Лабораторная работа №11.1

Тема: «Разработка программ с использованием интерфейсов».(IComparable, IComparer)

Цель работы: Сформировать умения работать с классами и объектами через интерфейсы.

Время на выполнение работы: 4 часа

Этапы работы:

I. Ознакомиться с теоретическими сведениями.

II. Выполнить задания, предложенные преподавателем.

III. Ответить на контрольные вопросы.

I. Теория

Интерфейс IComparer определен в пространстве имен System.Collections. Он содержит один метод Compare, возвращающий результат сравнения двух объектов, переданных ему в качестве параметров:

```
interface IComparer
{
    int Compare ( object ob1, object ob2 )
}
```

Принцип применения этого интерфейса состоит в том, что для каждого критерия сортировки объектов описывается небольшой вспомогательный класс, реализующий этот интерфейс. Объект этого класса передается в стандартный метод сортировки массива в качестве второго аргумента.

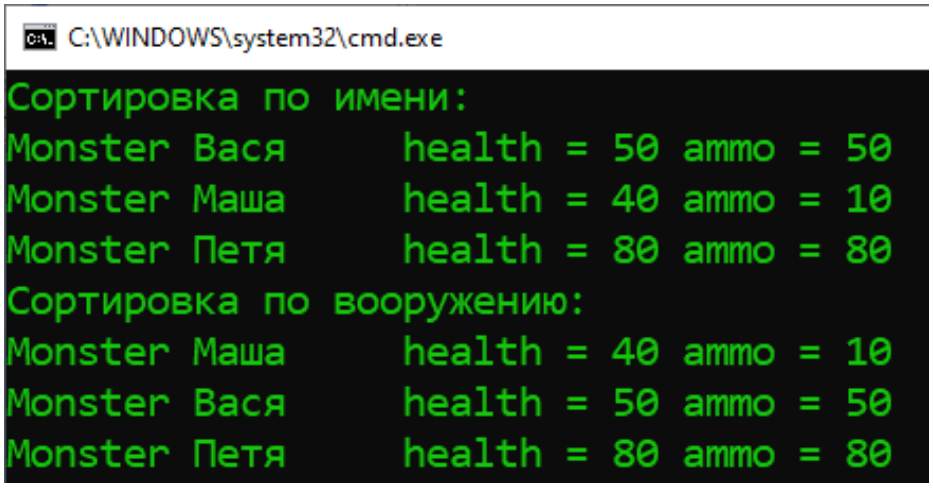
Пример сортировки массива объектов по именам (свойство Name, класс SortByName) и количеству вооружений (свойство Ammo, класс SortByAmmo) приведен в листинге 1.

Листинг 1. Сортировка по двум критериям

```
class Monster
    {
        int health;
        int ammo;

        public int Ammo
        {
            get { return ammo; }
            set
            {
                if (value > 0) ammo = value;
                else ammo = 0;
            }
        }
```

```
        public string Name
        {
            get;
            set;
        }

        public Monster(int health, int ammo, string name)
        {
            this.health = health;
            Ammo = ammo;
            Name = name;
        }

        public override  string ToString()
        {
            return $"Monster {Name} \t health = {health}
ammo = {Ammo}";
        }
    }

public class SortByName : IComparer
//
    {
        int IComparer.Compare(object ob1, object ob2)
        {
            Monster m1 = (Monster)ob1;
            Monster m2 = (Monster)ob2;
            return string.Compare(m1.Name, m2.Name);
        }
    }


public class SortByAmmo : IComparer
//
    {
        int IComparer.Compare(object ob1, object ob2)
        {
            Monster m1 = (Monster)ob1;
            Monster m2 = (Monster)ob2;
            return m1.Ammo - m2.Ammo;
        }
    }

class Runner
```

```
    {
        static void Main()
        {
            const int n = 3;
            Monster[] monsters = new Monster[n];

            monsters[0] = new Monster(50, 50, "Вася");
            monsters[1] = new Monster(80, 80, "Петя");
            monsters[2] = new Monster(40, 10, "Маша");

            Console.WriteLine("Сортировка по имени:");
            Array.Sort(monsters, new SortByName());
            foreach (Monster elem in monsters)
                Console.WriteLine( elem);

            Console.WriteLine("Сортировка по
вооружению:");
            Array.Sort(monsters, new SortByAmmo());
            foreach (Monster elem in monsters)
                Console.WriteLine(elem);
         Console.ReadKey();
        }
    }
class Runner
    {
        static void Main()
        {
            const int n = 3;
            Monster[] monsters = new Monster[n];

            monsters[0] = new Monster(50, 50, "Вася");
            monsters[1] = new Monster(80, 80, "Петя");
            monsters[2] = new Monster(40, 10, "Маша");

            Console.WriteLine("Сортировка по имени:");
            Array.Sort(monsters);
            foreach (Monster elem in monsters)
                Console.WriteLine( elem);


            Console.ReadKey();
        }
    }
```



Интерфейс IComparable определен в пространстве имен System. Он содержит всего один метод CompareTo, возвращающий результат сравнения двух объектов — текущего и переданного ему в качестве параметра:

```
interface IComparable
{
    int CompareTo( object obj )
}
```

Метод должен возвращать:

- 0, если текущий объект и параметр равны;
- отрицательное число, если текущий объект меньше параметра;
- положительное число, если текущий объект больше параметра.

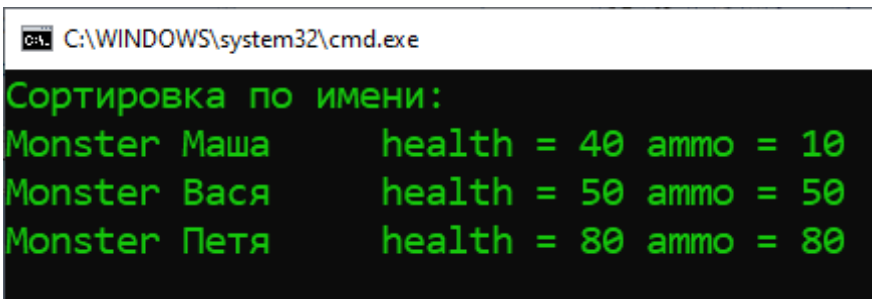
Реализуем интерфейс IComparable в знакомом нам классе Monster. В качестве критерия сравнения объектов выберем поле health. В листинге 2 приведена программа, сортирующая массив монстров по возрастанию величины, характеризующей их здоровье.

Листинг 2. Пример реализации интерфейса IComparable

```
class Monster: IComparable<Monster>
    {
        int health;
        int ammo;

        public int Ammo
        {
            get { return ammo; }
            set
            {
                if (value > 0) ammo = value;
                else ammo = 0;
            }
        }

        public string Name
```

```
        {
            get;
            set;
        }

        public Monster(int health, int ammo, string name)
        {
            this.health = health;
            Ammo = ammo;
            Name = name;
        }

        public override  string ToString()
        {
            return $"Monster {Name} \t health = {health}
ammo = {Ammo}";
        }
        public int CompareTo(Monster other)
        {
            return health - other.health;
        }
}
```

Результат работы программы:



II. Задания

1. Задание

Задание 1 необходимо выполнить через интерфейс IComparer;

Варианты:

1. Создать массив целых чисел(случайно и от руки). Отсортировать его при помощи Array.Sort так, чтобы вначале были 10, затем 8, после 6, а далее в обычном порядке. **Пример** 10, 8, 8 ,25,14,3…
2. Создать массив double (случайно и от руки). Отсортировать его при помощи Array.Sort так, чтобы вначале были числа без дробной части, а после числа были бы отсортированы по дробной части. **Пример** 10,15,22, 90.01,3.02,1.25 ….
3. Создать массив целых чисел(случайно и от руки). Отсортировать его при помощи Array.Sort так, по остаткам от деления на 3, а в пределах остатка в обычном порядке. **Пример** 3,6,9, 4,10,13, 5,8 …

4. Создать массив char (только буквы)(случайно и от руки). Отсортировать его при помощи Array.Sort так, чтобы вначале были гласные в алфавитном порядке, а после согласные в алфавитном порядке. **Пример** a, i, b,c,f…

5. Создать массив String (от руки). Отсортировать его при помощи Array.Sort так, чтобы вначале были слова с наименьшим количеством букв.

6. Создать массив String (от руки). Отсортировать его при помощи Array.Sort так, чтобы вначале были слова с наибольшим количеством гласных букв.

7. Создать массив String(от руки). Отсортировать его при помощи Array.Sort так, чтобы вначале были слова из 3х букв, затем из 5 ти букв, затем из 6ти букв, а дальше все по алфавиту.

2. Задание

Задание 2 необходимо выполнить через интерфейс IComparable; Создать массив объектов и отсортировать его при помощи Array.Sort.

Вариант 1

Создать класс MyArray (массив типа int). В классе описать следующие элементы:

- поля – размерность;
- конструкторы без параметров с параметрами;
- метод сравнения двух объектов (сравнивать поэлементно до первого расхождения);
- метод вывода массива на экран в виде “{$a_1, a_2, a_3, ..., a_n$}” (ToString())

Вариант 2

Создать класс Rectangle (прямоугольник). В классе описать следующие элементы:

- поля – длины сторон;
- конструкторы без параметров с параметрами;
- **свойства** для установки и получения значений всех сторон;
- метод сравнения двух объектов (сравнивать прямоугольники по их площадям);
- метод вывода длин треугольника на экран в виде (a, b) (ToString()).

Вариант 3

Создать класс Complex (комплексное число). В классе описать следующие элементы:

- поля – мнимая и действительная части;
- конструкторы без параметров с параметрами;

- **свойства** для установки и получения значений всех координат;
- метод сравнения двух объектов (сравнивать действительные части, а при совпадении мнимые);
- метод вывода комплексного числа на экран в виде a+ib (ToString()).

Вариант 4

Создать класс MyMatrix (матрица типа int). В классе описать следующие элементы:

- поля – размерность (может быть не квадратной);
- конструкторы без параметров с параметрами;;
- метод сравнения двух объектов (сравнивать по сумме диагональных элементов матрицы);
- метод вывода матрицы на экран в виде таблицы (ToString()).

Вариант 5

Создать класс Triangle (треугольник). В классе описать следующие элементы:

- длины сторон;
- конструкторы без параметров с параметрами; (проверка на создание треунольника)
- **свойства** для установки и получения значений всех сторон;
- метод сравнения двух объектов (сравнивать треугольники по их площадям);
- метод вывода длин треугольника на экран в виде (a,b,c). (ToString())

Вариант 6

Создать класс Point (точка). В классе описать следующие элементы:

- поля – координаты по осям Ox и Oy;
- конструкторы без параметров с параметрами;;
- **свойства** для установки и получения значений всех координат;
- метод сравнения двух объектов (сравнивать расстояние от текущей точки до начала координат);
- метод вывода координат точки на экран в виде (x,y). (ToString())

Вариант 7

Создать класс Fraction (обыкновенная дробь). В классе описать следующие элементы:

- поля – числитель и знаменатель;
- конструкторы с параметрами и без; (проверка на неправильную дробь, привести к правильной)
- **свойства** для установки и получения значений всех координат;
- метод сравнения двух объектов (обычное сравнение двух дробей);

- метод вывода дроби на экран в виде n / d. (ToString()), если есть целая часть, то ее тоже написать

Вариант 8

Создать класс Circle (окружность). В классе описать следующие элементы:

- координаты центра и радиус окружности;
- конструкторы с параметрами и без;
- **свойства** для установки и получения значений всех координат;
- метод сравнения двух объектов (сравнивать окружности по их площадям);
- метод вывода окружности на экран в виде (x,y - r). ToString()

Вариант 9

Создать класс MyString (строка). В классе описать следующие элементы:

- строка и ее размерность;
- конструкторы с параметрами и без;
- **свойства** для установки и получения значений всех полей;
- метод сравнения двух объектов (сравнивать строки по количеству вхождений букв «а»);
- метод вывода строки на экран (количества букв а — строка). ToString()

Вариант 10

Создать класс RightAngledTriangle (прямоугольный треугольник). В классе описать следующие элементы:

- длины двух катетов;
- конструкторы с параметрами и без;
- **свойства** для установки и получения значений всех сторон;
- метод сравнения двух объектов (сравнивать треугольники по их гипотенузам);
- метод вывода длин треугольника на экран в виде (a,b,c).

ПРИМЕЧАНИЕ ПРЕПОДАВАТЕЛЬ МОЖЕТ ПОПРОСИТЬ ИЗМЕНИТЬ ПРОГРАММУ.

III. Контрольные вопросы

1. Чем отличается IComparer от ICompareble.
2. Как должна работать операция Compare(CompareTo).
3. Особенности интерфейсов
4. Какие члены может реализовывать интерфейс
5. Преимущество интерфейсов над абстрактными классми и недостатки