МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»

Кафедра автоматизированных систем обработки информации и управления

Отчет по лабораторной работе № 8

по дисциплине «Технологии программирования»

Тема: «Перегрузка операторов»

Выполнил: Стычинский С.В., МВИ-121

Проверили: Кузьмина Т. М. и Адаев Р. Б.

Москва 2023

Цель работы

Изучение перегрузки операторов для объектов пользовательских классов.

Этапы выполнения лабораторной работы

1. Описать пользовательский класс, заданный вариантом задания.

2. В пользовательском классе, используя статические методы, выполнить перегрузку указанных вариантом операторов.

3. Создать 2 объекта пользовательского класса. Значения полей этих объектов, а также значения дополнительных переменных должны задаваться пользователем.

4. На форму поместить кнопки, при нажатии на которые должны выполняться перегруженные операторы.

5. Результаты выполнения перегруженных операторов должны выводится на экран.

Задание

5. Равнобедренный треугольник

a. \* (а\*с) умножение высот, возвращаемое значение - число.

b. + (а+с) треугольник у которого площадь больше, тип возвращаемого значения совпадает с типом операндов.

c. Треугольник больше, если его высота больше, меньше, если высота меньше.

d. / (а/к) треугольник у которого стороны уменьшены в к раз, тип возвращаемого значения совпадает с типом первого операнда.

Теоретический материал

Перегрузка операторов

В С# существует ряд операторов для работы со встроенными типами данных. Это операторы «+», «-», «!», «==», «!=» и т.д. Например, бинарный оператор «+» выполняет операцию сложения над численными типами данных. Этот же самый оператор над строками выполняет конкатенацию (склеивание двух строк). Это происходит потому, что оператор «+» перегружен в классе string. Перегрузка оператора - это реализация своего собственного функционала оператора для конкретного класса. Перегрузка оператора похожа на перегрузку методов, но требует специального оформления. Для перегрузки операторов используется служебное слово operator. Если мы хотим перегрузить некоторый оператор, то в соответствующем классе мы должны создать общедоступный статический метод с именем operator имя\_оператора. Например operator+ или operator\*. Отметим, что при перегрузке операторов хотя бы один параметр должен иметь тип T, где T — тип, который содержит объявление оператора.

Распечатка программы

Класс Form1

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Reflection.Emit;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace lab8

{

public partial class Form1 : Form

{

private int size = 0;

public Form1()

{

InitializeComponent();

label6.Text = "";

}

private void Boxes()

{

if (listBox1.SelectedIndex != -1)

{

string str = "Объект " + size + ": " + TrList.PiList[TrList.p].getInfo();

listBox1.Items.Insert(listBox1.SelectedIndex + 1, str);

listBox2.Items.Insert(listBox2.SelectedIndex + 1, str);

size++;

}

else

{

string str = "Объект " + size + ": " + TrList.PiList[TrList.p].getInfo();

listBox1.Items.Insert(size, str);

listBox2.Items.Insert(size, str);

size++;

}

}

private void button1\_Click(object sender, EventArgs e)

{

// проверка на число

if (!int.TryParse(textBox1.Text, out \_) || !int.TryParse(textBox2.Text, out \_))

{

if (!int.TryParse(textBox1.Text, out \_))

textBox1.Text = "";

if (!int.TryParse(textBox2.Text, out \_))

textBox2.Text = "";

MessageBox.Show("Введите целые числа");

return;

}

// проверка что число >0

if (Convert.ToInt32(textBox1.Text) <= 0 || Convert.ToInt32(textBox2.Text) <= 0)

{

if (Convert.ToInt32(textBox1.Text) <= 0)

textBox1.Text = "";

if (Convert.ToInt32(textBox2.Text) <= 0)

textBox2.Text = "";

MessageBox.Show("Числа не могут быть отрицательными/равными нулю");

return;

}

//РАБОТАЕМ

RavnobedTriangle TempTriangle = new RavnobedTriangle(Convert.ToInt32(textBox1.Text), Convert.ToInt32(textBox2.Text));

TrList.Add(TempTriangle);

Boxes();

}

//Кнопка очистки, очищаем листбоксы и список

private void button2\_Click(object sender, EventArgs e)

{

listBox1.Items.Clear();

listBox2.Items.Clear();

label6.Text = "";

TrList.Clear();

size = 0;

}

//класс для проверки выбранных объектов

public bool proverka()

{

if (listBox1.SelectedIndex == -1)

{

MessageBox.Show("Выберите первый объект");

return false;

}

if (listBox2.SelectedIndex == -1)

{

MessageBox.Show("Выберите второй объект");

return false;

}

return true;

}

//+

private void button4\_Click(object sender, EventArgs e)

{

if (!proverka()) return; //проверка на выбранные объекты

if (TrList.PiList[listBox1.SelectedIndex] + TrList.PiList[listBox2.SelectedIndex] == null)

{ //в случае если треугольники одинаковые

return;

}

label6.Text = "Треугольник с большей площадью:\n" + (TrList.PiList[listBox1.SelectedIndex] + TrList.PiList[listBox2.SelectedIndex]).getInfo().ToString();

}

//\*

private void button3\_Click(object sender, EventArgs e)

{

if (!proverka()) return; //проверка на выбранные объекты

label6.Text = "Умножение высот:\n" + (TrList.PiList[listBox1.SelectedIndex] \* TrList.PiList[listBox2.SelectedIndex]).ToString();

}

//< >

private void button5\_Click(object sender, EventArgs e)

{

if (!proverka()) return; //проверка на выбранные объекты

if (TrList.PiList[listBox1.SelectedIndex] > TrList.PiList[listBox2.SelectedIndex])

{

label6.Text = "Больший треугольник:\n" + TrList.PiList[listBox1.SelectedIndex].getInfo().ToString();

}

else if (TrList.PiList[listBox1.SelectedIndex] < TrList.PiList[listBox2.SelectedIndex]) {

label6.Text = "Больший треугольник:\n" + TrList.PiList[listBox2.SelectedIndex].getInfo().ToString();

} else

{

label6.Text = "Треугольники равны";

}

}

// /

private void button6\_Click(object sender, EventArgs e)

{

//проверка на число

if (!int.TryParse(textBox3.Text, out \_))

{

MessageBox.Show("Ошибка. Введите число.");

return;

}

//проверка на выбранный объект

if (listBox1.SelectedIndex == -1)

{

MessageBox.Show("Выберите элемент из 1 списка");

return;

}

//проверка что число >0

if (Convert.ToInt32(textBox3.Text) <= 0)

{

MessageBox.Show("Число не должно быть равно нулю или быть меньше, чем 0");

return;

}

//работаем

int k = Convert.ToInt32(textBox3.Text);

RavnobedTriangle res = TrList.PiList[listBox1.SelectedIndex] / (double)k;

label6.Text = "Результат:\n"+res.getInfo();

}

}

}

Класс TrList

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab8

{

public class TrList

{

public static List<RavnobedTriangle> PiList = new List<RavnobedTriangle>();

public static int p = -1;

public static void Add(RavnobedTriangle v)

{

PiList.Insert(++p, v);

}

public static void Clear()

{

PiList.Clear();

p = -1;

}

public static int Count

{

get { return PiList.Count; }

}

}

}

Класс RavnobedTriangle

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Remoting.Messaging;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lab8

{

//Равнобедренный треугольник - вариант 5 (мой 19)

//a. \* (а\* с) умножение высот, возвращаемое значение - число.

//b. + (а+с) треугольник у которого площадь больше, тип возвращаемого значения

//совпадает с типом операндов.

//c. <,> Треугольник больше, если его высота больше, меньше, если высота меньше.

//d. / (а/к) треугольник у которого стороны уменьшены в к раз, тип возвращаемого

//значения совпадает с типом первого операнда.

public class RavnobedTriangle

{

private double height { get; set; }

private double side { get; set; }

public RavnobedTriangle(double side, double height)

{

this.side = side;

this.height = height;

}

private double S()

{

return (1.0/2)\*height \* side;

}

public string getInfo()

{

return "Высота:"+Math.Round(height,2)+"; Основание:" + Math.Round(side,2) +"; Площадь:"+S().ToString();

}

//a. \* (а\* с) умножение высот, возвращаемое значение - число.

public static double operator \*(RavnobedTriangle ob1, RavnobedTriangle ob2)

{

return ob1.height \* ob2.height;

}

//b. + (а+с) треугольник у которого площадь больше, тип возвращаемого значения совпадает с типом операндов.

public static RavnobedTriangle operator +(RavnobedTriangle ob1, RavnobedTriangle ob2)

{

double Ob1S = ((1.0 / 2) \* ob1.side \* ob1.height);

double Ob2S = ((1.0 / 2) \* ob2.side \* ob2.height);

if (Ob1S > Ob2S) {

return ob1;

}

else if (Ob1S < Ob2S)

{

return ob2;

}

else

{

MessageBox.Show("треугольники одинаковые");

return null;

}

}

//c. <,> Треугольник больше, если его высота больше, меньше, если высота меньше.

public static bool operator <(RavnobedTriangle ob1, RavnobedTriangle ob2)

{

return (ob1.height < ob2.height);

}

public static bool operator >(RavnobedTriangle ob1, RavnobedTriangle ob2)

{

return (ob1.height > ob2.height);

}

//d. / (а/к) треугольник у которого стороны уменьшены в к раз, тип возвращаемого

//значения совпадает с типом первого операнда.

public static RavnobedTriangle operator /(RavnobedTriangle ob1, double k)

{

double side = ob1.side / k;

double height = ob1.height / k;

RavnobedTriangle newOb = new RavnobedTriangle(side, height);

return newOb;

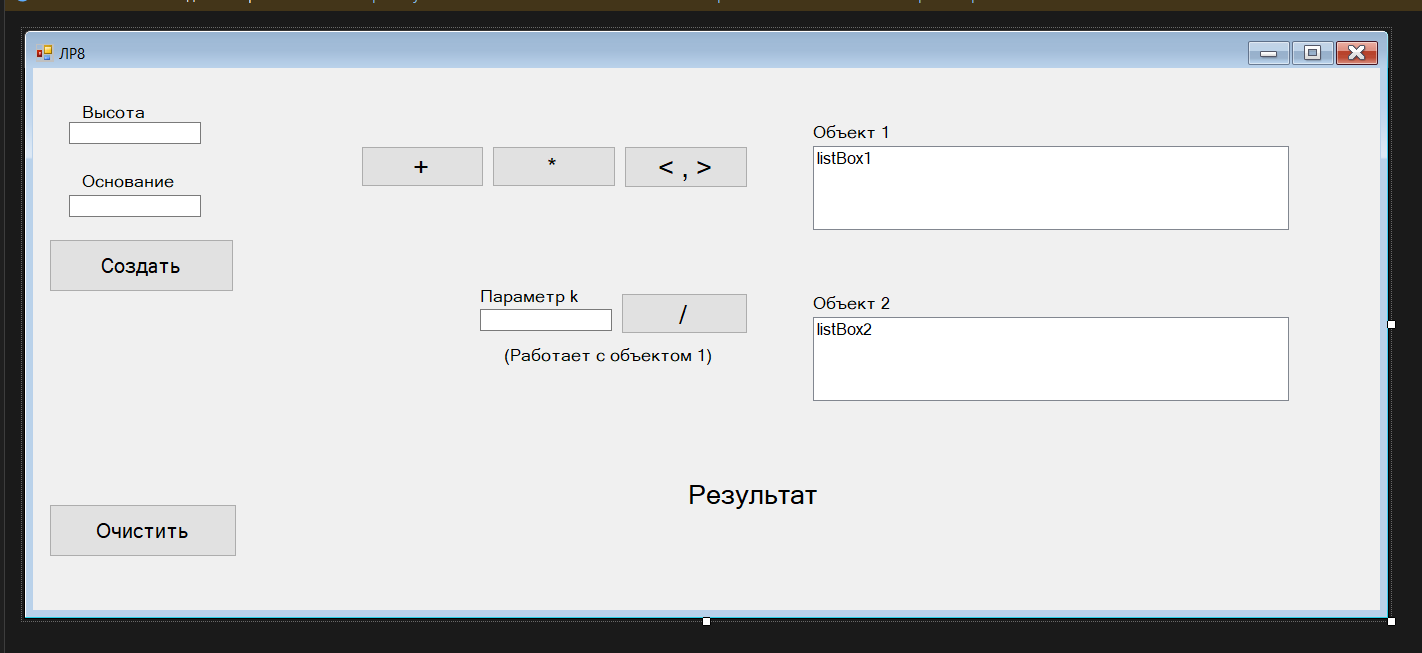
}

}

}

Результат работы программы

Форма:



Работа программы:

