Московский государственный технический университет им. Н.Э. Баумана

Факультет «Радиотехнический»
Кафедра «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №6 «Работа с коллекциями в языке C#»

Выполнил:
студент группы РТ5-31Б:
Шарафутдинов М.Э.

Подпись и дата:

Проверил:
преподаватель кафедры ИУ5
Гапанюк Ю.Е.

Подпись и дата:

Москва, 2024 г.

Постановка задачи

Разработать программу, реализующую работу с коллекциями.

1. Программа должна быть разработана в виде консольного приложения на языке C#.
2. Создать объекты классов «Прямоугольник», «Квадрат», «Круг».
3. Для реализации возможности сортировки геометрических фигур для класса «Геометрическая фигура» добавить реализацию интерфейса IComparable. Сортировка производится по площади фигуры.
4. Создать коллекцию класса ArrayList. Сохранить объекты в коллекцию. Отсортировать коллекцию. Вывести в цикле содержимое коллекции.
5. Создать коллекцию класса List<Figure>. Сохранить объекты в коллекцию. Отсортировать коллекцию. Вывести в цикле содержимое коллекции.
6. Модифицировать класс разреженной матрицы (проект SparseMatrix) для работы с тремя измерениями – x,y,z. Вывод элементов в методе ToString() осуществлять в том виде, который Вы считаете наиболее удобным. Разработать пример использования разреженной матрицы для геометрических фигур.
7. Реализовать класс «SimpleStack» на основе односвязного списка. Класс SimpleStack наследуется от класса SimpleList (проект SimpleListProject). Необходимо добавить в класс методы:
• public void Push(T element) – добавление в стек;
• public T Pop() – чтение с удалением из стека.
8. Пример работы класса SimpleStack реализовать на основе геометрических фигур.

Текст программы

Файл Program.cs

```
using System.Collections;

namespace class_lab3
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                #region создание геометрических фигур
                Rectangle myRect = new Rectangle(2, 5);
                myRect.Print();
                Square mySquare = new Square(5);
                mySquare.Print();
                Circle myCircle = new Circle(5);
                myCircle.Print();
```

```
            #endregion

            #region создание коллекции класса ArrayList
            Console.WriteLine("\nСоздание коллекции ArrayList и сортировка");
            ArrayList shapesArr = [myCircle, myRect, mySquare];
            shapesArr.Sort();
            foreach (object obj in shapesArr)
            {
                if (obj is geometricShape)
                {
                    geometricShape res = (geometricShape)obj;
                    res.Print();
                }
            }
            #endregion

            #region создание коллекции класса List
            Console.WriteLine("\nСоздание коллекции List и сортировка");
            List<geometricShape> shapesList = [myRect, myCircle, mySquare];
            shapesList.Sort();
            foreach (geometricShape gs in shapesList)
            {
                gs.Print();
            }
            #endregion

            #region работа с матрицей
            SparseMatrix MyMatrix = new SparseMatrix(5, 5);
            MyMatrix.AddElement(90, 50, 2);
            MyMatrix.AddElement(1, 1, 2);
            MyMatrix.AddElement(1, 4, 3);
            MyMatrix.AddElement(5, 5, 8);

            MyMatrix.PrintElements();
            Console.WriteLine(MyMatrix.ToString());
            #endregion

            #region работа со стеком
            var myStack = new SimpleStack<geometricShape>();
            myStack.PrintElements();
            myStack.Push(myRect);
            myStack.Push(mySquare);
            myStack.Push(myCircle);

            myStack.PrintElements();
            myStack.Pop();
            myStack.PrintElements();
            #endregion
        }
        catch (Exception ex) {Console.WriteLine("Ошибка: " + ex.Message);}
    }
```

```
    }
}
```

Файл geometricShape.cs

```
using System.ComponentModel.Design;

namespace class_lab3
{
    abstract class geometricShape: IPrint, IComparable
    {
        public abstract double Area();
        public abstract string ToString();
        public void Print()
        {
            Console.WriteLine(this.ToString());
        }
        public int CompareTo(object? obj)
        {
            if (obj is geometricShape shape) return
Area().CompareTo(shape.Area());
            else throw new ArgumentException("бла бла бла: ");
        }
    }
}
```

Файл IPrint.cs

```
namespace class_lab3
{
    interface IPrint
    {
        public void Print();
    }
}
```

Файл rectangle.cs

```
namespace class_lab3
{
    class Rectangle : geometricShape, IPrint
    {
        public Rectangle(double h, double w)
        {
            this.Height = h;
            this.Width = w;
        }
        public double Height { get; set; }
        public double Width { get; set; }
        public override double Area()
        {
            return Height * Width;
```

```
        }
        public override string ToString()
        {
            return ("Прямоугольник со сторонами " + Height.ToString() + " и " +
Width.ToString() + "; площадь: " + this.Area());
        }
    }
}
```

Файл square.cs

```
namespace class_lab3
{
    class Square : Rectangle
    {
        public Square(double a) : base(a, a) {}
        public override string ToString()
        {
            return ("Квадрат со стороной " + Height.ToString() + "; площадь: " +
this.Area());
        }
    }
}
```

Файл circle.cs

```
namespace class_lab3
{
    class Circle : geometricShape
    {
        public Circle(double R)
        {
            Radius = R;
        }
        public double Radius { get; set; }
        public override double Area()
        {
            return Math.PI * Radius * Radius;
        }
        public override string ToString()
        {
            return ("Круг с радиусом " + Radius.ToString() + "; площадь: " +
this.Area());
        }
    }
}
```

Файл SparseMatrix.cs

```
namespace class_lab3
{
    class MatrixElement : IComparable
```

```
    {
        public MatrixElement(int x, int y, int elm)
        {
            X = x;
            Y = y;
            Value = elm;
        }
        public int X { get; set; }
        public int Y { get; set; }
        public int Value { get; set; }
        public int CompareTo(object? obj)
        {
            if (obj is MatrixElement elm1)
            {
                if (this.Y > elm1.Y) return 1;
                else if (this.Y < elm1.Y) { return -1; }
                else if (this.X == elm1.X) { throw new ArgumentException("Разные
элементы с совпадающими позициями"); }
                else return this.X.CompareTo(elm1.X);
            }
            else throw new ArgumentException("Некорректный тип элемента");
        }

    }

    class SparseMatrix
    {
        public SparseMatrix(int X, int Y)
        {
            dimX = X;
            dimY = Y;
            Elements = new List<MatrixElement>();
        }
        public int dimX { get; set; }
        public int dimY { get; set; }
        public List<MatrixElement> Elements { get; set; }
        public void AddElement(int x, int y, int val)
        {
            if (x < 0 || y < 0 || (x > this.dimX) || (y > this.dimY))
                Console.WriteLine("Элемент выходит за пределы матрицы");
            else
            {
                Elements.Add(new MatrixElement(x, y, val));
                Elements.Sort();
            }
        }
        public void PrintElements()
        {
            Console.WriteLine("\nМатрица " + this.dimY + " X " + this.dimX);
            foreach (MatrixElement elm in Elements)
            {
```

```
                Console.WriteLine("Элемент на позиции X = " + elm.X + ", Y = " +
elm.Y + " имеет значение " + elm.Value);
            }
        }
        public override string ToString()
        {
            string res = "";
            int jNum = 0;
            for (int j = 0; j < this.dimY; j++)
            {
                for (int i = 0; i < this.dimX; i++)
                {
                    if (Elements.Count == 0 || Elements.Count <= jNum ||
!(Elements[jNum].X == i + 1 && Elements[jNum].Y == j + 1)) res = res + "0 ";
                    else
                    {
                        res = res + Elements[jNum].Value + " ";
                        jNum++;
                    }
                }
                res = res + "\n";
            }
            return res;
        }
    }
}
```

Файл Stack.cs

```
namespace class_lab3
{
    class ElementOfList<Typ>
    {
        public ElementOfList(Typ val)
        {
            Value = val;
            Next = null;
        }
        public Typ Value { get; set; }
        public ElementOfList<Typ>? Next { get; set; }
    }

    class SimpleStack<Typ>
    {
        public SimpleStack()
        {
            Element = null;
        }
        public ElementOfList<Typ>? Element { get; set; }
        public void Push(Typ elem)
        {
```

```
            ElementOfList<Typ> tempElem = new ElementOfList<Typ>(elem);
            if (Element == null)
            {
                Element = tempElem;
            }
            else
            {
                tempElem.Next = Element;
                Element = tempElem;
            }
        }

        public void Pop()
        {
            if (Element != null)
            {
                Element = Element.Next;
            }
        }

        public void PrintElements()
        {
            var temp = Element;
            Console.WriteLine("Стек: ");
            if (temp == null) Console.WriteLine("Стек пуст");
            else
            {
                while (temp != null)
                {
                    Console.WriteLine(temp.Value.ToString());
                    temp = temp.Next;
                }
            }
        }
    }
}
```

Результат выполнения программы

```
Прямоугольник со сторонами 2 и 5; площадь: 10
Квадрат со стороной 5; площадь: 25
Круг с радиусом 5; площадь: 78,53981633974483

Создание коллекции ArrayList и сортировка
Прямоугольник со сторонами 2 и 5; площадь: 10
Квадрат со стороной 5; площадь: 25
Круг с радиусом 5; площадь: 78,53981633974483

Создание коллекции List и сортировка
Прямоугольник со сторонами 2 и 5; площадь: 10
Квадрат со стороной 5; площадь: 25
Круг с радиусом 5; площадь: 78,53981633974483
Элемент выходит за пределы матрицы

Матрица 5 X 5
Элемент на позиции X = 1, Y = 1 имеет значение 2
Элемент на позиции X = 1, Y = 4 имеет значение 3
Элемент на позиции X = 5, Y = 5 имеет значение 8
2 0 0 0 0
0 0 0 0 0
0 0 0 0 0
3 0 0 0 0
0 0 0 0 8

Стек:
Стек пуст
Стек:
class_lab3.Circle
class_lab3.Square
class_lab3.Rectangle
Стек:
class_lab3.Square
class_lab3.Rectangle
```