine("Лабораторная работа 10. Задание 5.");

Company google = new Company("Google", 1998);

Company xerox = new Company("Xerox", 1906);

Company apple = new Company("Apple", 1986);

Company tesla = new Company("Tesla", 2004);

HashSet<Company> companies = new HashSet<Company>{ google, xerox, apple, tesla };

List<Person> people = new List<Person>()

{

new Person("Gван Иванов", new DateTime(1990,5,15), "Разработчик",60000,google),

new Person("Xария Петрова", new DateTime(1985,8,20), "Менеджер",80000,xerox),

new Person("Сергей Смирнов", new DateTime(1992,3,10), "Аналитик",50000,google),

new Person("Анна Кузнецова", new DateTime(1988,12,5), "Дизайнер",70000,xerox),

new Person("Петр Петров", new DateTime(1950,3,5), "Директор",20000,google),

new Person("Ольга Сидорова", new DateTime(1985,4,20), "Шофер",45000,xerox),

new Person("Кузьма Плоткин", new DateTime(1999,11,10), "Грузчик",34000,google),

new Person("Игорь Турецкий", new DateTime(1950,2,5), "Дизайнер",90000,xerox)

};

//var maxAgeByCompany = people

// .GroupBy(x => x.Company.Name)

// .Select(g => new

// {

// CompanyName = g.Key,

// MaxAge = g.Max(p => DateTime.Now.Year - p.BirthDate.Year)

// });

//foreach (var person in maxAgeByCompany)

//{

// Console.WriteLine($"Компания: {person.CompanyName}, Максимальный возраст: {person.MaxAge} лет");

//}

//Вариант 1 Список сотрудников, которые старше их компаний. Включает имя,

//возраст, Company.

//var result = from p in people

// where p.BirthDate.Year - p.Company.YearOfFoundation < 0

// select new

// {

// Name = p.Name,

// Voz = DateTime.Now.Year - p.BirthDate.Year,

// Com = p.Company.Name,

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.Voz} {person.Com}");

//}

//Вариант 2 Список компаний, в которых не работает ни одного сотрудника.

//var result = new HashSet<Company>(people.Select(x=>x.Company));

//companies.ExceptWith(result);

//foreach (var company in companies)

//{

// Console.WriteLine(company.Name);

//}

//Вариант 3 Список должностей с названием компании и окладом, без повторений.

//var result = people.

// Select(g => new

// {

// g.Position,

// CompanyName = g.Company.Name,

// g.Salary

// }).

// GroupBy(x => new { x.Position, x.CompanyName, x.Salary }).

// Select(l => l.First());

//foreach (var item in result)

//{

// Console.WriteLine($"{item.Position } {item.CompanyName } {item.Salary }");

//}

//Вариант 4 Список сотрудников, чье имя начинается с той же буквы, что и их

//компания.Включает имя сотрудника и имя компании.

//var result = from l in people

// where l.Company.Name.StartsWith(l.Name[0])

// select new

// {

// Name = l.Name,

// Company = l.Company.Name,

// };

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Name} {person.Company}");

//}

//Вариант 5 Список компаний, в которых более одного сотрудника. Содержит

//название компании и к-во сотрудников.

//var result = people.

// GroupBy(x => x.Company.Name).

// Where(h => h.Count() > 1).

// Select(n => new

// {

// Com = n.Key,

// Kol = n.Count()

// });

//foreach(var com in result)

//{

// Console.WriteLine($"{com.Com} {com.Kol}");

//}

//Вариант 6 Список лет, в которых были основаны компании. Содержит год и к-во

//компаний.

//var result = companies.

// GroupBy(x => x.YearOfFoundation).

// Select(g => new

// {

// god = g.Key,

// kol = g.Count()

// });

//foreach(var com in result)

//{

// Console.WriteLine($"{com.god} {com.kol} ");

//}

//Вариант 7 Список должностей с суммарным окладом, который получают все

//сотрудники на этой должности в этой компании. Содержит компанию,

//должность, сумму.

//var result = people.

// GroupBy(x => new { x.Company.Name, x.Position }).

// Select(g => new

// {

// Com = g.Key.Name,

// pos=g.Key.Position,

// okl = g.Sum(p=>p.Salary)

// });

//foreach (var person in result)

//{

// Console.WriteLine($"{person.Com} {person.pos} {person.okl}");

//}

//Вариант 8 Список лет, от минимального, до максимального, в который не было

//основано ни одной компании и не родился ни один из сотрудников.

//var res1=from l in people

// orderby l.BirthDate.Year

// select l.BirthDate.Year;

//var res2 = from l in companies

// orderby l.YearOfFoundation

// select l.YearOfFoundation;

//int min = res1.Union(res2).Min();

//int max = res2.Union(res1).Max();

//var ints = Enumerable.Range(min, max - min + 1).Except(res1).Except(res2);

//foreach (var p in ints)

//{

// Console.Write(p+" ");

//}

//Вариант 9 Сумму окладов по каждой компании.

var res = people.

GroupBy(x => x.Company.Name).

Select(g => new

{

CompanyName=g.Key,

okl=g.Sum(p=>p.Salary)

});

foreach (var person in res)

{

Console.WriteLine($"{person.CompanyName} {person.okl}");

}

Console.ReadKey();

