МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ижевский государственный технический университет имени М. Т. Калашникова»

Факультет «Информатика и вычислительная техника»

Кафедра «Программное обеспечение»

Работа защищена с оценкой

«\_\_\_\_»

Дата \_\_\_\_\_\_\_\_\_\_

Подпись \_\_\_\_\_\_\_\_\_\\_\_\_\_\_\_\_\_\_\_

КУРСОВАЯ РАБОТА

по дисциплине «Программирование»

на тему: «Проекты на основе WF»

Выполнил

Студент группы Б18-191-3 Э. О. Халилов

Руководитель

д.т.н., профессор В. Г. Тарасов

Рецензия:

степень достижения поставленной цели работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

полнота разработки темы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

уровень самостоятельности работы обучающегося\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

недостатки работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc57927707)

[1. ПОСТАНОВКА ЗАДАЧИ И ИНСТРУМЕНТЫ ДЛЯ ЕЕ РЕШЕНИЯ 4](#_Toc57927708)

[2. СОЗДАНИЕ ГРАФИЧЕСКОГО ПРИЛОЖЕНИЯ 5](#_Toc57927709)

[2.1 Установка среды разработки и ее компонентов 5](#_Toc57927710)

[2.2 Создание нового проекта в Visual Studio 5](#_Toc57927711)

[3. ОСНОВЫ РАБОТЫ С WINDOWS FORMS 7](#_Toc57927712)

[3.1 Графический дизайнер 7](#_Toc57927713)

[3.2 Панель инструментов 7](#_Toc57927714)

[3.3 Запуск приложения 10](#_Toc57927715)

[3.4 Свойства 11](#_Toc57927716)

[4. РАЗРАБОТКА ПРОГРАММЫ 13](#_Toc57927717)

[4.1 Подготовка объектов 13](#_Toc57927718)

[4.2 Подготовка элементов 13](#_Toc57927719)

[4.3 Логика работы программы 14](#_Toc57927720)

[5. КОНТРОЛЬНЫЙ ПРИМЕР 17](#_Toc57927721)

[ЗАКЛЮЧЕНИЕ 18](#_Toc57927722)

[ПРИЛОЖЕНИЕ 1 19](#_Toc57927723)

# ВВЕДЕНИЕ

Цель работы заключается в разработке программы, решающую головоломку типа "Инки".

Программа состоит в том, что пользователь выбирает размер поля и количество областей, на которые разделена таблица. Затем пользователь вводит описание данных областей, после чего рисуется квадратное поле, поделенное на области. Каждая область закрашена своим цветом. Нажав на кнопку "Начать сначала", всё сбрасывается и программа переходит в самое начало.

# 1. ПОСТАНОВКА ЗАДАЧИ И ИНСТРУМЕНТЫ ДЛЯ ЕЕ РЕШЕНИЯ

Разработать программу, решающую головоломку типа "Инки".

Расположить на форме кнопки и поля для ввода данных, позволяющие пользователю взаимодействовать с программой, а также само поле.

Программа делит поле на области по описанию из полей ввода и закрашивает области разными цветами. Появляется кнопка "Начать сначала", нажав которую, программа возвращается к началу.

Для реализации данной задачи выбрана платформа .Net, технология Windows forms, язык C#. Программный код пишется в среде разработки Visual Studio 2019.

# 2. СОЗДАНИЕ ГРАФИЧЕСКОГО ПРИЛОЖЕНИЯ

## 2.1 Установка среды разработки и ее компонентов

Для создания графического проекта нам потребуется среда разработки Visual Studio. Устанавливаем среду разработки и ее компоненты с помощью Visual Studio Installer (Разработка классических приложений .Net):

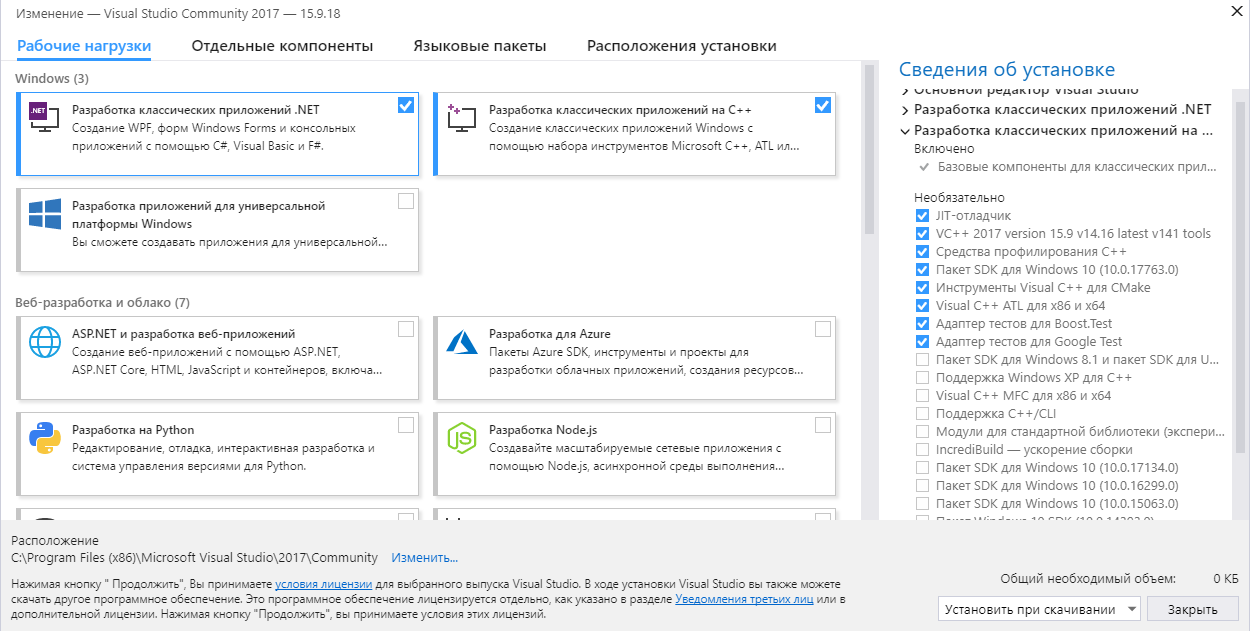


Рис 2.1 Установка среды разработки и ее компонентов

## 2.2 Создание нового проекта в Visual Studio

После установки среды и всех ее компонентов, запустим Visual Studio и создадим проект графического приложения. Для этого в меню выберем пункт File (Файл) и в подменю выберем New - > Project (Создать - > Проект). После этого перед нами откроется диалоговое окно создания нового проекта:

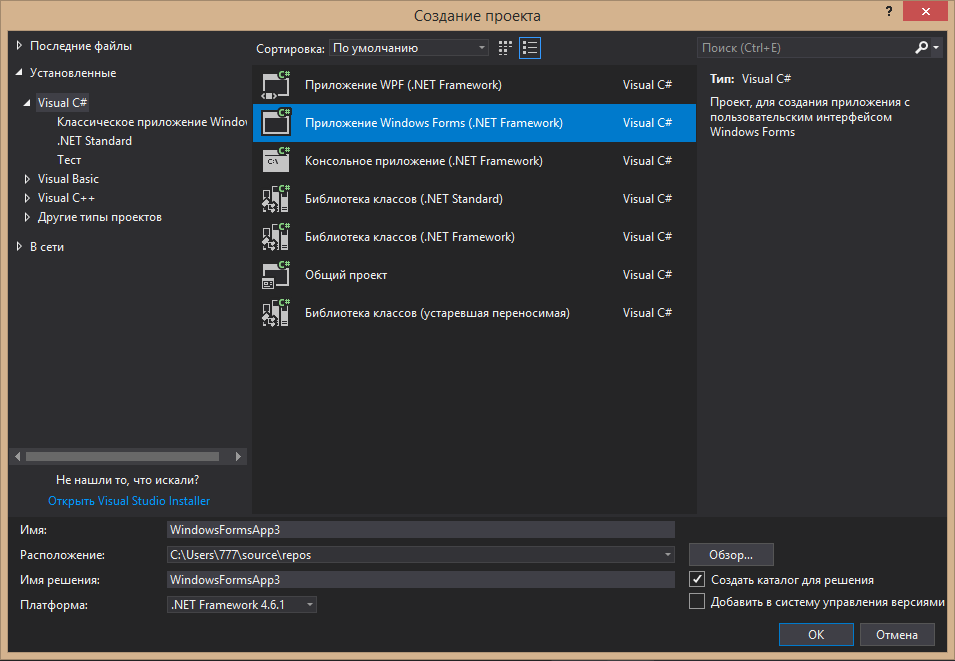


Рис. 2.2 Создание нового проекта

Среди типов проектов выбираем Приложение Windows Forms (.Net Framework). После этого нажимаем OK. После этого Visual Studio откроет наш проект с созданными по умолчанию файлами:

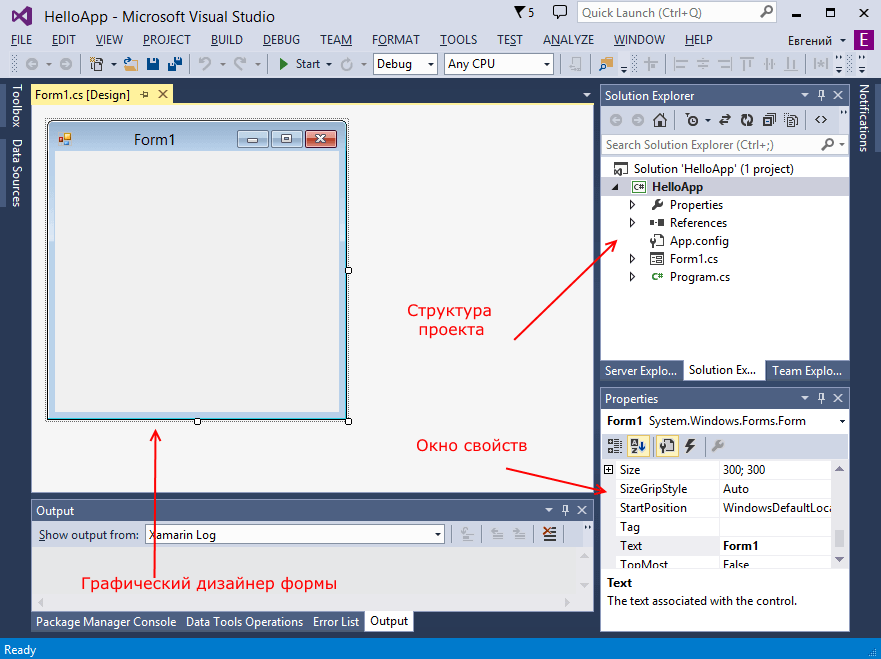


Рис. 2.3 Открытие графического приложения

# 3. ОСНОВЫ РАБОТЫ С WINDOWS FORMS

## 3.1 Графический дизайнер

Большую часть пространства Visual Studio занимает графический дизайнер, который содержит форму будущего приложения. Пока она пуста и имеет только заголовок Form1. Справа находится окно файлов решения/проекта - Solution Explorer (Обозреватель решений). Там и находятся все связанные с нашим приложением файлы, в том числе файлы формы Form1.cs.

Внизу справа находится окно свойств - Properties. Так как в данный момент выбрана форма как элемент управления, то в этом поле отображаются свойства, связанные с формой.

Теперь найдем в этом окне свойство формы Text и изменим его значение на любое другое:

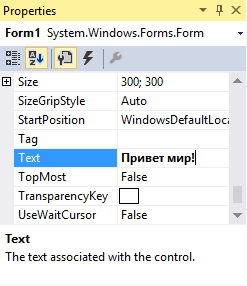


Рис. 3.1 Свойства формы

Таким образом мы поменяли заголовок формы.

## 3.2 Панель инструментов

Теперь перенесем на поле какой-нибудь элемент управления, например, кнопку. Для этого найдем в левой части Visual Studio вкладку Toolbox (Панель инструментов). Нажмем на эту вкладку, и у нас откроется панель с элементами, откуда мы можем с помощью мыши перенести на форму любой элемент:

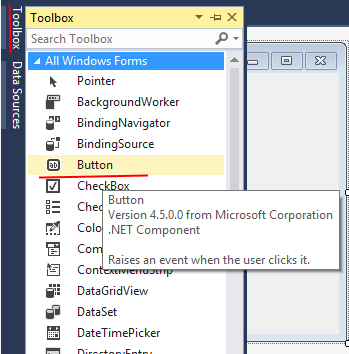


Рис. 3.2 Панель инструментов

Найдем среди элементов кнопку и, захватив ее указателем мыши, перенесем на форму:

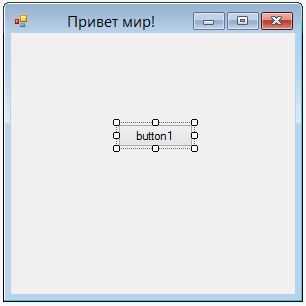


Рис. 3.3 Добавление кнопки на форму

Это визуальная часть. Теперь приступим к самому программированию. Добавим простейший код на языке C#, который бы выводил сообщение по нажатию кнопки. Для этого мы должны перейти в файл кода, который связан с этой формой. Если у нас не открыт файл кода, мы можем нажать на форму правой кнопкой мыши и в появившемся меню выбрать View Code (Посмотреть файл кода):

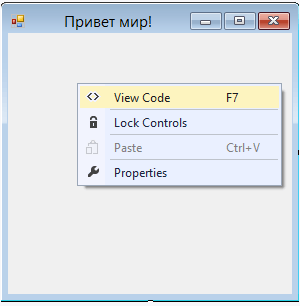


Рис. 3.4 Посмотреть файл кода

Однако можно воспользоваться другим способом, чтобы не писать много лишнего кода. Нужно указатель мыши на кнопку и щелкнуть по ней двойным щелчком. Мы автоматически попадаем в файл кода Form1.cs. Добавим вывод сообщения по нажатию кнопки, изменив код следующим образом:

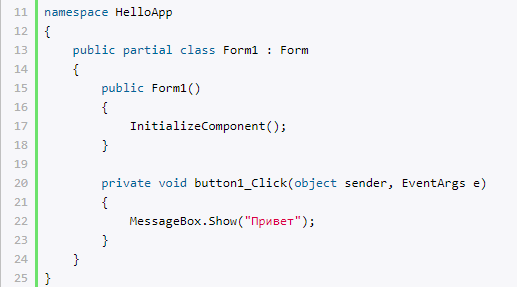


Рис. 3.5 Сообщение по нажатию кнопки

## 3.3 Запуск приложения

Чтобы запустить приложение в режиме отладки, нажмем на клавишу F5 или на зеленую стрелочку на панели Visual Studio. После этого запустится наша форма с одинокой кнопкой. И если мы нажмем на кнопку на форме, то нам будет отображено сообщение с приветствием.

После запуска приложения студия компилирует его в файл с расширением exe. Найти данный файл можно, зайдя в папку проекта и далее в каталог bin/Debug или bin/Release

Рассмотрев вкратце создание проекта графического приложения, мы можем перейти к обзору основных компонентов и начнем мы с форм.

Внешний вид приложения является нам преимущественно через формы. Формы являются основными строительными блоками. Они предоставляют контейнер для различных элементов управления. А механизм событий позволяет элементам формы отзываться на ввод пользователя, и, таким образом, взаимодействовать с пользователем.

При открытии проекта в Visual Studio в графическом редакторе мы можем увидеть визуальную часть формы - ту часть, которую мы видим после запуска приложения и куда мы переносим элементы с панели управления. Но на самом деле форма скрывает мощный функционал, состоящий из методов, свойств, событий и прочее. Рассмотрим основные свойства форм.

Если мы запустим приложение, то нам отобразится одна пустая форма. Однако даже такой простой проект с пустой формой имеет несколько компонентов:

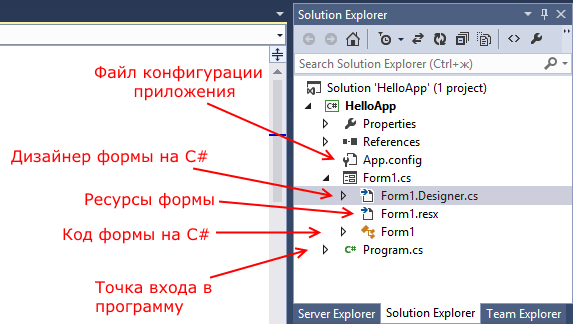


Рис. 3.6 Компоненты

## 3.4 Свойства

С помощью специального окна Properties (Свойства) справа Visual Studio предоставляет нам удобный интерфейс для управления свойствами элемента:

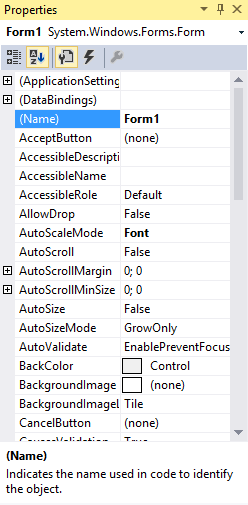


Рис. 3.7 Свойства элемента

Большинство этих свойств оказывает влияние на визуальное отображение формы. Основные свойства:

* Name: устанавливает имя формы - точнее имя класса, который наследуется от класса Form
* BackColor: указывает на фоновый цвет формы. Щелкнув на это свойство, мы сможем выбрать тот цвет, который нам подходит из списка предложенных цветов или цветовой палитры
* BackgroundImage: указывает на фоновое изображение формы
* BackgroundImageLayout: определяет, как изображение, заданное в свойстве BackgroundImage, будет располагаться на форме.
* ControlBox: указывает, отображается ли меню формы. В данном случае под меню понимается меню самого верхнего уровня, где находятся иконка приложения, заголовок формы, а также кнопки минимизации формы и крестик. Если данное свойство имеет значение false, то мы не увидим ни иконку, ни крестика, с помощью которого обычно закрывается форма
* Cursor: определяет тип курсора, который используется на форме
* Enabled: если данное свойство имеет значение false, то она не сможет получать ввод от пользователя, то есть мы не сможем нажать на кнопки, ввести текст в текстовые поля и т.д.
* Font: задает шрифт для всей формы и всех помещенных на нее элементов управления. Однако, задав у элементов формы свой шрифт, мы можем тем самым переопределить его
* ForeColor: цвет шрифта на форме
* FormBorderStyle: указывает, как будет отображаться граница формы и строка заголовка. Устанавливая данное свойство в None можно создавать внешний вид приложения произвольной формы
* HelpButton: указывает, отображается ли кнопка справки формы
* Icon: задает иконку формы
* Location: определяет положение по отношению к верхнему левому углу экрана, если для свойства StartPosition установлено значение Manual
* MaximizeBox: указывает, будет ли доступна кнопка максимизации окна в заголовке формы
* MinimizeBox: указывает, будет ли доступна кнопка минимизации окна
* MaximumSize: задает максимальный размер формы
* MinimumSize: задает минимальный размер формы
* Opacity: задает прозрачность формы
* Size: определяет начальный размер формы
* StartPosition: указывает на начальную позицию, с которой форма появляется на экране
* Text: определяет заголовок формы
* TopMost: если данное свойство имеет значение true, то форма всегда будет находиться поверх других окон
* Visible: видима ли форма, если мы хотим скрыть форму от пользователя, то можем задать данному свойству значение false
* WindowState: указывает, в каком состоянии форма будет находиться при запуске: в нормальном, максимизированном или минимизированном

# 4. РАЗРАБОТКА ПРОГРАММЫ

## 4.1 Подготовка объектов

Для игры потребуется окно размером 900x600 px, название окна "Инки".

## 4.2 Подготовка элементов

Создадим Label размером 400x25 с текстом "Выберите размер поля", Label размером 30х30, отображающий выбранный размер поля, две кнопки размером 35х35, позволяющие изменять размеры поля, Label размером 300х25 с текстом "Введите количество областей:", TextBox размером 25х25 для ввода количества областей и кнопку размером 150х30 для отправки количества областей:

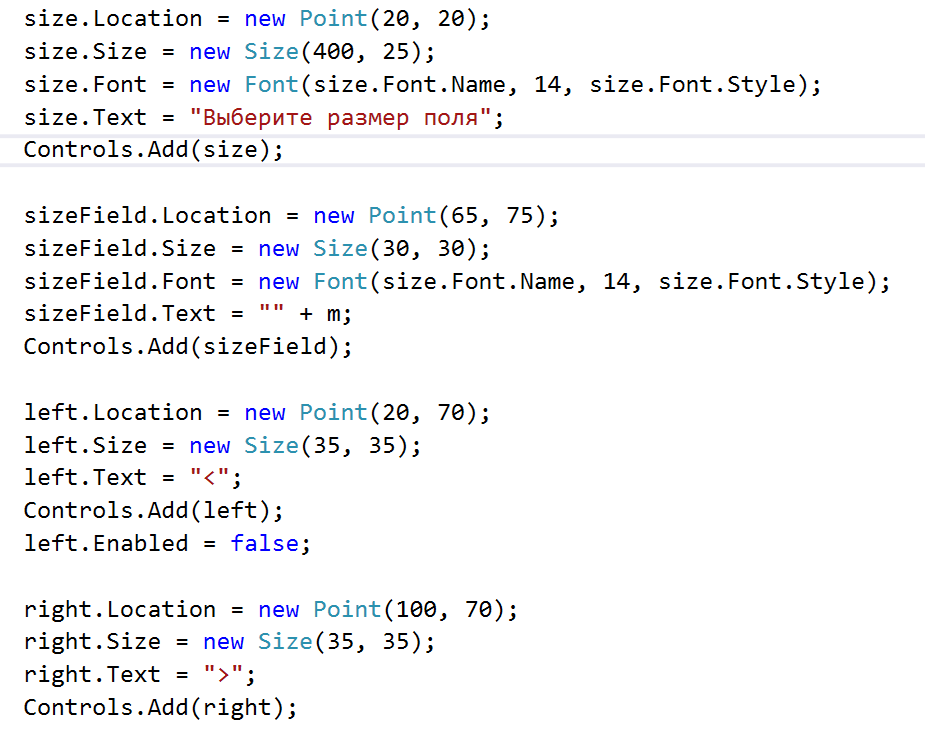


Рис. 4.1

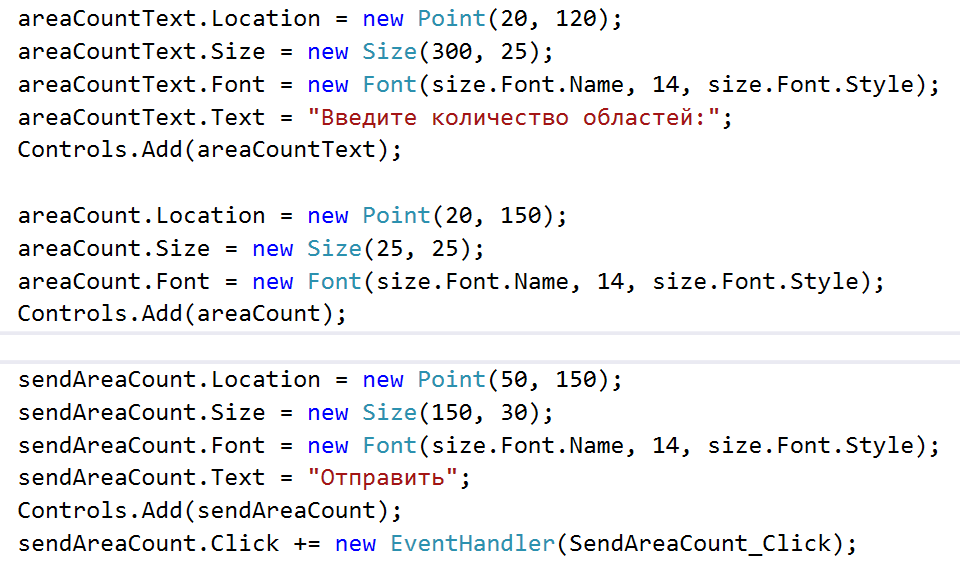


Рис. 4.2

## 4.3 Логика работы программы

Вид формы при запуске программы:

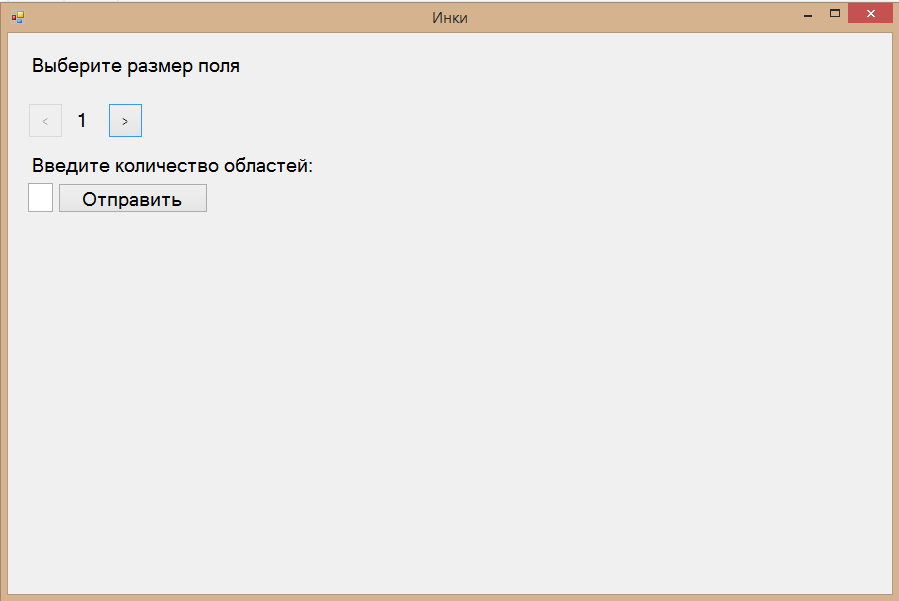


Рис. 4.3 - Форма при запуске программы

После выбора размера поля и количества областей появляется несколько полей для ввода описаний для каждой области и кнопка "Отправить" :

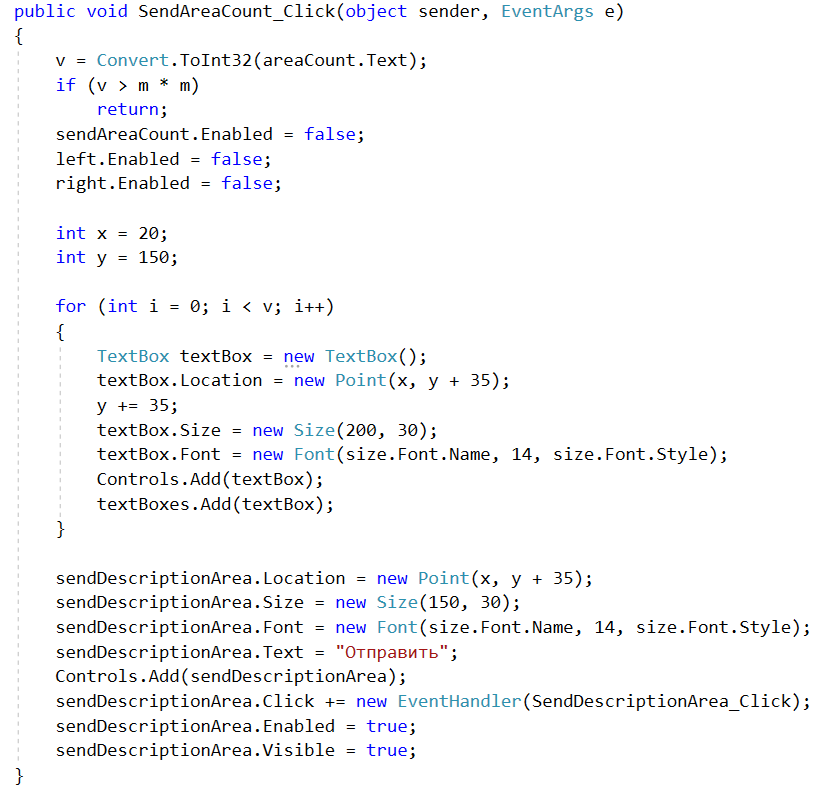


Рис. 4.4 - Создание полей для ввода описаний для каждой области и кнопки отправить

После ввода описаний областей и нажатия на кнопку "Отправить" рисуется поле, поделенное на области, которые закрашены разными цветами и появляется кнопка "Начать сначала":

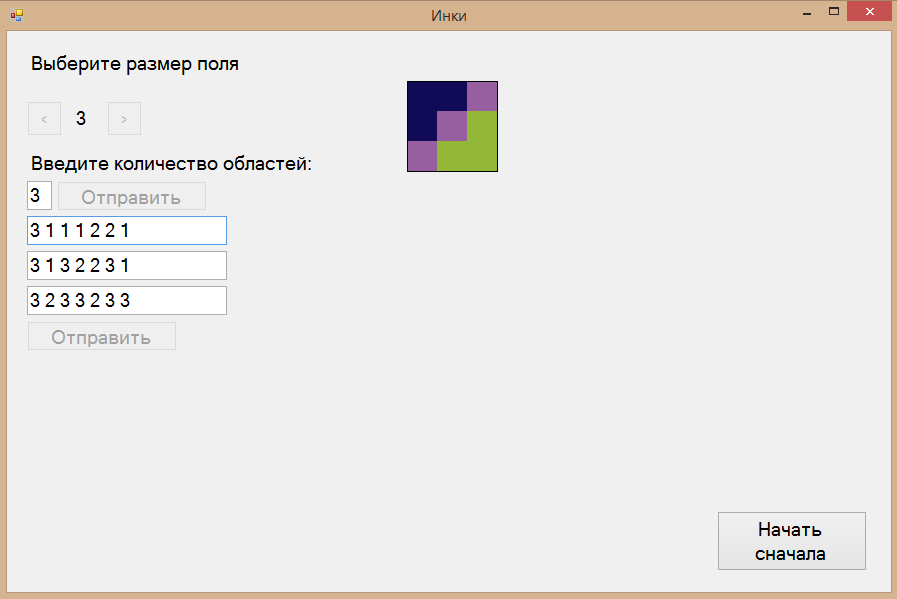


Рис. 4.5 - Результата программы

После нажатия кнопки "Начать сначала" все данные принимают свои начальные значения:

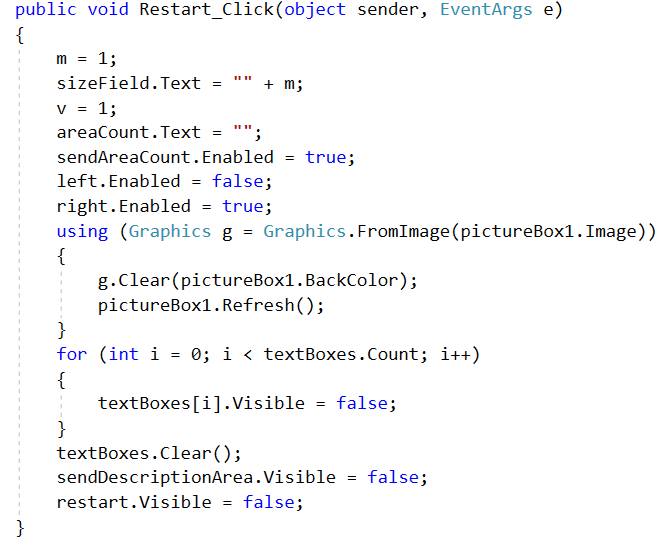


Рис. 4.6

После этого форма приобретает начальный вид, программу можно начать сначала:

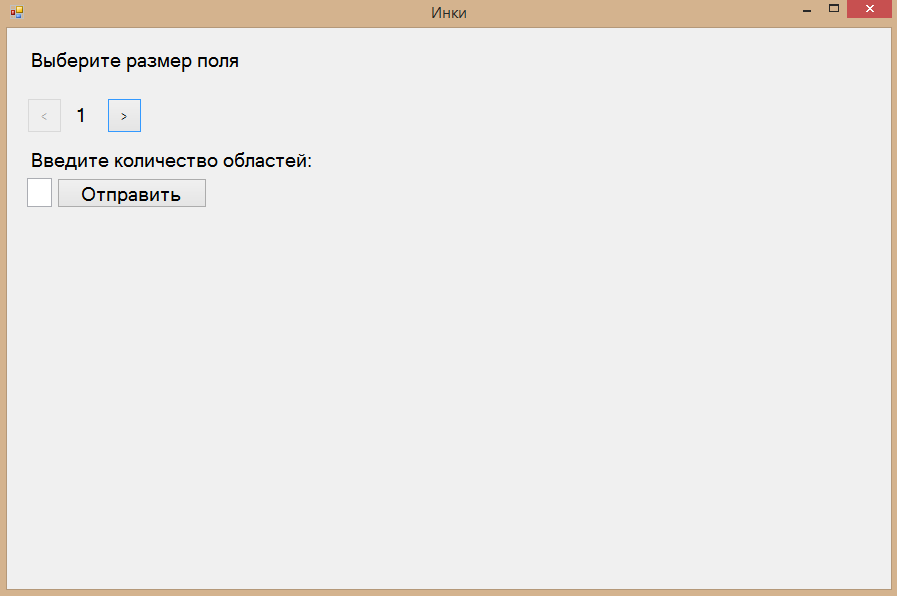
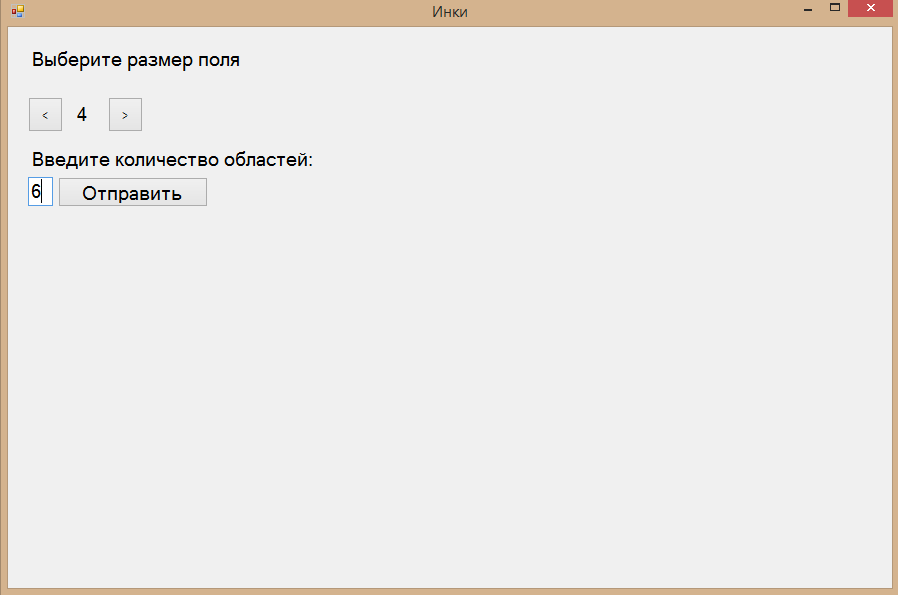
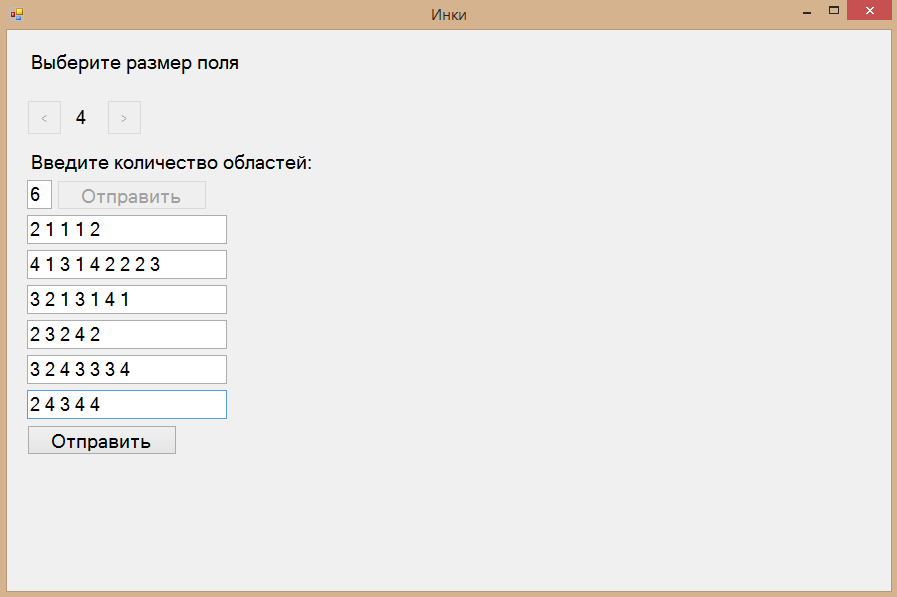


Рис. 4.7 - Начать сначала

# 5. КОНТРОЛЬНЫЙ ПРИМЕР

На рис. 5.1 представлен пример работы приложения:





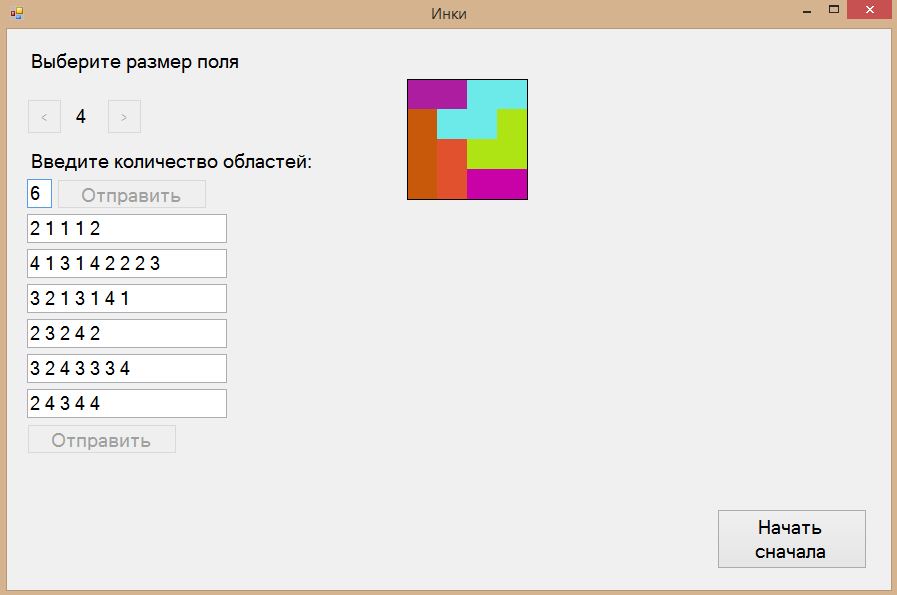


Рис. 5.1 - Работа программы

# ЗАКЛЮЧЕНИЕ

В ходе курсовой работы разработана программа, решающая головоломку типа "Инки".

Для реализации приложения выбрана платформа .Net, технология Windows forms, язык C#. Программный код написан в среде разработки Visual Studio 2019.

# ПРИЛОЖЕНИЕ 1

ТЕКСТ ПРОГРАММЫ

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 WindowsFormsApp4

{

public partial class Form1 : Form

{

static PictureBox pictureBox1 = new PictureBox();

static public int m = 1;

Label size = new Label();

Label sizeField = new Label();

Button left = new Button();

Button right = new Button();

Label areaCountText = new Label();

TextBox areaCount = new TextBox();

Button sendAreaCount = new Button();

static public int v = 1;

List<TextBox> textBoxes = new List<TextBox>();

Button sendDescriptionArea = new Button();

Graphics g = pictureBox1.CreateGraphics();

Random rnd = new Random(DateTime.Now.Millisecond);

Button restart = new Button();

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

size.Location = new Point(20, 20);

size.Size = new Size(400, 25);

size.Font = new Font(size.Font.Name, 14, size.Font.Style);

size.Text = "Выберите размер поля";

Controls.Add(size);

sizeField.Location = new Point(65, 75);

sizeField.Size = new Size(30, 30);

sizeField.Font = new Font(size.Font.Name, 14, size.Font.Style);

sizeField.Text = "" + m;

Controls.Add(sizeField);

left.Location = new Point(20, 70);

left.Size = new Size(35, 35);

left.Text = "<";

Controls.Add(left);

left.Enabled = false;

right.Location = new Point(100, 70);

right.Size = new Size(35, 35);

right.Text = ">";

Controls.Add(right);

left.Click += new EventHandler(Left\_Click);

right.Click += new EventHandler(Right\_Click);

areaCountText.Location = new Point(20, 120);

areaCountText.Size = new Size(300, 25);

areaCountText.Font = new Font(size.Font.Name, 14, size.Font.Style);

areaCountText.Text = "Введите количество областей:";

Controls.Add(areaCountText);

areaCount.Location = new Point(20, 150);

areaCount.Size = new Size(25, 25);

areaCount.Font = new Font(size.Font.Name, 14, size.Font.Style);

Controls.Add(areaCount);

sendAreaCount.Location = new Point(50, 150);

sendAreaCount.Size = new Size(150, 30);

sendAreaCount.Font = new Font(size.Font.Name, 14, size.Font.Style);

sendAreaCount.Text = "Отправить";

Controls.Add(sendAreaCount);

sendAreaCount.Click += new EventHandler(SendAreaCount\_Click);

restart.Location = new Point(710, 480);

restart.Size = new Size(150, 60);

restart.Font = new Font(size.Font.Name, 14, size.Font.Style);

restart.Text = "Начать сначала";

Controls.Add(restart);

restart.Click += new EventHandler(Restart\_Click);

restart.Visible = false;

}

public void Left\_Click (object sender, EventArgs e)

{

m--;

if (m == 1)

left.Enabled = false;

sizeField.Text = "" + m;

right.Enabled = true;

}

public void Right\_Click (object sender, EventArgs e)

{

m++;

if (m == 10)

right.Enabled = false;

sizeField.Text = "" + m;

left.Enabled = true;

}

public void SendAreaCount\_Click(object sender, EventArgs e)

{

v = Convert.ToInt32(areaCount.Text);

if (v > m \* m)

return;

sendAreaCount.Enabled = false;

left.Enabled = false;

right.Enabled = false;

int x = 20;

int y = 150;

for (int i = 0; i < v; i++)

{

TextBox textBox = new TextBox();

textBox.Location = new Point(x, y + 35);

y += 35;

textBox.Size = new Size(200, 30);

textBox.Font = new Font(size.Font.Name, 14, size.Font.Style);

Controls.Add(textBox);

textBoxes.Add(textBox);

}

sendDescriptionArea.Location = new Point(x, y + 35);

sendDescriptionArea.Size = new Size(150, 30);

sendDescriptionArea.Font = new Font(size.Font.Name, 14, size.Font.Style);

sendDescriptionArea.Text = "Отправить";

Controls.Add(sendDescriptionArea);

sendDescriptionArea.Click += new EventHandler(SendDescriptionArea\_Click);

sendDescriptionArea.Enabled = true;

sendDescriptionArea.Visible = true;

}

public void SendDescriptionArea\_Click(object sender, EventArgs e)

{

pictureBox1.Location = new Point(400, 50);

pictureBox1.Size = new Size(m \* 30 + 1, m \* 30 + 1);

pictureBox1.Image = new Bitmap(pictureBox1.Width, pictureBox1.Height);

Controls.Add(pictureBox1);

for (int i = 0; i < textBoxes.Count; i++)

{

string[] numbers = textBoxes[i].Text.Split(' ');

int k = Convert.ToInt32(numbers[0]);

if (numbers.Length - 1 != k \* 2 || numbers.Length == 0)

return;

sendDescriptionArea.Enabled = false;

using (Graphics g = Graphics.FromImage(pictureBox1.Image))

{

SolidBrush color = new SolidBrush(Color.FromArgb(rnd.Next(256), rnd.Next(256), rnd.Next(256)));

for (int j = 1; j < numbers.Length; j+=2)

{

int x = Convert.ToInt32(numbers[j]);

int y = Convert.ToInt32(numbers[j + 1]);

g.FillRectangle(color, y \* 30 - 30, x \* 30 - 30, y \* 30, x \* 30);

}

}

}

using (Graphics g = Graphics.FromImage(pictureBox1.Image))

{

g.DrawLine(new Pen(Color.Black, 1), 0, 0, m \* 30, 0);

g.DrawLine(new Pen(Color.Black, 1), m \* 30, 0, m \* 30, m \* 30);

g.DrawLine(new Pen(Color.Black, 1), m \* 30, m \* 30, 0, m \* 30);

g.DrawLine(new Pen(Color.Black, 1), 0, m \* 30, 0, 0);

}

restart.Visible = true;

}

public void Restart\_Click(object sender, EventArgs e)

{

m = 1;

sizeField.Text = "" + m;

v = 1;

areaCount.Text = "";

sendAreaCount.Enabled = true;

left.Enabled = false;

right.Enabled = true;

using (Graphics g = Graphics.FromImage(pictureBox1.Image))

{

g.Clear(pictureBox1.BackColor);

pictureBox1.Refresh();

}

for (int i = 0; i < textBoxes.Count; i++)

{

textBoxes[i].Visible = false;

}

textBoxes.Clear();

sendDescriptionArea.Visible = false;

restart.Visible = false;

}

}

}