Министерство транспорта Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

«Российский университет транспорта (МИИТ)»

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа №2

по дисциплине:

«Методы программирования»

на тему:  
«Конструктор ГПИ. Фигуры»

Выполнил: ст. гр. ТКИ-341

Черкас Н.В., Родин К.П.

Вариант №12

Проверил: к.т.н., доцент 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, список должен быть организован в виде кнопок с иконками. Фигуры из списка должны быть расположены в следующем порядке: 6, 7, 11, 8, 9, 3, 5, 4, 10, 2, 1. Так же необходимо поместить сложное комбинированное изображение «Дом» . Таким образом, фигуры должны быть расположены в следующем порядке: равносторонний треугольник, окружность, прямоугольник, ромб, трапеция, эллипс, круг, равнобедренный треугольник, параллелограмм, прямоугольный треугольник, квадрат.

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

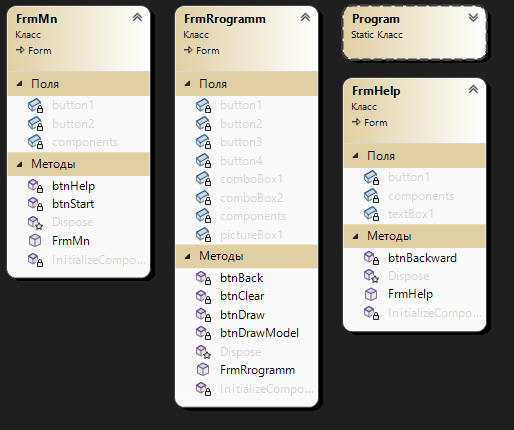


Рисунок 1 – Диаграмма классов

# **4. Составление сети Петри запрограммированного технологического процесса.**

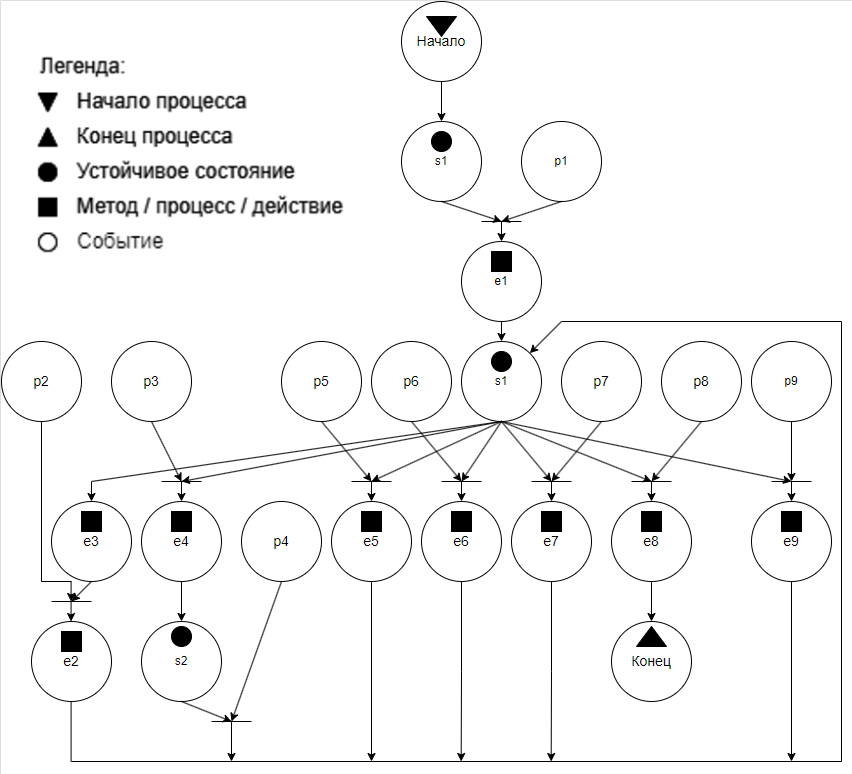


Рисунок 2 – Сеть Петри основной формы

**Описание сети Петри**

- состояния (states)

s1 – форма ожидает действий пользователя

s2 – форма находится в свернутом состоянии

- действия (effects)

e1 – переход в основную форму

e2 – отрисовка фигуры

e3 – пользователь выбирает фигуру

e4 – форма сворачивается

e5 – форма максимизирована/минимизирована

e6 – отрисовка готовой модели

e7 – очистка формы

e8 – закрытие формы

e9 - возвращение в главное меню

- события (prompts)

p1 – нажата кнопка «Старт»

p2 – нажата кнопка отрисовки фигуры

p3 – нажата кнопка «свернуть»

p4 – пользователь разворачивает форму

p5 – нажата кнопка «максимизировать/минимизировать»

p6 – нажата кнопка отрисовки готовой модели

p7 – нажата кнопка очистки формы

p8 – нажата кнопка закрытия формы

p9 – нажата кнопка «Вернуться в главное меню»

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

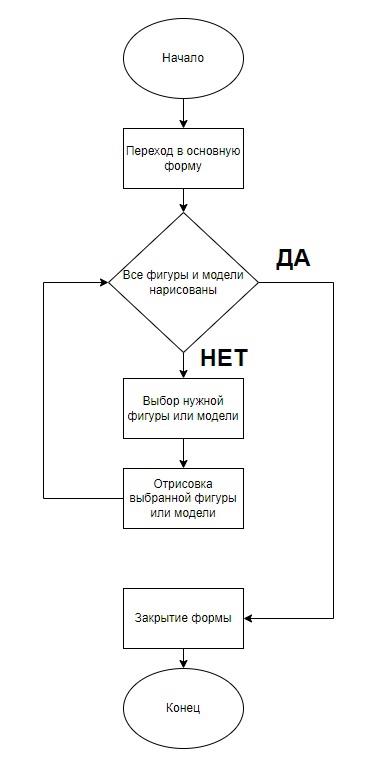


Рисунок 3 – Алгоритм кнопки "Начало работы с приложением"

Алгоритм отрисовки фигуры «Круг»:

* Создать поле 400х400
* Выбрать кисть с цветом коричневый
* Залить эллипс с координатой верхнего левого угла (-50; -50) и размерами (100; 100)

Алгоритм отрисовки фигуры «Равнобедренный треугольник»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатами точек

(-50; 50), (50; 50), (0; -50)

Алгоритм отрисовки фигуры «Ромб»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатами точек

(0; -50), (50; 0), (0; 50), (-50; 0)

Алгоритм отрисовки фигуры «Прямоугольный треугольник»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатами точек

(-50; 50), (50; 50), (-50; -50)

Алгоритм отрисовки фигуры «Эллипс»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать эллипс с координатой верхнего левого угла (-50; -30) и размерами (100; 60)

Алгоритм отрисовки фигуры «Квадрат»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатами верхнего левого угла

(-50; -50) и размерами (100; 100)

Алгоритм отрисовки фигуры «Окружность»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать эллипс с координатой верхнего левого угла (-50; -50) и размерами (100; 100)

Алгоритм отрисовки фигуры «Параллелограмм»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатами точек

(-40; 40), (20; 40), (40; -40), (-20; -40)

Алгоритм отрисовки фигуры «Трапеция»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатами точек

(-60; 40), (60; 40), (40; -40), (-40; -40)

Алгоритм отрисовки фигуры «Прямоугольник»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатой верхнего левого угла (-50; -50) и размерами (100; 100)

Алгоритм отрисовки фигуры «Равносторонний треугольник»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать полигон с координатами точек
* (0; -50), (43; 25), (-43; 25)

Алгоритм отрисовки фигуры «Дом»:

* Создать поле 400х400
* Выбрать ручку с цветом коричневый и толщиной 3
* Нарисовать равнобедренный треугольник с координатами (0; -50),
* (-43; -25), (43; 25)
* Нарисовать прямоугольник с координатами верхнего левого угла (-43; 25) и размерами (86; 86)
* Нарисовать прямоугольник с координатами верхнего левого угла (-30; 32) и размерами (60; 60)
* Нарисовать линию с координатами (-30; 55) и (30; 55)
* Нарисовать линию с координатами (0; 32) и (0; 92)
* Нарисовать эллипс с координатами верхнего левого угла (-15; 15) и размерами (30; 30)
* Нарисовать линию с координатами (-15; 0) и (15; 0)
* Нарисовать линию с координатами (0; -15) и (0; 15)
* Нарисовать линию с координатами (37; -30) и (37; 15)
* Нарисовать линию с координатами (22; -30) и (22; -13)
* Нарисовать линию с координатами (22; -30) и (37; -30)

# **6. Подбор тестовых примеров.**

Тестовые примеры:

* Запустить программу (проверка открытия основной и дочерней формы)
* Нажать кнопку «Старт»
* Проверить корректность отрисовки каждой из фигур
* Проверить кнопку «максимизировать/минимизировать»

**7. Листинг (код) составленного программного обеспечения.**

Форма главного меню:

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;

namespace WindowsFormsApp1

{

public partial class FrmMn : Form

{

public FrmMn()

{

InitializeComponent();

this.Cursor = Cursors.Hand;

}

private void btnStart(object sender, EventArgs e)

{

FrmRrogramm fr2 = new FrmRrogramm();

fr2.Show();

Hide();

}

private void btnHelp(object sender, EventArgs e)

{

FrmHelp fr4 = new FrmHelp();

fr4.Show();

Hide();

}

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

{

}

}

}

Основная 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;

namespace WindowsFormsApp1

{

public partial class FrmRrogramm : Form

{

public FrmRrogramm()

{

InitializeComponent();

this.Cursor = Cursors.Hand;

}

private void btnDraw(object sender, EventArgs e)

{

Graphics graphics = this.pictureBox1.CreateGraphics();

switch (comboBox1.SelectedIndex)

{

case 0: Point[] equilateralTrianglePoints = { new Point(Location.X, Location.Y - 50), new Point(Location.X + 43, Location.Y + 25), new Point(Location.X - 43, Location.Y + 25) }; graphics.DrawPolygon(Pens.SandyBrown, equilateralTrianglePoints); break;

case 1: graphics.DrawEllipse(Pens.SandyBrown, Location.X - 50, Location.Y - 50, 100, 100); break;

case 2: graphics.DrawRectangle(Pens.SandyBrown, Location.X - 50, Location.Y - 50, 200, 100); break;

case 3: Point[] rhombusPoints = { new Point(Location.X, Location.Y - 50), new Point(Location.X + 50, Location.Y), new Point(Location.X, Location.Y + 50), new Point(Location.X - 50, Location.Y) }; graphics.DrawPolygon(Pens.SandyBrown, rhombusPoints); break;

case 4: Point[] invertedTrapezoidPoints = { new Point(Location.X - 60, Location.Y + 40), new Point(Location.X + 60, Location.Y + 40), new Point(Location.X + 40, Location.Y - 40), new Point(Location.X - 40, Location.Y - 40) }; graphics.DrawPolygon(Pens.SandyBrown, invertedTrapezoidPoints); break;

case 5: graphics.DrawEllipse(Pens.SandyBrown, Location.X - 50, Location.Y - 30, 100, 60); break;

case 6: graphics.FillEllipse(Brushes.SandyBrown, Location.X - 50, Location.Y - 50, 100, 100); break;

case 7: Point[] isoscelesTrianglePoints = { new Point(Location.X - 50, Location.Y + 50), new Point(Location.X + 50, Location.Y + 50), new Point(Location.X, Location.Y - 50) }; graphics.DrawPolygon(Pens.SandyBrown, isoscelesTrianglePoints); break;

case 8: Point[] parallelogramPoints = { new Point(Location.X - 40, Location.Y + 40), new Point(Location.X + 20, Location.Y + 40), new Point(Location.X + 40, Location.Y - 40), new Point(Location.X - 20, Location.Y - 40) }; graphics.DrawPolygon(Pens.SandyBrown, parallelogramPoints); break;

case 9: Point[] trianglePoints = { new Point(Location.X - 50, Location.Y + 50), new Point(Location.X + 50, Location.Y + 50), new Point(Location.X - 50, Location.Y - 50) }; graphics.DrawPolygon(Pens.SandyBrown, trianglePoints); break;

case 10: graphics.DrawRectangle(Pens.SandyBrown, Location.X - 50, Location.Y - 50, 100, 100); break;

}

}

private void btnDrawModel(object sender, EventArgs e)

{

Graphics graphics = this.pictureBox1.CreateGraphics();

switch (comboBox2.SelectedIndex)

{

case 0: // Крыша

Point[] Roof = { new Point(Location.X, Location.Y - 50), new Point(Location.X + 43, Location.Y + 25), new Point(Location.X - 43, Location.Y + 25) };

graphics.DrawPolygon(Pens.SandyBrown, Roof);

// Стены

graphics.DrawRectangle(Pens.SandyBrown, Location.X - 43, Location.Y + 25, 86, 86);

// Окно

graphics.DrawRectangle(Pens.White, Location.X - 30, Location.Y + 32, 60, 60);

// Ставни в окна

graphics.DrawLine(Pens.White, Location.X - 30, Location.Y + 55, Location.X + 30, Location.Y + 55);

graphics.DrawLine(Pens.White, Location.X, Location.Y + 32, Location.X, Location.Y + 92);

// Окно в крыше

graphics.DrawEllipse(Pens.White, Location.X - 15, Location.Y - 15, 30, 30);

// Ставни в окне на крыше

graphics.DrawLine(Pens.White, Location.X - 15, Location.Y, Location.X + 15, Location.Y);

graphics.DrawLine(Pens.White, Location.X, Location.Y - 15, Location.X, Location.Y + 15);

graphics.DrawLine(Pens.SandyBrown, Location.X + 37, Location.Y - 30, Location.X + 37, Location.Y + 15);

graphics.DrawLine(Pens.SandyBrown, Location.X + 22, Location.Y - 30, Location.X + 22, Location.Y - 13);

graphics.DrawLine(Pens.SandyBrown, Location.X + 22, Location.Y - 30, Location.X + 37, Location.Y - 30);

break;

}

}

private void btnClear(object sender, EventArgs e)

{

pictureBox1.Image = null;

}

private void btnBack(object sender, EventArgs e)

{

FrmMn fr1 = new FrmMn();

fr1.Show();

Hide();

}

}

}

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;

namespace WindowsFormsApp1

{

public partial class FrmHelp : Form

{

public FrmHelp()

{

InitializeComponent();

this.Cursor = Cursors.Hand;

}

private void btnBackward(object sender, EventArgs e)

{

FrmMn fr1 = new FrmMn();

fr1.Show();

Hide();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

}

}

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

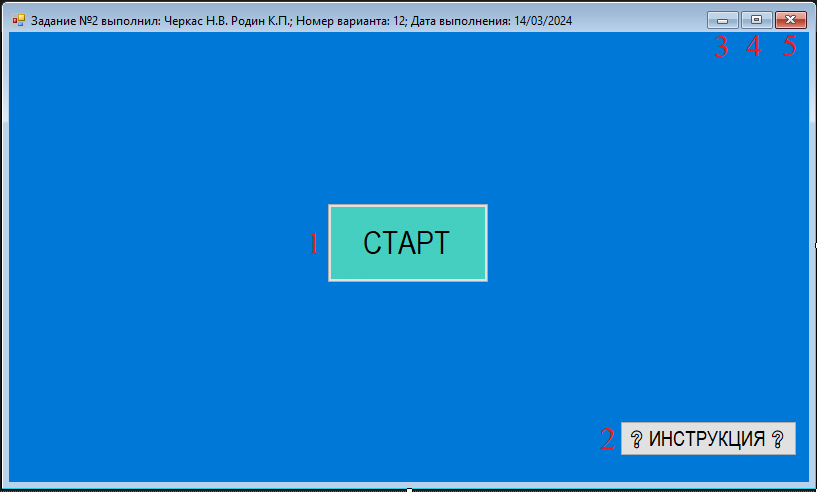


Рисунок 4 – Интерфейс главного меню

1. Кнопка начала работы программы
2. Кнопка вызова инструкции
3. Кнопка «свернуть»
4. Кнопка «максимизировать/минимизировать»
5. Кнопка «закрыть»

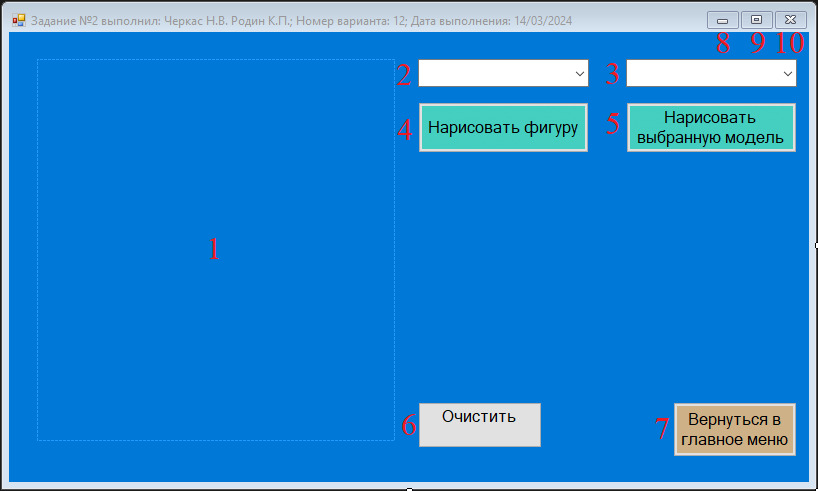


Рисунок 5 – Интерфейс основной формы

1. Зона отрисовки
2. Окошко выбора фигуры
3. Окошко выбора готовой модели
4. Кнопка отрисовки фигуры
5. Кнопка отрисовки готовой модели
6. Кнопка очистки зоны отрисовки
7. Кнопка возвращения в главное меню
8. Кнопка «свернуть»
9. Кнопка «максимизировать/минимизировать»
10. Кнопка «закрыть»

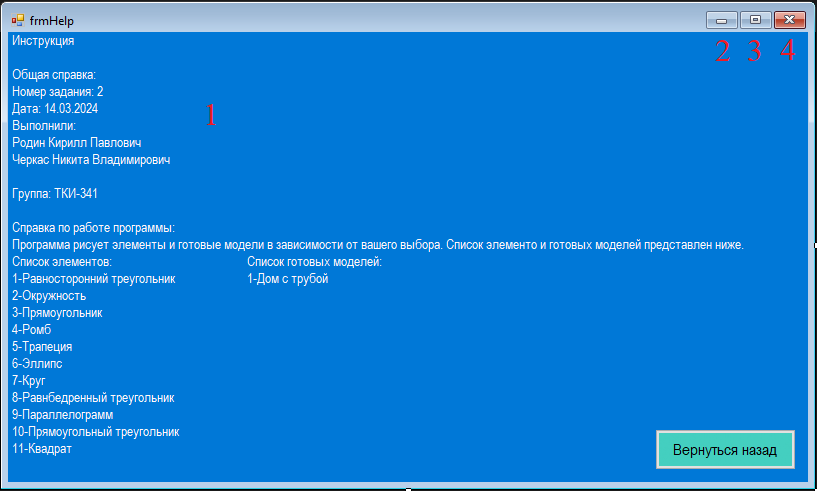


Рисунок 6 – Интерфейс вспомогательной формы

1. Зона для выгрузки текста из файла
2. Кнопка «свернуть»
3. Кнопка «максимизировать/минимизировать»
4. Кнопка «закрыть

# **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 7 – Название формы

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



Рисунок 8 – Время на момент создания законченной программы

4. Чётные варианты заменяют стандартный курсор экранной формы со «стрелки» на «руку».

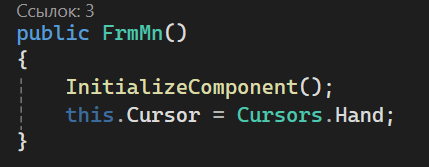


Рисунок 9 – Фрагмент кода, в котором происходит замена курсора

5. Справочная информация должна быть вызвана в дочерней экранной форме и считана в статический по размеру ярлык (Label) из текстового файла (нечётные варианты)

см. Рисунок 7; листинг вспомогательной формы

7. Чётные варианты заменяют цвет подложки в соответствии с представленной ниже таблицей:

Таблица 1 – Таблица системных цветов подложки

|  |  |
| --- | --- |
| 2. | *ActiveCaption* |
| 4. | *AppWorkspace* |
| 6. | *ControlText* |
| 8. | *Desktop* |
| 10. | *GradientActiveCaption* |
| 12. | *Highlight* |
| 14. | *HotTrack* |
| 16. | *Info* |
| 18. | *MenuHighlight* |
| 20. | *Window* |



Рисунок 10 – Выбранный фоновый цвет

8. Исходное состояние всех элементов, расположенных на главной экранной форме, должно быть настроено через перечень параметров этих элементов.

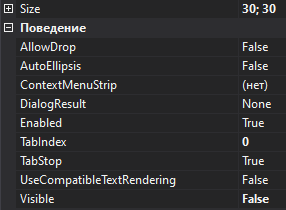


Рисунок 11 – Параметры кнопки "вернуться в главное меню"

9. В качестве исходного состояния принимается заранее известная и заполненная элементами структура списков, все элементы экранной формы за исключением главного меню находятся либо в недоступном состоянии (*.Enabled = false*), либо в невидимом состоянии (*.Visible = false*).

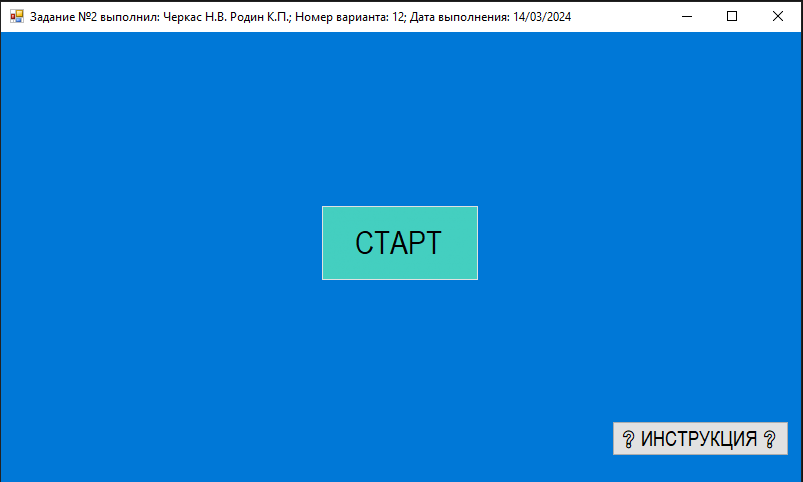


Рисунок 12 – Интерфейс до нажатия кнопки "Старт"

10. Пункт меню «Начало работы с приложением» должен реализовывать активацию доступа пользователя к элементам или отображение элементов на экранной форме для пользователя; список должен быть организован в виде кнопок с иконками; фигуры должны быть расположены в следующем порядке: круг, равнобедренный треугольник, ромб, прямоугольный треугольник, эллипс, квадрат, параллелограмм, прицел, трапеция, равносторонний треугольник, прямоугольник, окружность.

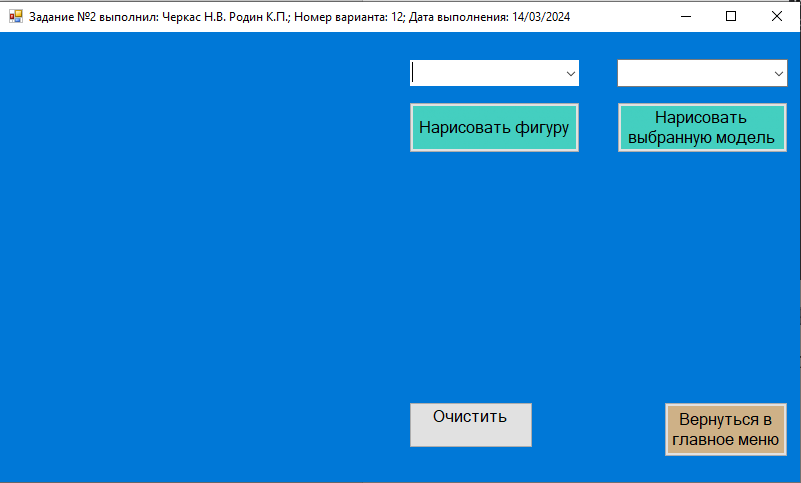


Рисунок 13 – Интерфейс после нажатия кнопки "Начало работы с приложением"

11. Все элементы программы должны носить значащие имена переменных, в которых отражено существо этих элементов, например, главная экранная форма – *frmMain*, ярлык – *lblHelp*, комбинированный список – *cmbFigures* и т.д.

см. Рисунки 1-3

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

Тестовые примеры:

* Запустить программу (проверка открытия основной и дочерней формы)

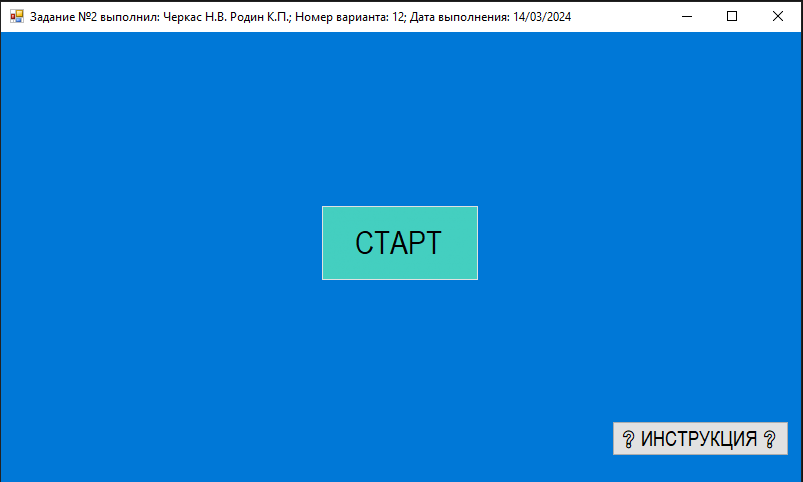


Рисунок 14 – Открытие главной формы

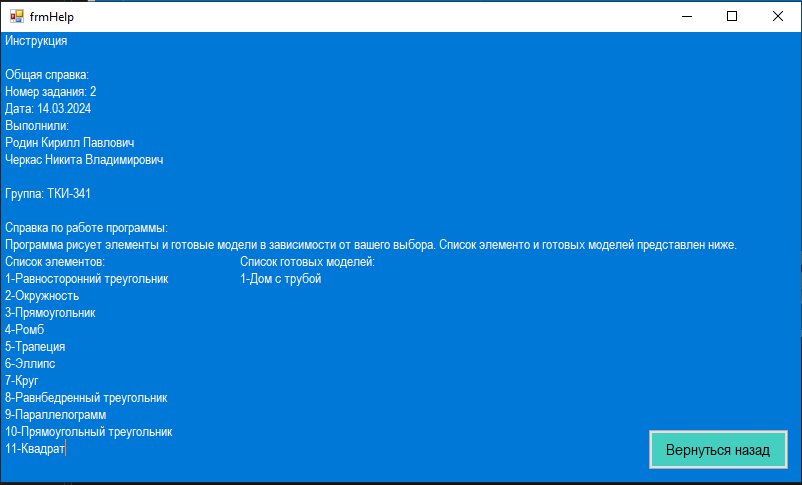


Рисунок 15 – Открытие вспомогательной формы

* Нажать кнопку «Начало работы с приложением»

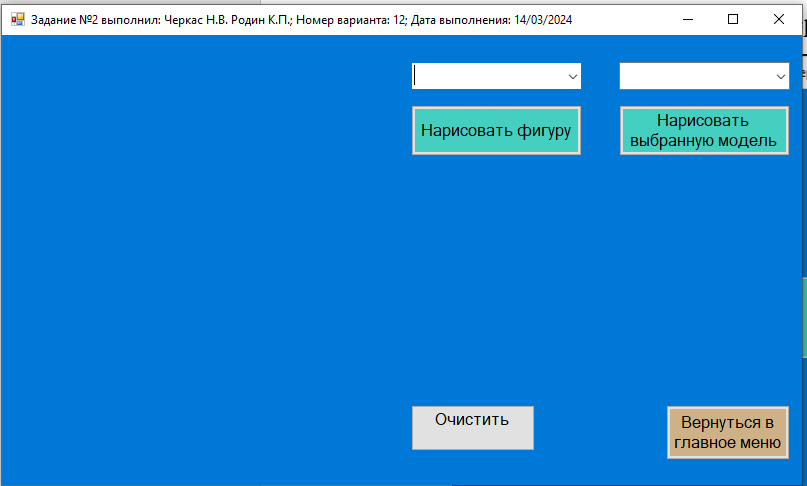


Рисунок 16 – Форма после начала работы

* Проверить корректность отрисовки каждой из фигур

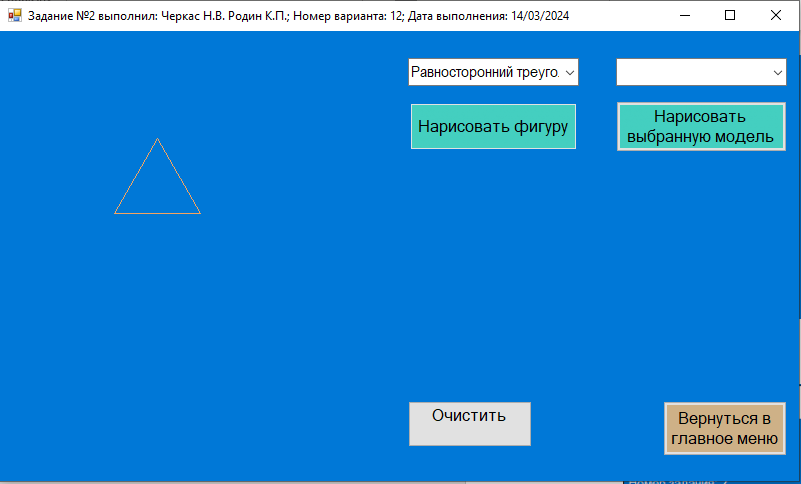


Рисунок 17 – Отрисовка равностороннего треугольника

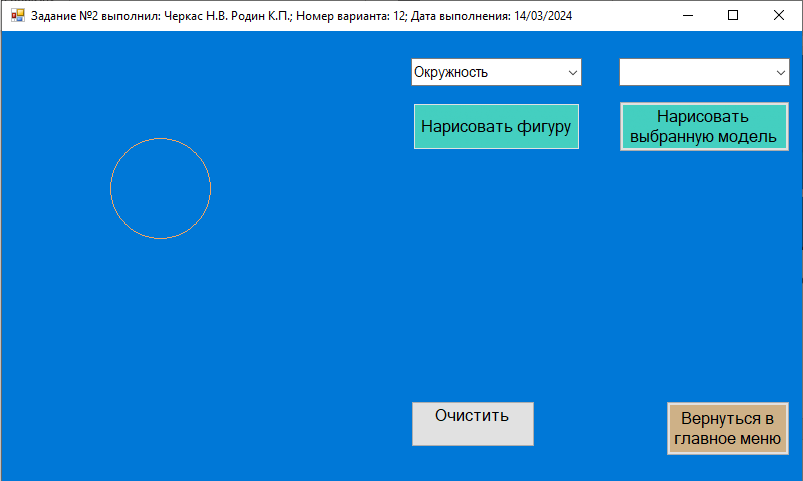


Рисунок 18– Отрисовка окружности

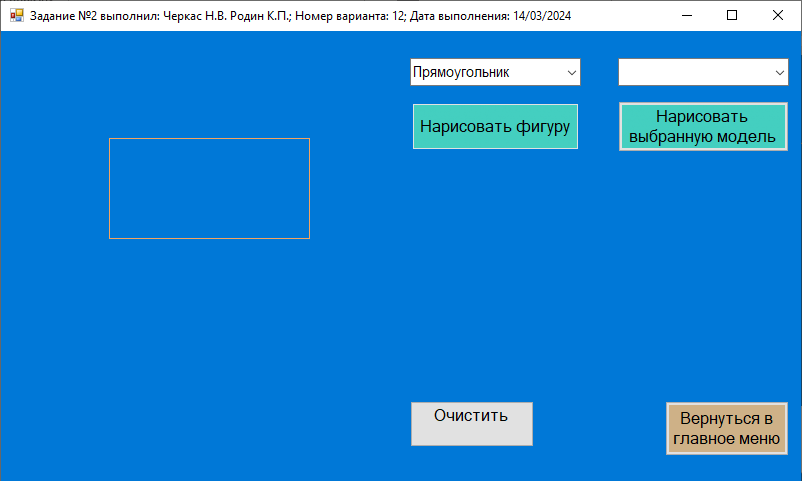


Рисунок 19 – Отрисовка ромба

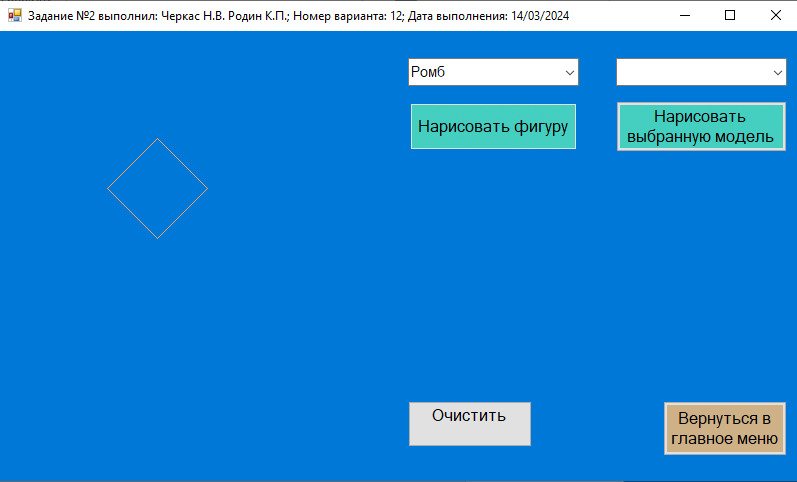


Рисунок 20 – Отрисовка ромба



Рисунок 21 – Отрисовка трапеции

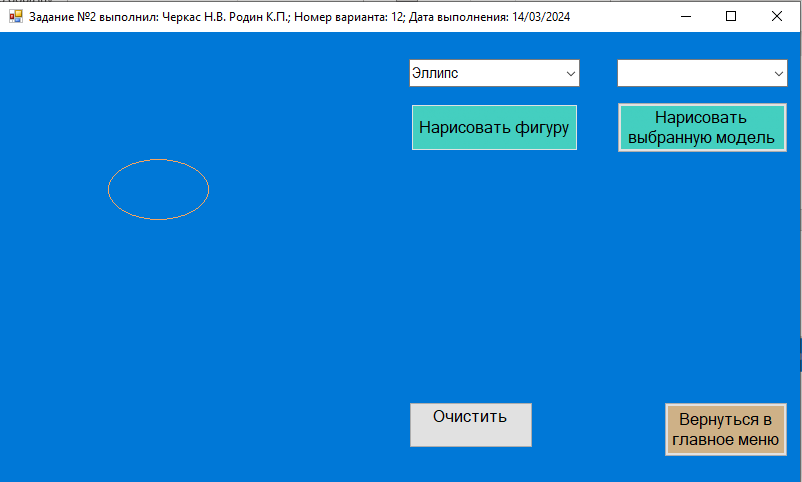


Рисунок 22 – Отрисовка эллипса

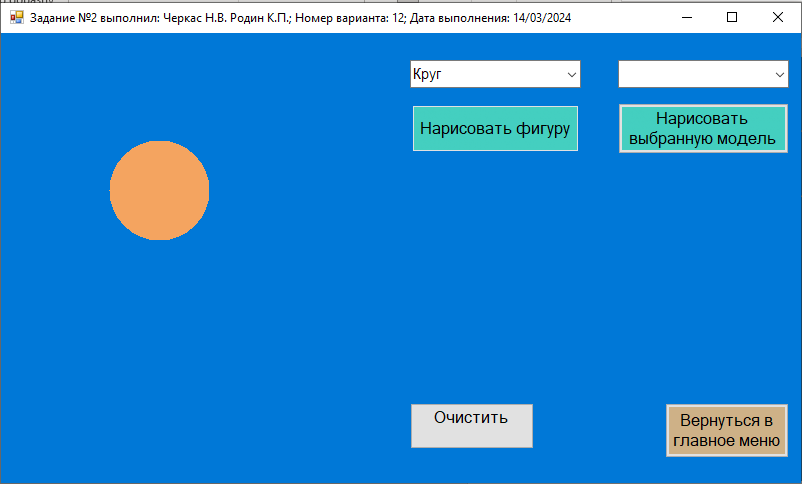


Рисунок 23 – Отрисовка круга

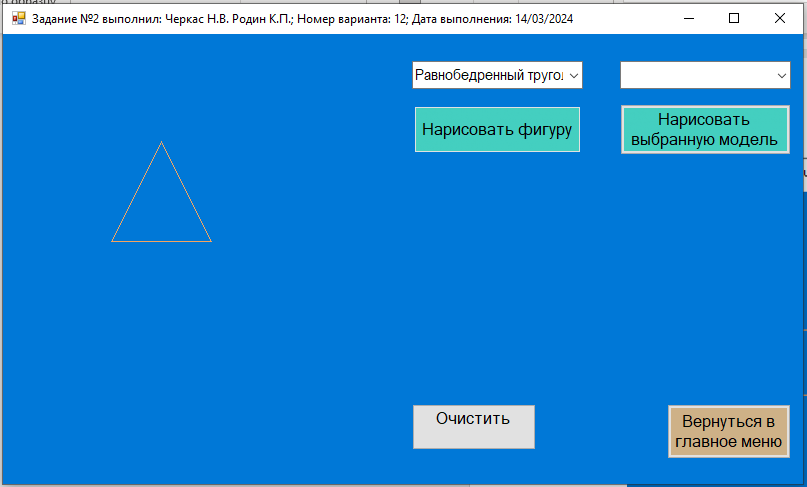


Рисунок 24 – Отрисовка равнобедренного треугольника

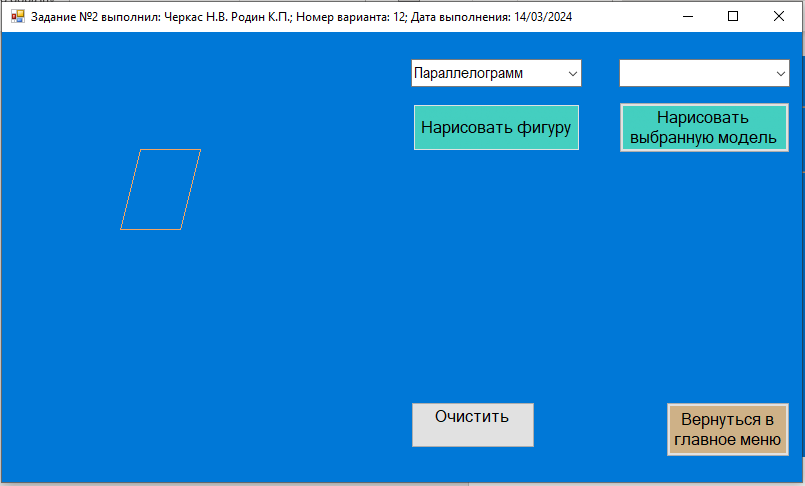


Рисунок 25 – Отрисовка параллелограмма

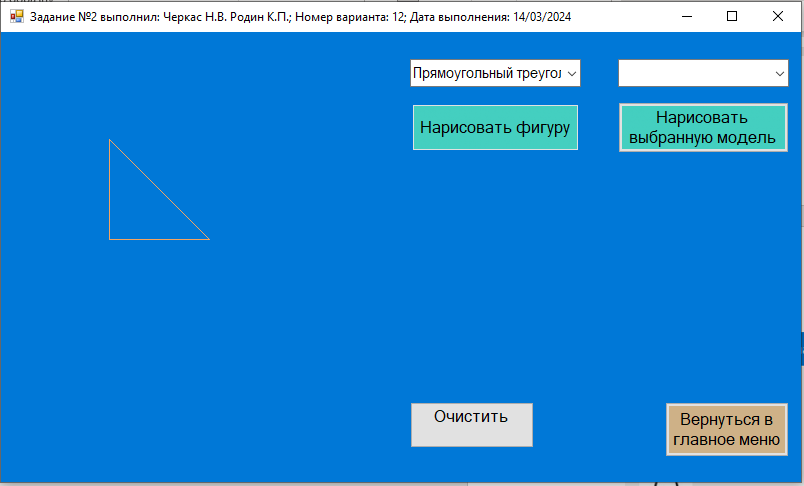


Рисунок 26 – Отрисовка прямоугольного треугольника

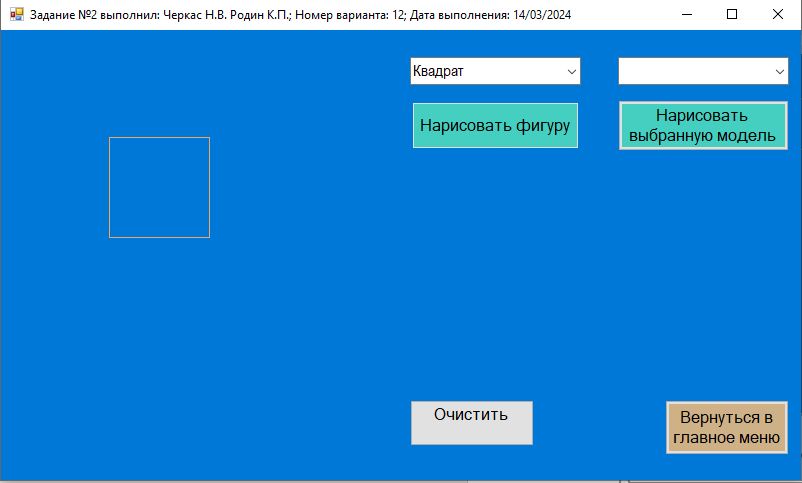


Рисунок 27 – Отрисовка квадрата

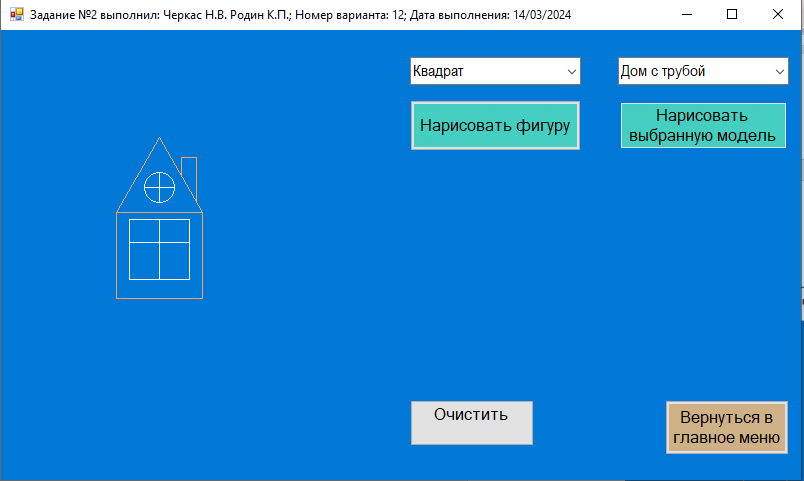


Рисунок 28 – Отрисовка дома

# **11. Формулировку вывода о проделанной работе**

В рамках данной работы были закреплены навыки разработки визуального пользовательского интерфейса, освоена работа с текстовыми файлами и кодировкой в среде *Microsoft* *Visual Studio*, были получены знания о реализации настройки множественных состояний объектов посредством управления компонентами со внутренней индексацией.