**Отчет по лабораторной работе № 3**

**по курсу «Базовые компоненты интернет технологий»**

Выполнил:

студент группы ИУ5-33

Кондратьев Максим

Дата: 01.11.2017

**Описание задания лабораторной работы:**

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 Matrix (представлен в разделе «Вспомогательные материалы для выполнения лабораторных работ») 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. КлассSimpleStack наследуется от класса SimpleList (представлен в разделе 8 «Вспомогательные материалы для выполнения лабораторных работ»). 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 на языке C#.**

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Collections;

namespace Лаб3

{

class Program

{

static void Main(string[] args)

{

Rectangle rect = new Rectangle(10, 20);

Square sqr = new Square(12);

Circle circle = new Circle(4);

//------------------------------------------------------------

ArrayList list = new ArrayList();

list.Add(rect); list.Add(sqr); list.Add(circle);

list.Sort();

foreach (Figure obj in list)

{

System.Console.WriteLine(obj.GetType().Name);

obj.Print();

}

Console.WriteLine("-------------------------------------------");

List<Figure> list2 = new List<Figure>();

list2.Add(rect); list2.Add(sqr); list2.Add(circle);

list2.Sort();

foreach (Figure obj in list2)

{

System.Console.WriteLine(obj.GetType().Name);

obj.Print();

}

Console.WriteLine("-------------------------------------------");

SparseMatrix.Matrix<Figure> drop = new SparseMatrix.Matrix<Figure>(2, 2, 1, new Circle(0));

drop[0, 0, 0] = new Circle(1);

drop[1, 0, 0] = new Square(2);

drop[1, 1, 0] = new Rectangle(1, 2);

System.Console.WriteLine(drop.ToString());

Console.WriteLine("-------------------------------------------");

SimpleStack<Figure> primer = new SimpleStack<Figure>();

primer.Add(circle); primer.Add(rect); primer.Add(sqr);

primer.Sort();

for (int i = primer.Count; i > 0; i--)

{

System.Console.WriteLine(primer.Pop());

}

System.Console.ReadLine();

}

}

}

SimpleList.cs и SparseMatrix.cs согласно методичке.

MYClasses.cs

using System;

abstract class Figure : IPrint, IComparable

{

abstract public double find\_square();

public void Print()

{

System.Console.WriteLine(this.ToString());

}

public int CompareTo(Object rhs)

{

Figure obj = rhs as Figure;

if (this.find\_square() - obj.find\_square() > 0.0001)

{

if (this.find\_square() < obj.find\_square())

{

return -1;

}

else

{

return 1;

}

}

else

{

return 0;

}

}

}

class Rectangle : Figure, IPrint

{

public double width { get; set; }

public double heigth { get; set; }

public Rectangle(double width1, double heigth1)

{

heigth = heigth1; width = width1;

}

public override double find\_square()

{

return heigth \* width;

}

public override string ToString() => "Прямоугольник "+width.ToString() + " на " + heigth.ToString();

}

class Square : Rectangle, IPrint

{

public Square(double heigth1) : base(heigth1, heigth1)

{ }

public override string ToString() => "Квадрат стороной " + heigth.ToString();

}

class Circle : Figure, IPrint

{

public double radius { get; set; }

public Circle(double radius1)

{

radius = radius1;

}

public override double find\_square()

{

return (Math.PI \* Math.Pow(radius, 2));

}

public override string ToString() => "Круг радиусом " + radius.ToString();

}

interface IPrint

{

void Print();

}

class SimpleStack<T> : SimpleList.SimpleList<T> where T : IComparable

{

public void Push(T element)

{

Add(element);

}

public T Pop()

{

SimpleList.SimpleListItem<T> itemPopped = last;

Count = Count - 1;

if (Count == 0)

{

last = null;

first = null;

}

else

{

SimpleList.SimpleListItem<T> newLastItem = this.GetItem(Count - 1);

newLastItem.next = null;

last = newLastItem;

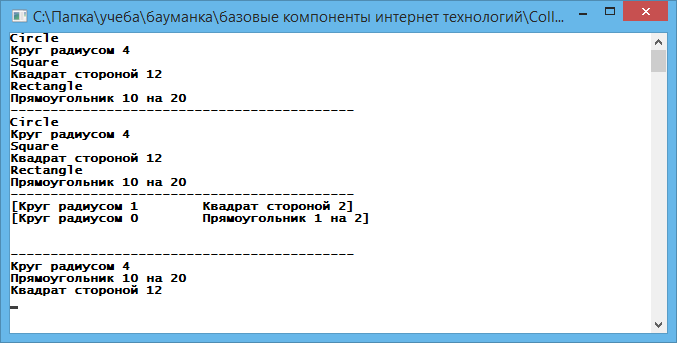
}

return itemPopped.data;

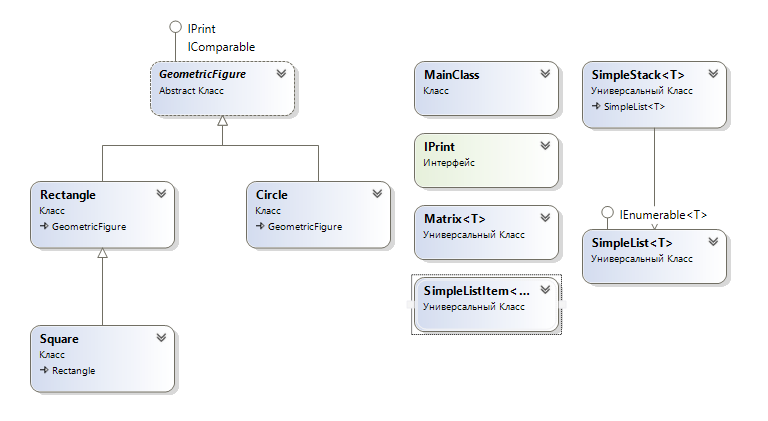
}

}

**Результаты выполнения программы:**



**Диаграмма классов:**

****