

**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ (МИИТ)**

**ИНСТИТУТ ТРАНСПОРТНОЙ ТЕХНИКИ И СИСТЕМ УПРАВЛЕНИЯ**

**Лабораторная работа №2**

**по дисциплине** «**Методы программирования**»

«Конструктор ГПИ. Фигуры»

**Выполнил:** ст. гр. ТКИ-342

Речкалов Р.Ю.

Ярий А.А.

**Проверил:** к.т.н. Сафронов А.И.

**Москва – 2024 г**

# 1. Цель работы

«Закрепить навыки разработки визуального пользовательского интерфейса, освоить работу с текстовыми файлами и кодировкой в среде *Microsoft* *Visual Studio*, научиться реализовывать настройку множественных состояний объектов посредством управления компонентами со внутренней индексацией».

# 2. Формулировка задачи

«В интегрированной среде разработки *Microsoft* *Visual Studio* разработать программу в режиме *Windows Forms Application* на языке *Visual C#*, представляющую собой экранную форму, содержащую главное меню, позволяющее:

1. Начать работу с приложением.
2. Прервать работу приложения.
3. Предоставить пользователю справочную информацию о работе с приложением.

Сама программа должна реализовывать вывод в графический элемент управления (например, *PictureBox*) главной экранной формы плоскостную геометрическую фигуру, выбираемую пользователем из списка (вид списка \* задаётся вариантом индивидуального задания). Список должен обязательно содержать следующие пункты:

1. «Квадрат»,
2. «Прямоугольный треугольник»,
3. «Эллипс»,
4. «Равнобедренный треугольник»,
5. «Круг»,
6. «Равносторонний треугольник»,
7. «Окружность»,
8. «Ромб»,
9. «Трапеция»,
10. «Параллелограмм»,
11. «Прямоугольник».

Согласно заданию, нужно заменить стандартный курсор экранной формы со «стрелки» на «руку», заменить цвет подложки на HotTrack, порядок фигур должен быть следующим: ромб, прям треугольник, параллелограмм, окружность, эллипс, прямоугольник, круг, трапеция, равнобедренный треугольник, квадрат, равнобедренный треугольник. Так же должно быть сложное комбинированное изображение воздушный шар на нитке с бликом, стоящий на 4 позиции. Элементы управления должны быть организованны комбинированным списком.

# 3. Составление диаграммы классов, входящих в состав решения.

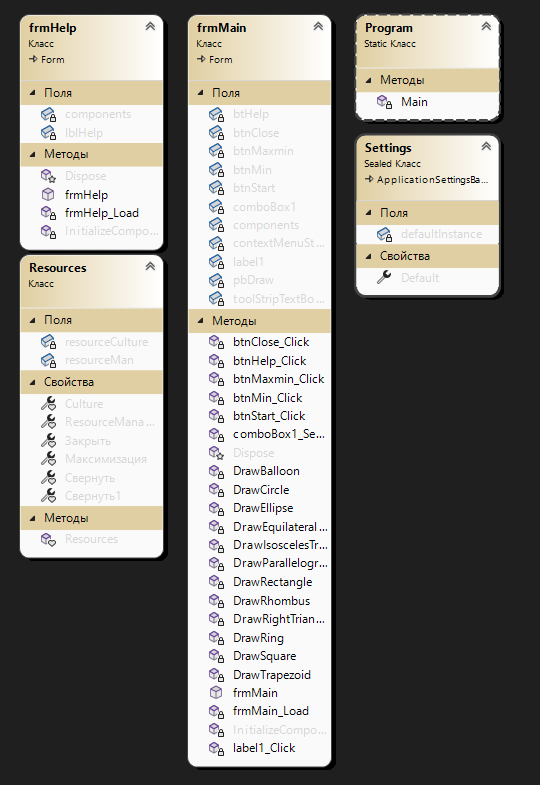


Рисунок 1 – Диаграмма классов

# 4. Составление сети Петри запрограммированного технологического процесса.

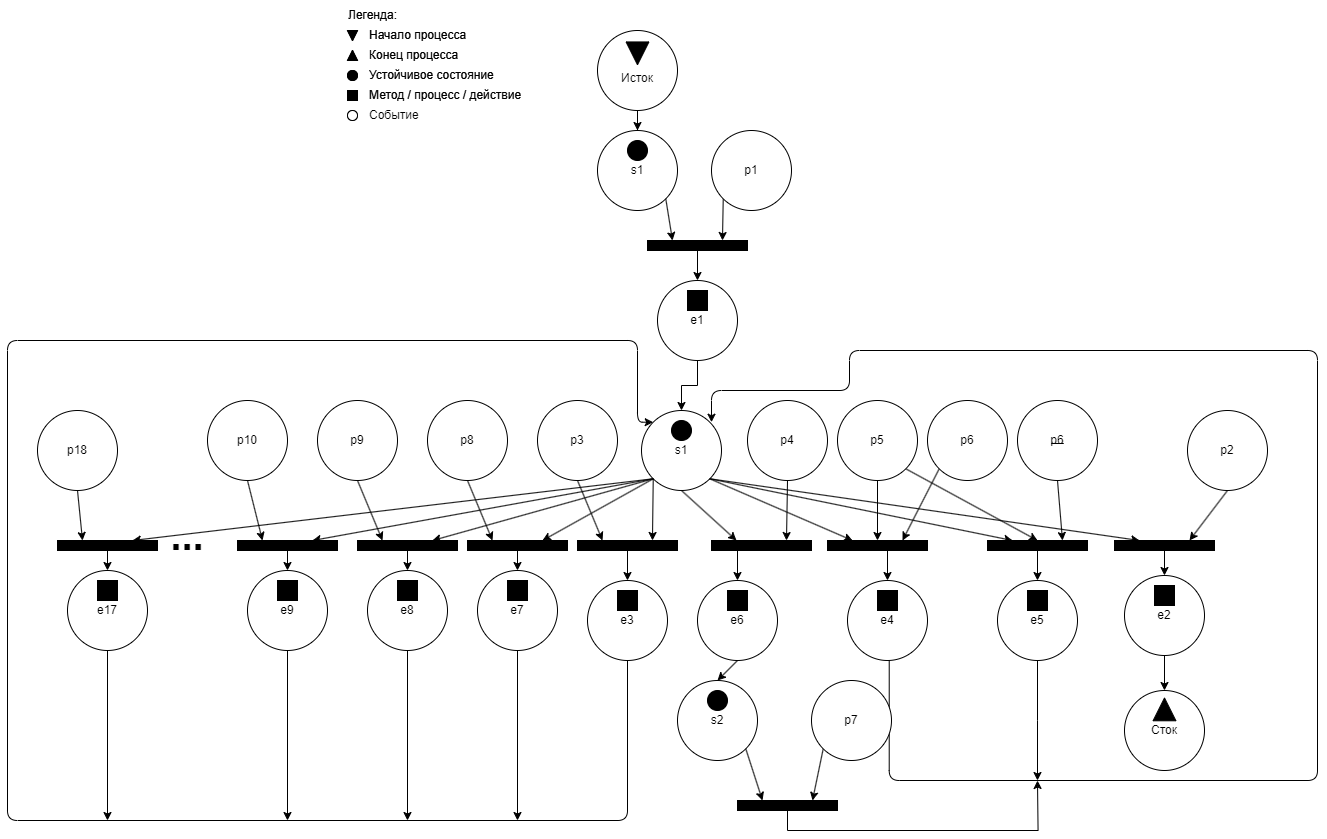


Рисунок 2 – Сеть Петри основной формы

**Описание сети Петри**

- состояния (states)

s1 – форма ожидает действий пользователя

s2 – форма находится в свернутом состоянии

- действия (effects)

e1 – разблокируются кнопки взаимодействия;

кнопка «Начать работу с приложением» деактивируется

e2 – закрытие формы

e3 – отрисовка фигуры круг

e4 – форма минимизируется

e5 – форма максимизируется

e6 – форма сворачивается

e7 – отрисовка фигуры равнобедренный треугольник

e8 – отрисовка фигуры ромб

e9 – отрисовка фигуры прямоугольный треугольник

…

e17 – отрисовка фигуры окружность

- события (prompts)

p1 – нажата кнопка «Начать работу с приложением»

p2 – нажата кнопка закрытия

p3 – нажата кнопка отрисовки фигуры круг

p4 – нажата кнопка «свернуть»

p5 – нажата кнопка «максимизировать/минимизировать»

p6 – форма максимизирована

p7 – пользователь разворачивает форму

p8 – нажата кнопка отрисовки фигуры равнобедренный треугольник

p9 – нажата кнопка отрисовки фигуры ромб

p10 – нажата кнопка отрисовки фигуры прямоугольный треугольник

…

p18 – нажата кнопка отрисовки фигуры окружность

# 5. Составление схем алгоритмов методов в составе решения, отмеченных на сети Петри в качестве «эффектов» (метка ).



Рисунок 3 – Алгоритм кнопки "Начать работу с приложением"

Алгоритм отрисовки фигуры «Равносторонний треугольник»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать полигон с координатами точек:

(200, 300 - 100 \* (float)Math.Sqrt(3)),

(100, 300),

(300, 300).

Алгоритм отрисовки фигуры «Ромб»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать полигон с координатами точек:

(200, 100),

(150, 200),

(200, 300),

(250, 200).

Алгоритм отрисовки фигуры «Прямоугольный треугольник»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать полигон с координатами точек:

(100, 100),

(100, 300),

(300, 300).

Алгоритм отрисовки фигуры «Эллипс»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать эллипс с координатами верхнего левого угла (150, 100) и размерами (100, 200).

Алгоритм отрисовки фигуры «Квадрат»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать полигон с координатами точек:

(100, 100),

(100, 300),

(300, 300),

(300, 100).

Алгоритм отрисовки фигуры «Параллелограмм»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать полигон с координатами точек:

(150, 150),

(100, 250),

(250, 250),

(300, 150).

Алгоритм отрисовки фигуры «Трапеция»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать полигон с координатами точек:

(150, 150),

(250, 150),

(300, 250),

(100, 250).

Алгоритм отрисовки фигуры «Воздушный шар»:

Создать поле размером 400х400.

Выбрать кисть с цветом красным и ручку с цветом черным.

Нарисовать линию от точки (200, 400) до (200, 250).

Нарисовать эллипс (баллон) с координатами верхнего левого угла (150, 100) и размерами (100, 150).

Выбрать кисть с цветом белым и нарисовать маленький белый эллипс (блик) с координатами (170, 120) и размерами (20, 20).

Алгоритм отрисовки фигуры «Прямоугольник»:

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать полигон с координатами точек:

(100, 150),

(100, 250),

(300, 250),

(300, 150).

Алгоритм отрисовки фигуры «Кольцо» («Окружность»):

Создать поле размером 400х400.

Выбрать ручку с цветом белым и толщиной 3.

Нарисовать эллипс с координатами верхнего левого угла (100, 100) и размерами (200, 200).

# 6. Подбор тестовых примеров.

Тестовые примеры:

Запустить программу (проверка открытия основной и дочерней формы)

Нажать кнопку «Начало работы с приложением»

Проверить корректность отрисовки каждой из фигур

Проверить кнопку «максимизировать/минимизировать»

# 7. Листинг (код) составленного программного обеспечения.

Основная форма:

using System;

using System.Drawing;

using System.Windows.Forms;

namespace Task2

{

public partial class frmMain : Form

{

public frmMain()

{

InitializeComponent();

}

private void frmMain\_Load(object sender, EventArgs e)

{

comboBox1.Items.AddRange(new string[]

{

"Отрисовать ромб",

"Отрисовать прямоугольный треугольник",

"Отрисовать параллелограмм",

"Отрисовать воздушный шар",

"Отрисовать окружность",

"Отрисовать эллипс",

"Отрисовать прямоугольник",

"Отрисовать круг",

"Отрисовать трапецию",

"Отрисовать равносторонний треугольник",

"Отрисовать квадрат",

"Отрисовать равнобедренный треугольник"

});

comboBox1.SelectedIndexChanged += comboBox1\_SelectedIndexChanged;

}

private void DrawCircle(Graphics graphics)

{

SolidBrush brush = new SolidBrush(Color.White);

graphics.FillEllipse(brush, new Rectangle(100, 100, 200, 200));

}

private void DrawEquilateralTriangle(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[3] { new PointF(200, 300 - 100 \* (float)Math.Sqrt(3)), new PointF(100, 300), new PointF(300, 300) });

}

private void DrawRhombus(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[4] { new PointF(200, 100), new PointF(150, 200), new PointF(200, 300), new PointF(250, 200) });

}

private void DrawRightTriangle(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[3] { new PointF(100, 100), new PointF(100, 300), new PointF(300, 300) });

}

private void DrawEllipse(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawEllipse(pen, 150, 100, 100, 200);

}

private void DrawSquare(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[4] { new PointF(100, 100), new PointF(100, 300), new PointF(300, 300), new PointF(300, 100) });

}

private void DrawParallelogram(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[4] { new PointF(150, 150), new PointF(100, 250), new PointF(250, 250), new PointF(300, 150) });

}

private void DrawTrapezoid(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[4] { new PointF(150, 150), new PointF(250, 150), new PointF(300, 250), new PointF(100, 250) });

}

private void DrawBalloon(Graphics graphics)

{

SolidBrush balloonBrush = new SolidBrush(Color.Red);

Pen stringPen = new Pen(Color.Black, 2);

graphics.DrawLine(stringPen, 200, 400, 200, 250);

RectangleF balloonRect = new RectangleF(150, 100, 100, 150);

graphics.FillEllipse(balloonBrush, balloonRect);

SolidBrush highlightBrush = new SolidBrush(Color.White);

RectangleF highlightRect = new RectangleF(170, 120, 20, 20);

graphics.FillEllipse(highlightBrush, highlightRect);

}

private void DrawRectangle(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[4] { new PointF(100, 150), new PointF(100, 250), new PointF(300, 250), new PointF(300, 150) });

}

private void DrawRing(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawEllipse(pen, new Rectangle(100, 100, 200, 200));

}

private void DrawIsoscelesTriangle(Graphics graphics)

{

Pen pen = new Pen(Color.White, 3);

graphics.DrawPolygon(pen, new PointF[3] { new PointF(200, 100), new PointF(150, 300), new PointF(250, 300) });

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

if (comboBox1.SelectedItem == null) return;

Bitmap bitmap = new Bitmap(400, 400, System.Drawing.Imaging.PixelFormat.Format32bppPArgb);

Graphics graphics = Graphics.FromImage(bitmap);

pbDraw.Image = null;

string selectedFigure = comboBox1.SelectedItem.ToString();

switch (selectedFigure)

{

case "Отрисовать ромб":

DrawRhombus(graphics);

break;

case "Отрисовать прямоугольный треугольник":

DrawRightTriangle(graphics);

break;

case "Отрисовать параллелограмм":

DrawParallelogram(graphics);

break;

case "Отрисовать воздушный шар":

DrawBalloon(graphics);

break;

case "Отрисовать окружность":

DrawRing(graphics);

break;

case "Отрисовать эллипс":

DrawEllipse(graphics);

break;

case "Отрисовать прямоугольник":

DrawRectangle(graphics);

break;

case "Отрисовать круг":

DrawCircle(graphics);

break;

case "Отрисовать трапецию":

DrawTrapezoid(graphics);

break;

case "Отрисовать равносторонний треугольник":

DrawEquilateralTriangle(graphics);

break;

case "Отрисовать квадрат":

DrawSquare(graphics);

break;

case "Отрисовать равнобедренный треугольник":

DrawIsoscelesTriangle(graphics);

break;

}

pbDraw.Image = bitmap;

}

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

{

this.Close();

}

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

{

if (this.WindowState == FormWindowState.Normal)

{

this.WindowState = FormWindowState.Maximized;

}

else

{

this.WindowState = FormWindowState.Normal;

}

}

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

{

this.WindowState = FormWindowState.Minimized;

}

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

{

foreach (Control control in this.Controls)

{

control.Visible = true;

}

btnStart.Visible = false;

}

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

{

frmHelp myForm = new frmHelp();

myForm.Show();

}

private void label1\_Click(object sender, EventArgs e)

{

}

}

}

Дополнительная форма:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Task2

{

public partial class frmHelp : Form

{

public frmHelp()

{

InitializeComponent();

}

private void frmHelp\_Load(object sender, EventArgs e)

{

// Получаем путь к папке, где находится исполняемый файл приложения

string basePath = AppDomain.CurrentDomain.BaseDirectory;

// Формируем относительный путь к файлу справки

string helpFilePath = Path.Combine(basePath, "Resources", "HelpInformation.txt");

// Проверяем, существует ли файл по указанному пути

if (File.Exists(helpFilePath))

{

// Читаем содержимое файла и выводим его в метку lblHelp

using (StreamReader sr = new StreamReader(helpFilePath))

{

string info = sr.ReadToEnd();

lblHelp.Text = info;

}

}

else

{

// Если файл не найден, выводим сообщение об ошибке

lblHelp.Text = "Файл справки не найден.";

}

}

}

}

# 8. Графический пользовательский интерфейс программного обеспечения и его описание.

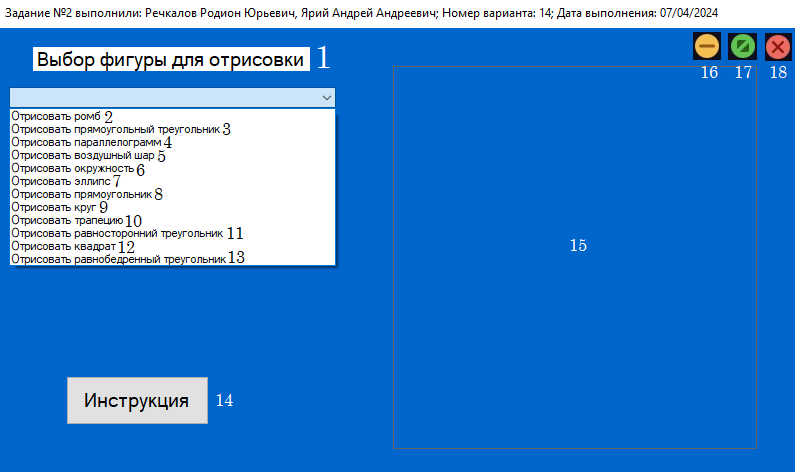


Рисунок 4 – Интерфейс основной формы

1. Текст над выпадающим списком
2. Кнопка отрисовки ромба
3. Кнопка отрисовки прямоугольного треугольника
4. Кнопка отрисовки параллелограмма
5. Кнопка отрисовки воздушного шара
6. Кнопка отрисовки окружности
7. Кнопка отрисовки эллипса
8. Кнопка отрисовки прямоугольника
9. Кнопка отрисовки круга
10. Кнопка отрисовки трапеции
11. Кнопка отрисовки равностороннего треугольника
12. Кнопка отрисовки квадрата
13. Кнопка отрисовки равнобедренного треугольника
14. Кнопка инструкции
15. Picture box
16. Кнопка для сворачивания приложения
17. Кнопка для открытия приложения во весь экран
18. Кнопка для закрытия приложения

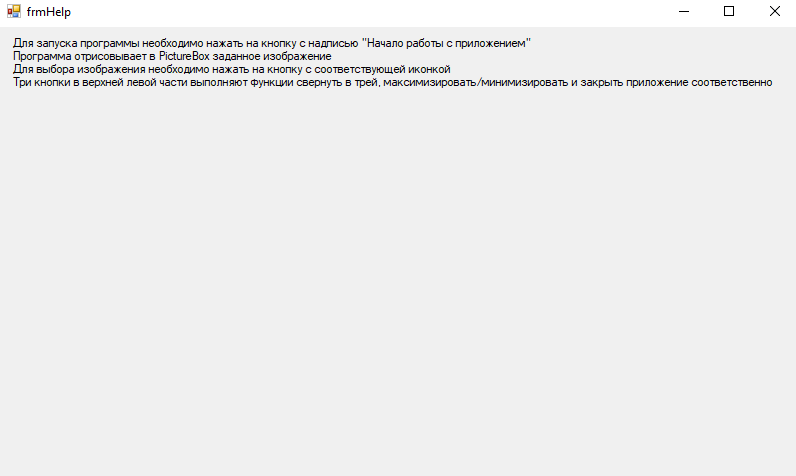


Рисунок 5 – Интерфейс вспомогательной формы

# 9. Подтверждение соответствия графического пользовательского интерфейса требованиям к оформлению.

1. Заголовок экранной формы должен содержать надпись вида: «Задание №2 выполнил: [Фамилия И.О. авторов]; Номер варианта: [Номер]; Дата выполнения: [дд/мм/гггг]».



Рисунок 6 – Название формы

2. Дата выполнения проставляется в момент, когда программа считается законченной и по ней можно готовить итоговый отчёт о выполнении работы.



Рисунок 7 – Время на момент создания законченной программы

3. Заменить стандартный курсор экранной формы со «стрелки» на «руку».



Рисунок 8 – Курсор

1. Заменить цвет подложки на HotTrack.



Рисунок 9 – Цвет подложки

1. Порядок фигур должен быть следующим: ромб, прям треугольник, параллелограмм, окружность, эллипс, прямоугольник, круг, трапеция, равнобедренный треугольник, квадрат, равнобедренный треугольник.

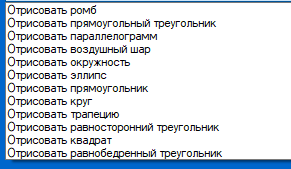


Рисунок 10 – Порядок нумерации фигур

1. Так же должно быть сложное комбинированное изображение воздушный шар на нитке с бликом, стоящий на 4 позиции.

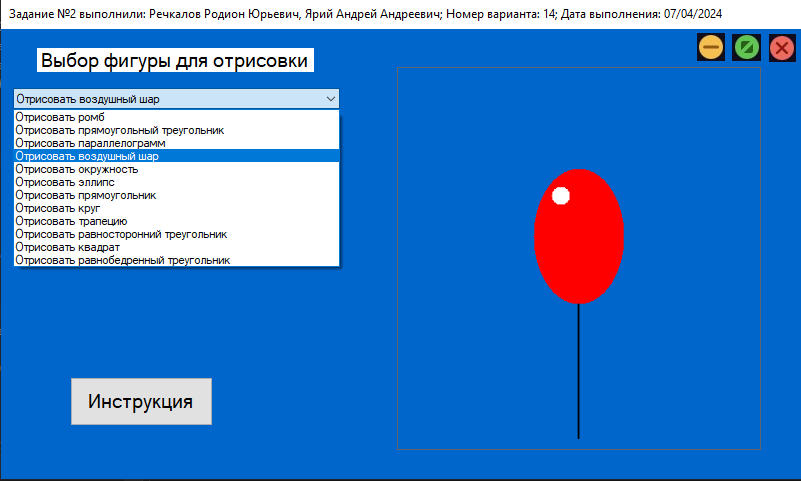


Рисунок 11 – Сложное комбинированное изображение

1. Элементы управления должны быть организованны комбинированным списком.

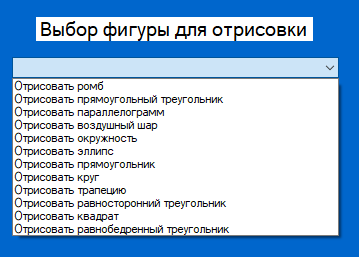


Рисунок 12 – Комбинированный список

# 10. Расчёт тестовых примеров с использованием составленного программного обеспечения.

Тестовые примеры:

Запустить программу (проверка открытия основной и дополнительной формы)

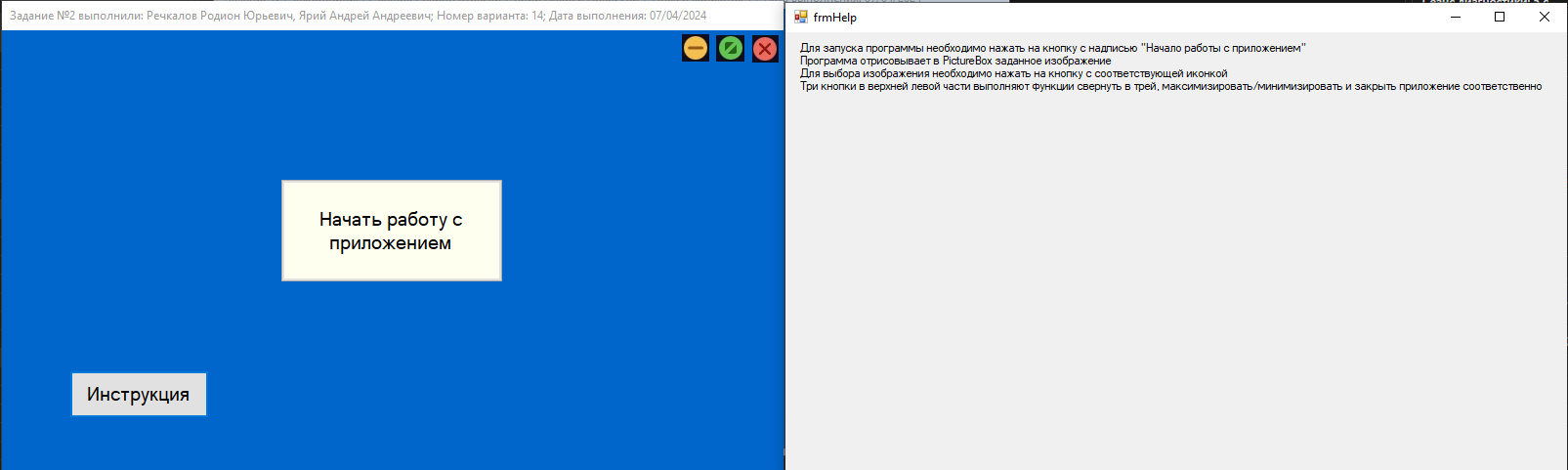


Рисунок 13 - Открытие основной и дополнительной формы

Нажать кнопку «Начать работу с приложением»

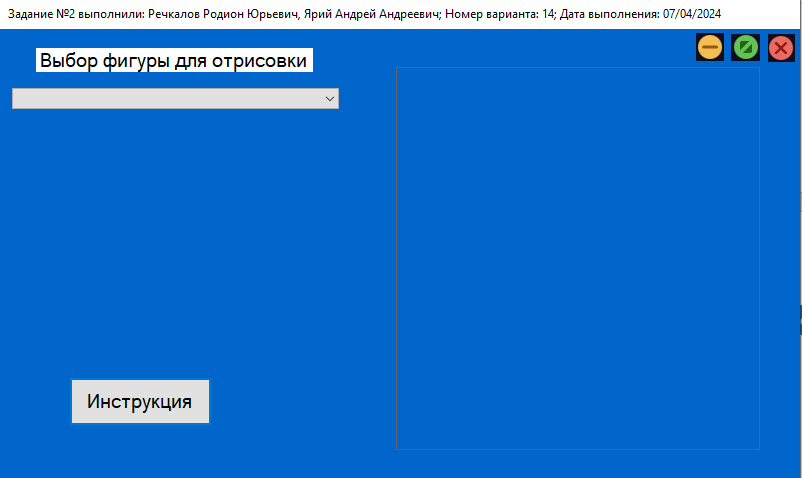


Рисунок 14 – Форма после начала работы с приложением

Проверить корректность отрисовки каждой из фигур

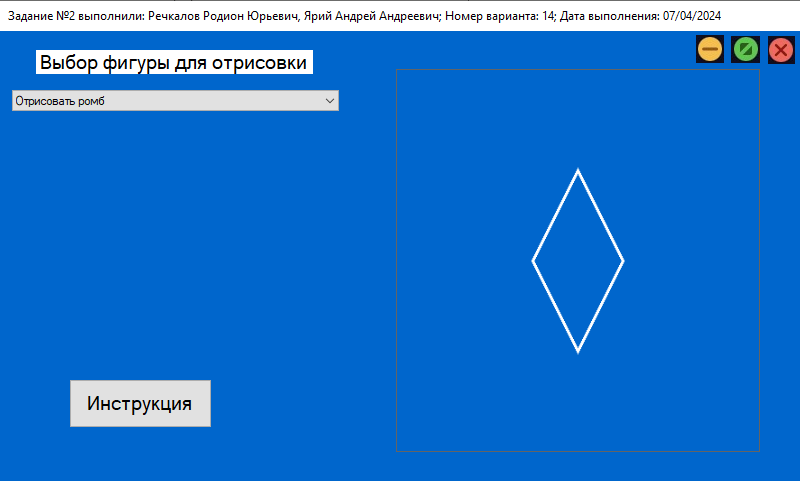


Рисунок 15 – Отрисовка ромба

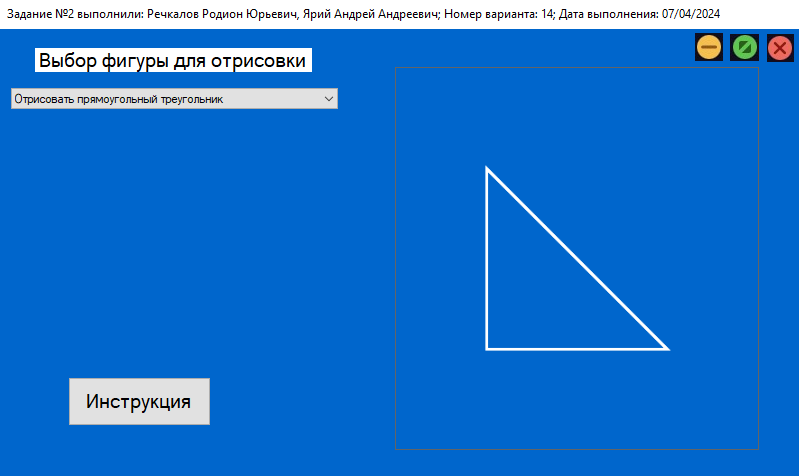


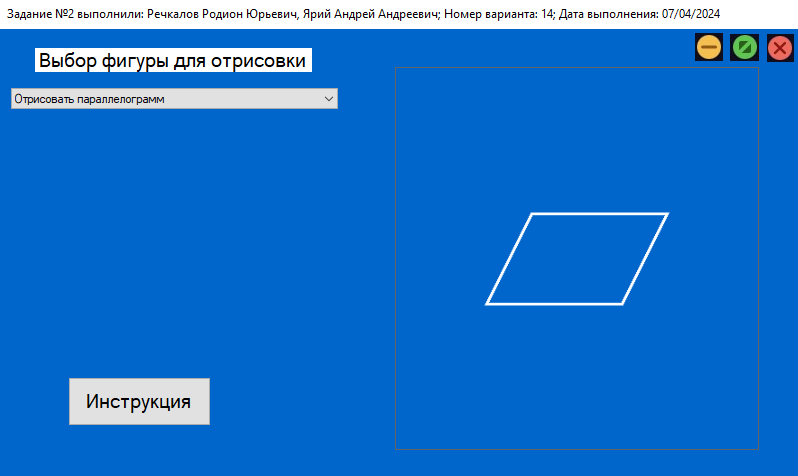
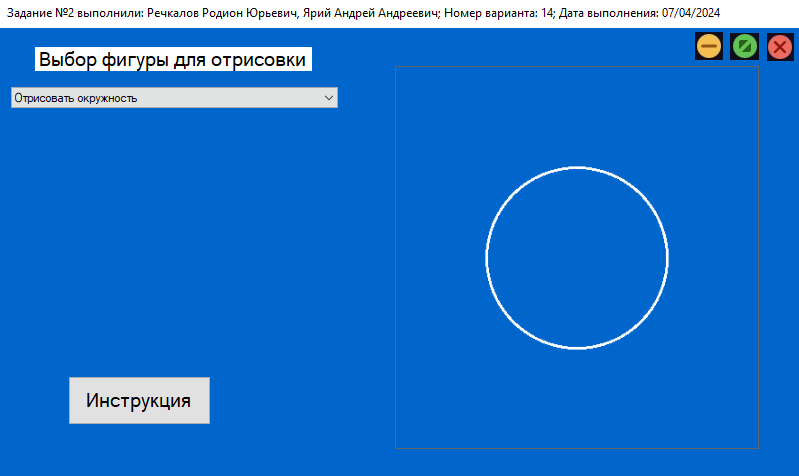
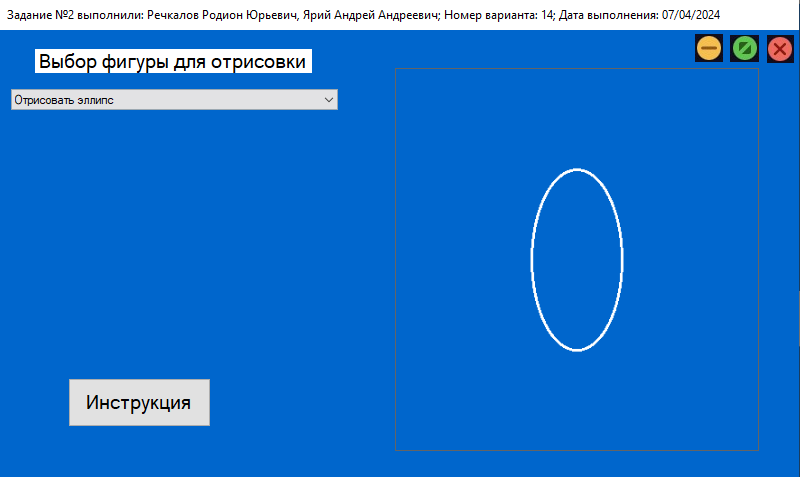
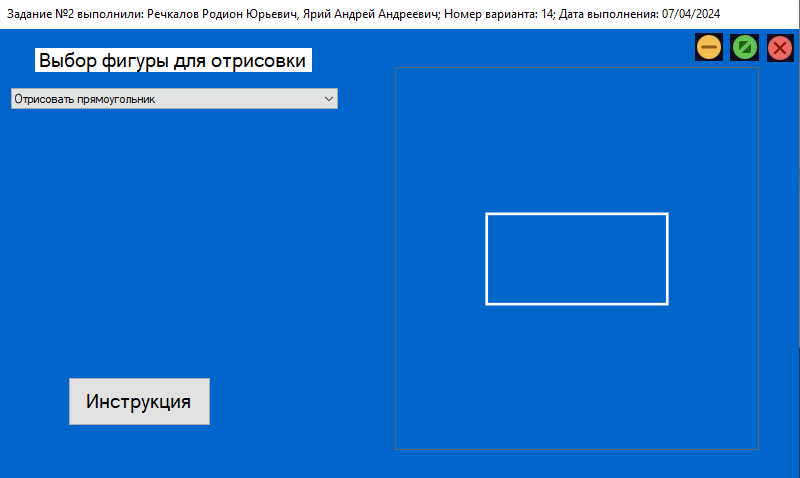
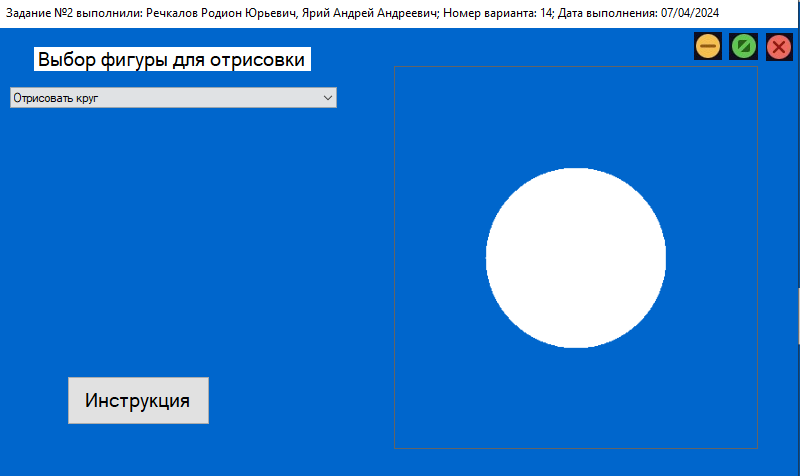
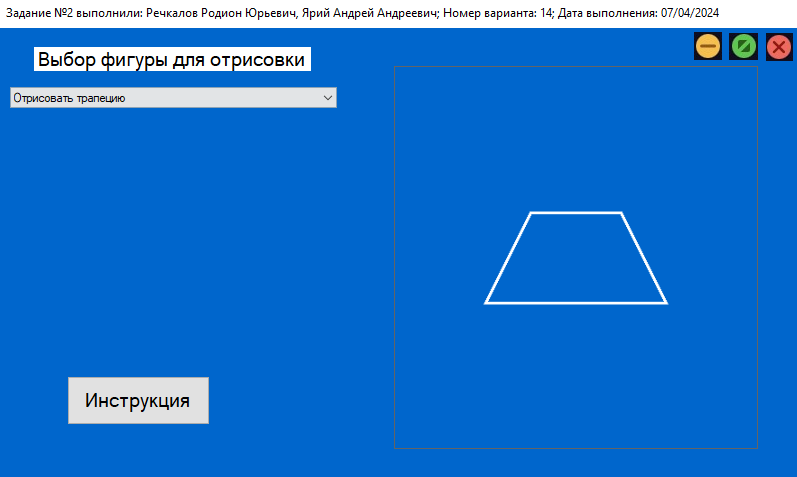
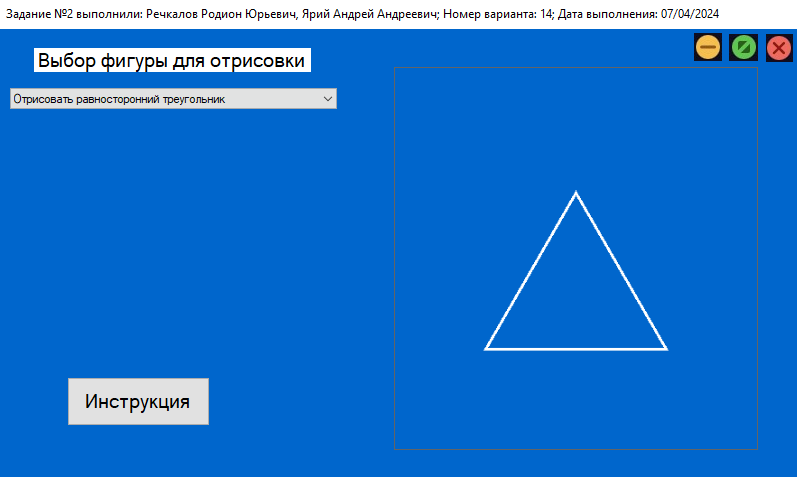
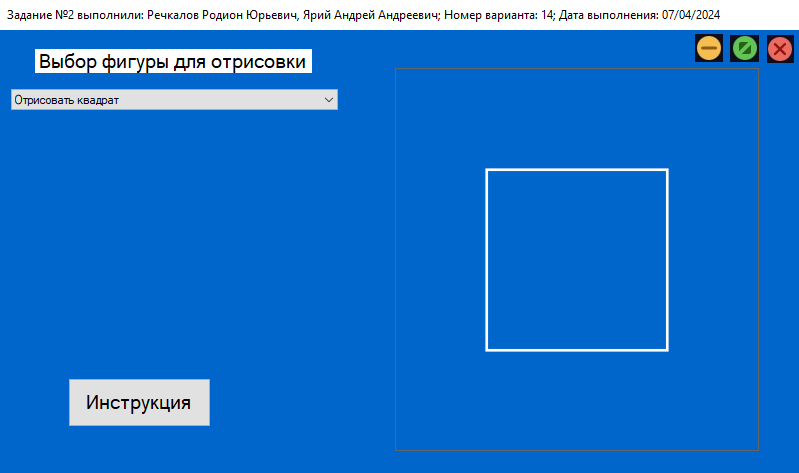
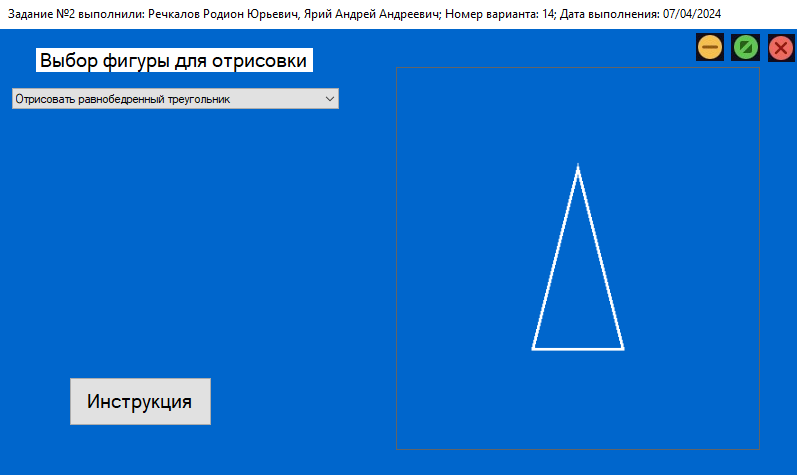
Рисунок 16 – Отрисовка прямоугольного треугольника Рисунок 17 – Отрисовка параллелограмма Рисунок 18 – Отрисовка воздушного шара Рисунок 19 – Отрисовка окружности  Рисунок 20 – Отрисовка эллипса  Рисунок 21 – Отрисовка прямоугольника  Рисунок 22 – Отрисовка круга  Рисунок 23 – Отрисовка трапеции  Рисунок 24 – Отрисовка равностороннего треугольника Рисунок 25 – Отрисовка квадрата 

Рисунок 26 – Отрисовка равнобедренного треугольника

Проверить кнопку «максимизировать/минимизировать»

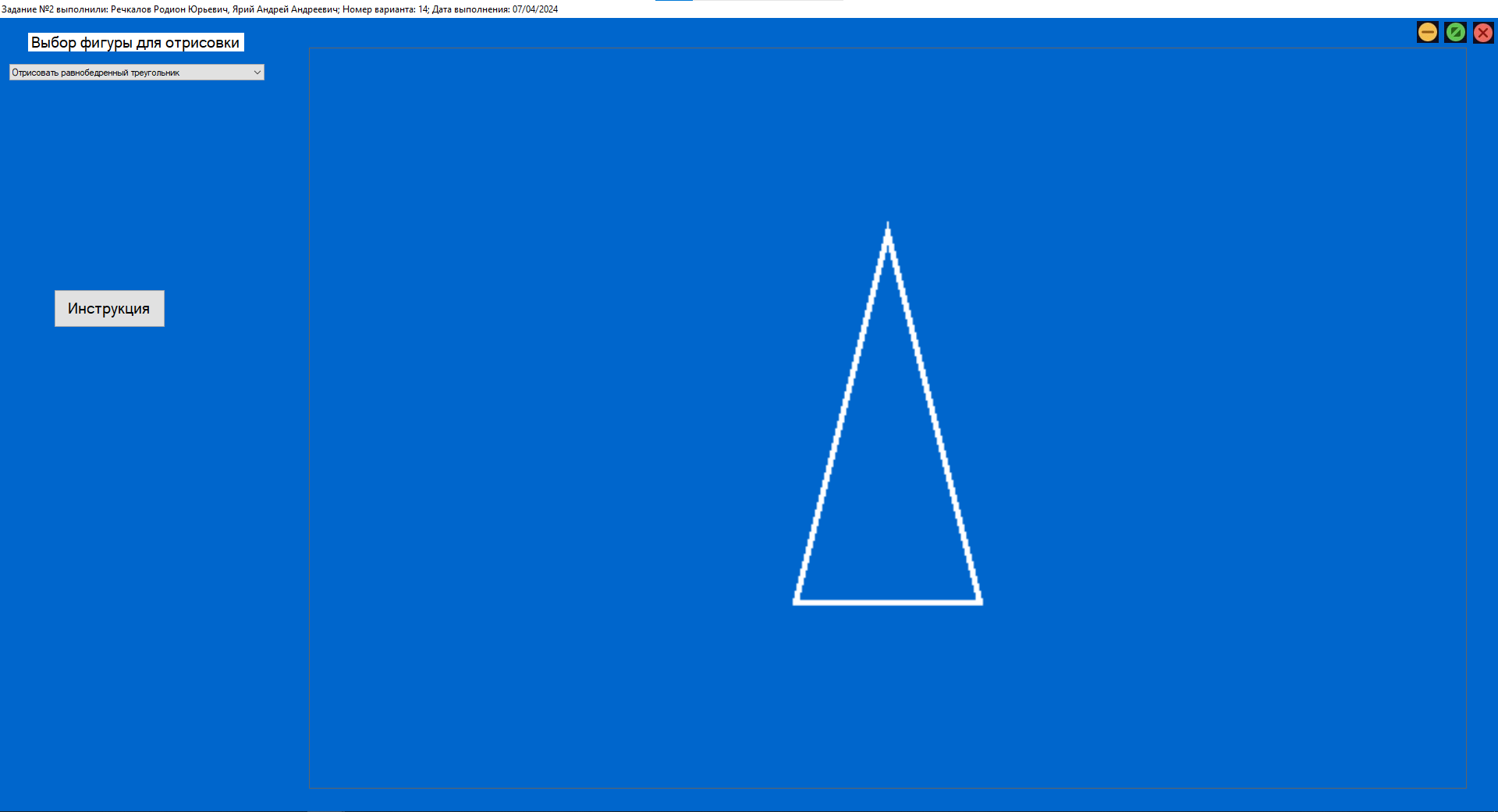


Рисунок 27 – Проверка кнопки «максимизировать/минимизировать»

# 11. Формулировку вывода о проделанной работе

В ходе выполнения данной работы мы приобрели и закрепили множество практических навыков в разработке приложений с графическим пользовательским интерфейсом на языке программирования C# с использованием среды Microsoft Visual Studio. Рассмотрим более подробно основные аспекты, освоенные в процессе работы:

1.Разработка графического пользовательского интерфейса

2.Работа с текстовыми файлами и кодировкой

3.Настройка множественных состояний объектов

4.Реализация графического вывода

5.Организация элементов управления

В целом, выполнение данной работы позволило не только закрепить теоретические знания, но и приобрести практические навыки, необходимые для разработки сложных приложений с графическим интерфейсом на языке C# в среде Microsoft Visual Studio. Мы научились реализовывать интерфейсы, работающие с графикой, управлять состояниями объектов и взаимодействовать с пользователем на высоком уровне.