# Лабораторная работа 6. Обобщения и коллекции.

## Задание.

**1. Использование коллекции ArrayList.**

Используя ArrayList, создать коллекцию объектов Interface1 (из предыдущей работы). Сразу проинициализировать ее значениями различных классов (в коде).

Спроектировать консольное меню со следующим функционалом:

1 – просмотр коллекции

2 – добавление элемента (данные вводим с клавиатуры)

3 – добавление элемента по указанному индексу (индекс и данные вводим с клавиатуры)

4 – нахождение элемента с начала коллекции (переопределить метод Equals или оператор == для вашего класса – сравнение только по полю name) (данные для поиска по полю name вводим с клавиатуры, вы должны иметь минимум 2 объекта в коллекции с одинаковыми именами)

5 – нахождение элемента с конца коллекции (данные для поиска по полю name вводим с клавиатуры, вы должны иметь минимум 2 объекта в коллекции с одинаковыми именами)

6 – удаление элемента по индексу (индекс вводим с клавиатуры)

7 – удаление элемента по значению (данные для поиска по полю name вводим с клавиатуры)

8 – реверс коллекции

9 – сортировка

10 – выполнение методов всех объектов, поддерживающих Interface2

0 – выход

*Внимание!*

*Изначально заполните коллекцию некоторым количеством объектов. В дальнейшем для добавления, удаления и поиска элементов всю информацию запрашивать у пользователя.*

using Csharp\_Lab5\_Var4\_ConsoleApp; // добавим ссылку на предыдущюю лабу

class Program

{

static void Main(string[] args)

{

ArrayList arrayList = new ArrayList()

{

new Printer("Canon"),

new Multifunction("Xerox","xxx"),

new Multifunction("Samsung","sss"),

new Printer("Epson"),

new Multifunction("Kyocera","kkk"),

new Printer("Xerox","XOX"),

new Printer("HPLaserJet")

};

while (true)

{

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("\* Главное меню \*");

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("1 – просмотр коллекции");

Console.WriteLine("2 – добавление элемента");

Console.WriteLine("3 – добавление элемента по указанному индексу");

Console.WriteLine("4 – нахождение элемента с начала коллекции");

Console.WriteLine("5 – нахождение элемента с конца коллекции");

Console.WriteLine("6 – удаление элемента по индексу");

Console.WriteLine("7 – удаление элемента по значению");

Console.WriteLine("8 – реверс коллекции");

Console.WriteLine("9 – сортировка");

Console.WriteLine("10 – выполнение методов всех объектов, поддерживающих интерфейс IScanning");

Console.WriteLine("0 – выход");

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

int otvet;

if (!Int32.TryParse(Console.ReadLine(), out otvet))

{

Console.WriteLine("Введите верное значение! Выход - 0");

continue;

}

switch (otvet)

{

case 1: method1(arrayList); break;

case 2: method2(arrayList); break;

case 3: method3(arrayList); break;

case 4: method4(arrayList); break;

case 5: method5(arrayList); break;

case 6: method6(arrayList); break;

case 7: method7(arrayList); break;

case 8: method8(arrayList); break;

case 9: method9(arrayList); break;

case 10: method10(arrayList); break;

default: return;

}

}

}

static void method1(ArrayList arrayList) // просмотр коллекции

{

Console.WriteLine("просмотр коллекции\n----------------------");

for(int i =0;i<arrayList.Count;i++)

{

Console.Write($"{i} "); // вывод индекса элемента

(arrayList[i] as Printer)?.Info();

}

Console.ReadKey();

}

static void DataEl(out string name,out string buffer)

{

Console.WriteLine("Название:");

name = Console.ReadLine();

Console.WriteLine("Содержимое буфера:");

buffer = Console.ReadLine();

}

static void method2(ArrayList arrayList)

{

Console.WriteLine("добавление элемента\n----------------------");

Console.WriteLine("Выберите какой элемент вы хотите добавить! 1- Принтер; 2 - МФУ");

int otvet;

if (Int32.TryParse(Console.ReadLine(), out otvet)) { }

if (otvet < 1 || otvet > 2)

{

Console.WriteLine("Выбор элемента неверный!");

return;

}

string name, buffer;

DataEl(out name,out buffer);

switch (otvet){

case 1:

arrayList.Add(new Printer(name, buffer));

break;

case 2:

arrayList.Add(new Multifunction(name, buffer));

break;

default:return;

}

Console.WriteLine("элемент добавлен!");

Console.ReadKey();

}

static void method3(ArrayList arrayList)

{

Console.WriteLine("добавление элемента по указанному индексу\n----------------------");

Console.WriteLine("введите индекс:");

int idx;

if(Int32.TryParse(Console.ReadLine(),out idx)) { }

if(idx<0 || idx > arrayList.Count)

{

Console.WriteLine("Неверный индекс!");

return;

}

Console.WriteLine("Выберите какой элемент вы хотите добавить! 1- Принтер; 2 - МФУ");

int otvet;

if (Int32.TryParse(Console.ReadLine(), out otvet)) { }

if (otvet < 1 || otvet > 2)

{

Console.WriteLine("Выбор элемента неверный!");

return;

}

string name, buffer;

DataEl(out name, out buffer);

switch (otvet)

{

case 1:

arrayList.Insert(idx,new Printer(name, buffer));

break;

case 2:

arrayList.Insert(idx,new Multifunction(name, buffer));

break;

default: return;

}

Console.WriteLine($"элемент добавлен по индексу {idx}!");

Console.ReadKey();

}

static void method4(ArrayList arrayList)

{

Console.WriteLine("нахождение элемента с начала коллекции\n----------------------");

Console.WriteLine("Name=");

string name = Console.ReadLine();

int idx = arrayList.IndexOf(new Printer(name));

if (idx < 0)

Console.WriteLine($"элемент с именем {name} не найден!");

else

Console.WriteLine($"первый элемент с начала коллекции с именем {name} имеет индекс {idx}!");

Console.ReadKey();

}

static void method5(ArrayList arrayList)

{

Console.WriteLine("нахождение элемента с конца коллекции\n----------------------");

Console.WriteLine("Name=");

string name = Console.ReadLine();

int idx = arrayList.LastIndexOf(new Printer(name));

if (idx < 0)

Console.WriteLine($"элемент с именем {name} не найден!");

else

Console.WriteLine($"первый элемент с конца коллекции с именем {name} имеет индекс {idx}!");

Console.ReadKey();

}

static void method6(ArrayList arrayList)

{

Console.WriteLine("удаление элемента по индексу\n----------------------");

Console.WriteLine("введите индекс:");

int idx;

if (Int32.TryParse(Console.ReadLine(), out idx)) { }

if (idx < 0 || idx > arrayList.Count)

{

Console.WriteLine("Неверный индекс!");

return;

}

arrayList.RemoveAt(idx);

Console.WriteLine($"элемент по индексу {idx} удален!");

Console.ReadKey();

}

static void method7(ArrayList arrayList)

{

Console.WriteLine("удаление элемента по значению\n----------------------");

Console.WriteLine("Name:");

string name;

name = Console.ReadLine();

Printer printer = new Printer(name);

if (arrayList.IndexOf(printer) == -1)

{

Console.WriteLine("элемент с таким именем не найден!");

return;

}

arrayList.Remove(printer);

Console.WriteLine($"элемент с именем {name} удален!");

Console.ReadKey();

}

static void method8(ArrayList arrayList)

{

Console.WriteLine("реверс коллекции\n----------------------");

arrayList.Reverse();

Console.WriteLine("успех!");

Console.ReadKey();

}

static void method9(ArrayList arrayList)

{

Console.WriteLine("сортировка\n----------------------");

arrayList.Sort();

Console.WriteLine("успех!");

Console.ReadKey();

}

static void method10(ArrayList arrayList)

{

Console.WriteLine("выполнение методов всех объектов, поддерживающих интерфейс IScanning\n----------------------");

foreach(object obj in arrayList)

{

if(obj is IScanning)

{

Console.WriteLine(obj);

(obj as IScanning).Scan("string");

}

}

Console.ReadKey();

}

}

**2. Использование параметризованной коллекции**

Используя параметризованную коллекцию, продемонстрировать ее возможности по аналогии с п.1.

Варианты:

1. List (меню как в ArrayList)
2. Dictionary (добавить работу с ключами)
3. LinkedList
4. Queue
5. SortedSet
6. SortedList
7. SortedDictionary

class Program

{

static void Main(string[] args)

{

Queue<Printer> queue = new Queue<Printer>();

queue.Enqueue(new Printer("Canon"));

queue.Enqueue(new Multifunction("Xerox"));

queue.Enqueue(new Multifunction("Samsung"));

queue.Enqueue(new Printer("Epson"));

queue.Enqueue(new Multifunction("Kyocera"));

queue.Enqueue(new Multifunction("Xerox"));

queue.Enqueue(new Printer("HPLaserJet"));

while (true)

{

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("\* Главное меню \*");

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("1 – просмотр коллекции");

Console.WriteLine("2 – добавление элемента в конец очереди");

Console.WriteLine("3 – просмотр первого элемента в очереди");

Console.WriteLine("4 – извлечение первого элемента из очереди");

Console.WriteLine("5 – количество элементов в очереди");

Console.WriteLine("0 – выход");

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

int otvet;

if (!Int32.TryParse(Console.ReadLine(), out otvet))

{

Console.WriteLine("Введите верное значение! Выход - 0");

continue;

}

switch (otvet)

{

case 1: method1(queue); break;

case 2: method2(queue); break;

case 3: method3(queue); break;

case 4: method4(queue); break;

case 5: method5(queue); break;

default: return;

}

}

}

static void method1(Queue<Printer> printers) // просмотр коллекции

{

Console.WriteLine("просмотр коллекции\n----------------------");

foreach (Printer printer in printers)

printer.Info();

Console.ReadKey();

}

static void method2(Queue<Printer> printers)

{

Console.WriteLine("добавление элемента в конец очереди\n----------------------");

Console.WriteLine("Выберите какой элемент вы хотите добавить! 1- Принтер; 2 - МФУ");

int otvet;

if (Int32.TryParse(Console.ReadLine(), out otvet)) { }

if (otvet < 1 || otvet > 2)

{

Console.WriteLine("Выбор элемента неверный!");

return;

}

Console.WriteLine("Название:");

string name = Console.ReadLine();

Console.WriteLine("Содержимое буфера:");

string buffer = Console.ReadLine();

switch (otvet)

{

case 1:

printers.Enqueue(new Printer(name, buffer));

break;

case 2:

printers.Enqueue(new Multifunction(name, buffer));

break;

default:

return;

}

Console.WriteLine($"элемент с именем {name} добавлен в конец очереди!");

Console.ReadKey();

}

static void method3(Queue<Printer> printers)

{

Console.WriteLine("просмотр первого элемента в очереди\n----------------------");

Console.Write("первый элемент - ");

printers.Peek().Info();

Console.ReadKey();

}

static void method4(Queue<Printer> printers)

{

Console.WriteLine("извлечение первого элемента из очереди\n----------------------");

Console.Write("извлекли первый элемент - ");

printers.Dequeue().Info();

Console.ReadKey();

}

static void method5(Queue<Printer> printers)

{

Console.WriteLine("количество элементов в очереди\n----------------------");

Console.Write("количество = "+ printers.Count);

Console.ReadKey();

}

}

}

**3. Обобщенный метод**

Создайте обобщенный метод, который получает массив произвольного типа и возвращает количество элементов, не равных null.

**4. LINQ с массивом**

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 (лямбда-выражение).

Варианты:

1. Вычислить сумму всех чисел в диапазоне от 10 до 20.
2. Определить максимум среди всех отрицательных чисел.
3. Определить количество чисел, кратных 3.
4. Найти среднее значение среди отрицательных чисел.
5. Определить количество четных положительных чисел.
6. Определить минимум среди четных положительных чисел.
7. Вычислить сумму всех отрицательных чисел.
8. Определить количество чисел в диапазоне от -5 до 5.
9. Вычислить произведение минимального и максимального чисел.
10. Найти максимальное число, кратное 7.

**5. LINQ с 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.

1. Список сотрудников, которые старше их компаний. Включает имя, возраст, Company.
2. Список компаний, в которых не работает ни одного сотрудника.
3. Список должностей с названием компании и окладом, без повторений.
4. Список сотрудников, чье имя начинается с той же буквы, что и их компания. Включает имя сотрудника и имя компании.
5. Список компаний, в которых более одного сотрудника. Содержит название компании и к-во сотрудников.
6. Список лет, в которых были основаны компании. Содержит год и к-во компаний.
7. Список должностей с суммарным окладом, который получают все сотрудники на этой должности в этой компании. Содержит компанию, должность, сумму.
8. Список лет, от минимального, до максимального, в который не было основано ни одной компании и не родился ни один из сотрудников.
9. Сумму окладов по каждой компании.
10. Максимальный возраст сотрудника для каждой компании.

class Company

{

public string Name { get; set; }

public int YearOfFoundation { get; set; }

public Company(string name, int year)

{

Name = name;

YearOfFoundation = year;

}

}

class Person

{

public string Name { get; set; }

public int YearOfBirth { get; set; }

public string Position { get; set; }

public int Salary { get; set; }

public Company Company { get; set; }

// конструкторы

public Person() : this("noname", new Company("noname",1900)) { }

public Person(string name, Company company) : this(name, 1900, "noposition", 0, company) { }

public Person(string name, int year, string position, int salary, Company company)

{

Name = name;

YearOfBirth = year;

Position = position;

Salary = salary;

Company = company;

}

public void Info()

{

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

}

}

class Program

{

// 3. Обобщенный метод

static int QuantityNull<T>(T[] array)

{

int result = 0;

foreach (T elem in array)

if (elem != null)

result++;

return result;

}

static void Main(string[] args)

{

// 3. Обобщенный метод

Printer[] printers = new Printer[]

{

new Printer("Kyocera","qwerty"),

null,

new Multifunction("Xerox", "yesterday"),

null,

new Printer("Samsung","good morning"),

new Printer("Epson","bye"),

new Multifunction("Canon", "togetho"),

null

};

Console.WriteLine("! обобщенный метод, который получает массив элементов типа Printer и возвращает количество элементов, не равных null");

Console.WriteLine($"количество элементов массива = {printers.Length} из них {QuantityNull<Printer>(printers)} не равны null\n");

// 4. LINQ с массивом (4-вариант. Найти среднее значение среди отрицательных чисел.)

int[] array = new int[30];

Random random = new Random();

Console.WriteLine("значения массива:");

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

{

array[i] = random.Next(-99, 99);

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

}

var avg = (from a in array where a < 0 select a).Average();

Console.WriteLine($"\n\n(LINQ)среднее значение среди отрицательных = {avg.ToString()}");

string average = array.Where(p => (p < 0)).Average().ToString();

Console.WriteLine($"\n(Метод расширения LINQ)среднее значение среди отрицательных = {average}");

// 5. LINQ с 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 - список сотрудников, чье имя начинается с той же буквы, что и их компания. Включает имя сотрудника и имя компании.

Console.WriteLine("\nLINQ с коллекцией");

Company Microsoft = new Company("Майкрософт", 1976);

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

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

{

new Person("Андрей",Microsoft),

new Person("Михаил",Microsoft),

new Person("Вячеслав",Microsoft),

new Person("Григорий",Google),

new Person("Виктор",Google),

new Person("Глеб",Google)

};

Console.WriteLine("Коллекция people:");

foreach (Person p in people)

p.Info();

var peop = people.Where(p => p.Name.First() == p.Company.Name.First()).Select(o => new { Name = o.Name, Company = o.Company.Name });

Console.WriteLine("\nНовая коллекция - список сотрудников, чье имя начинается с той же буквы, что и их компания.\nВключает имя сотрудника и имя компании:");

foreach (var pers in peop)

Console.WriteLine($"Имя:{pers.Name} Компания:{pers.Company}");

Console.ReadKey();

}

}

