Цель работы: изучить пользовательские интерфейсы в C# с использованием библиотеки GTK#.

Выполнение: диаграмма классов изображена на рисунке 1

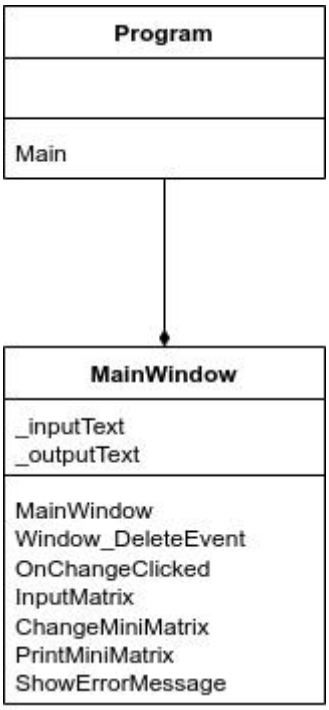

Рисунок 1 — Диаграма классов

Текст программы изображен на рисунках 2-6

```
using System;
using Gtk;

namespace _05
{
    class Program
    {
        [STAThread]
        public static void Main(string[] args)
        {
            Application.Init();

            var app = new Application("org._05._05", GLib.ApplicationFlags.None);
            app.Register(GLib.Cancellable.Current);

            var win = new MainWindow();
            app.AddWindow(win);

            win.Show();
            Application.Run();
        }
    }
}
```

Рисунок 2 — Код класса Program

```
class MainWindow : Window
{
    [UI] private TextView _inputText;
    [UI] private TextView _outputText;

    1 usage  overrides  extension methods  exposing APIs
    public MainWindow() : this(new Builder( resource_name: "MainWindow.glade")){...}

    1 usage  overrides  extension methods  exposing APIs
    private MainWindow(Builder builder) : base(builder.GetRawOwnedObject( name: "MainWindow")){...}

    1 usage  overrides  extension methods  exposing APIs
    private void Window_DeleteEvent(object sender, DeleteEventArgs a){...}

    usages  overrides  extension methods  exposing APIs
    private void OnChangeClicked(object sender, EventArgs e)
    {
        var text = _inputText.Buffer;
        if (text.LineCount % 3 != 0)
        {
            ShowErrorMessage("Количество строк должно быть кратно 3!");
            return;
        }

        var matrix = new int[text.LineCount][];
        if (InputMatrix(matrix) == 1) return;
        ChangeMiniMatrix(matrix);
        PrintMiniMatrix(matrix);
    }

    3 usages
    private void ShowErrorMessage(string message)
    {
        using var md = new MessageDialog(
            parent_window: this,
            DialogFlags.Modal,
            MessageType.Error,
            bt: ButtonsType.Ok,
            message);
        md.Title = "Ошибка!";
        md.Run();
    }
```

Рисунок 3 — Методы класса MainWindow, отвечающие за GTK#

```
private int InputMatrix(int[][] matrix)
{
    var text = _inputText.Buffer;
    var i = 0;
    foreach (var row :string in text.Text.Split("\n"))
    {
        var elemRow :string[] = row.Trim().Split(" ");
        if (elemRow.Length != text.LineCount)
        {
            ShowErrorMessage($"Количество элементов должно быть равно {text.LineCount}!");
            return 1;
        }

        var intRow = new int[text.LineCount];
        for (var j = 0; j < text.LineCount; j++)
        {
            if (!int.TryParse(elemRow[j], out intRow[j]))
            {
                ShowErrorMessage("Ошибка при вводе данных!");
                return 1;
            }
        }

        matrix[i++] = intRow;
    }

    return 0;
}
```

Рисунок 4 — Метод ввода матрицы

```
private void PrintMiniMatrix(int[][] matrix)
{
    _outputText.Buffer.Clear();
    var n :int = _inputText.Buffer.LineCount;
    for (var i = 0; i < n; i++)
    {
        for (var j = 0; j < n; j++)
        {
            _outputText.Buffer.Text += matrix[i][j] + " ";
            if (j == (n / 3 - 1) || j == (n / 3 * 2 - 1))
                _outputText.Buffer.Text += " ";
        }

        _outputText.Buffer.Text += "\n";
        if (i == (n / 3 - 1) || i == (n / 3 * 2 - 1))
            _outputText.Buffer.Text += "\n";
    }
}
```

Рисунок 5 — Метод вывода на экран матрицы

```
private void ChangeMiniMatrix(int[][] matrix)
{
    var n :int  = _inputText.Buffer.LineCount;
    // Меняем матрицы на главной диагонали
    for (var i = 0; i < n / 3; i++)
    {
        for (var j = 0; j < n / 3; j++)
        {
            var temp :int  = matrix[i][j];
            matrix[i][j] = matrix[i + n * 2 / 3][j + n * 2 / 3];
            matrix[i + n * 2 / 3][j + n * 2 / 3] = temp;
        }
    }

    // Меняем матрицы на побочной диагонали
    for (var i :int  = n * 2 / 3; i < n; i++)
    {
        for (var j = 0; j < n / 3; j++)
        {
            var temp :int  = matrix[i][j];
            matrix[i][j] = matrix[i - n * 2 / 3][j + n * 2 / 3];
            matrix[i - n * 2 / 3][j + n * 2 / 3] = temp;
        }
    }
}
```

Рисунок 6 — Метод изменения матрицы

Тестирование программы: окно программы изображено на рисунке 7. Окно обработчика ошибок изображено на рисунке 8

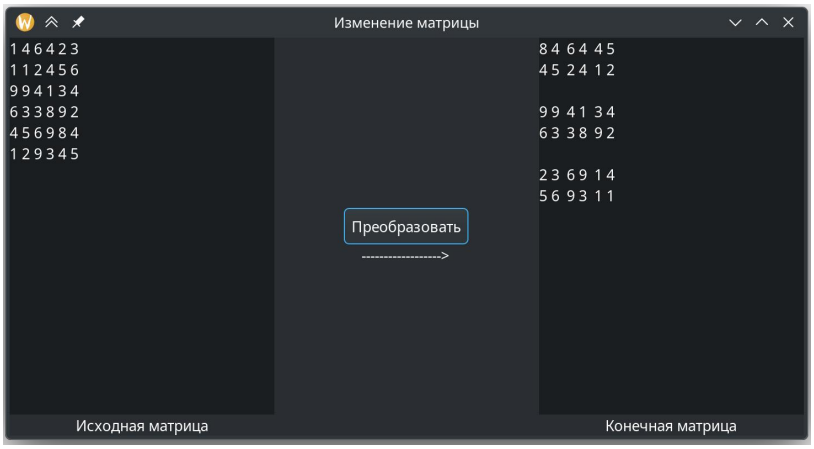

Рисунок 7 — Окно программы

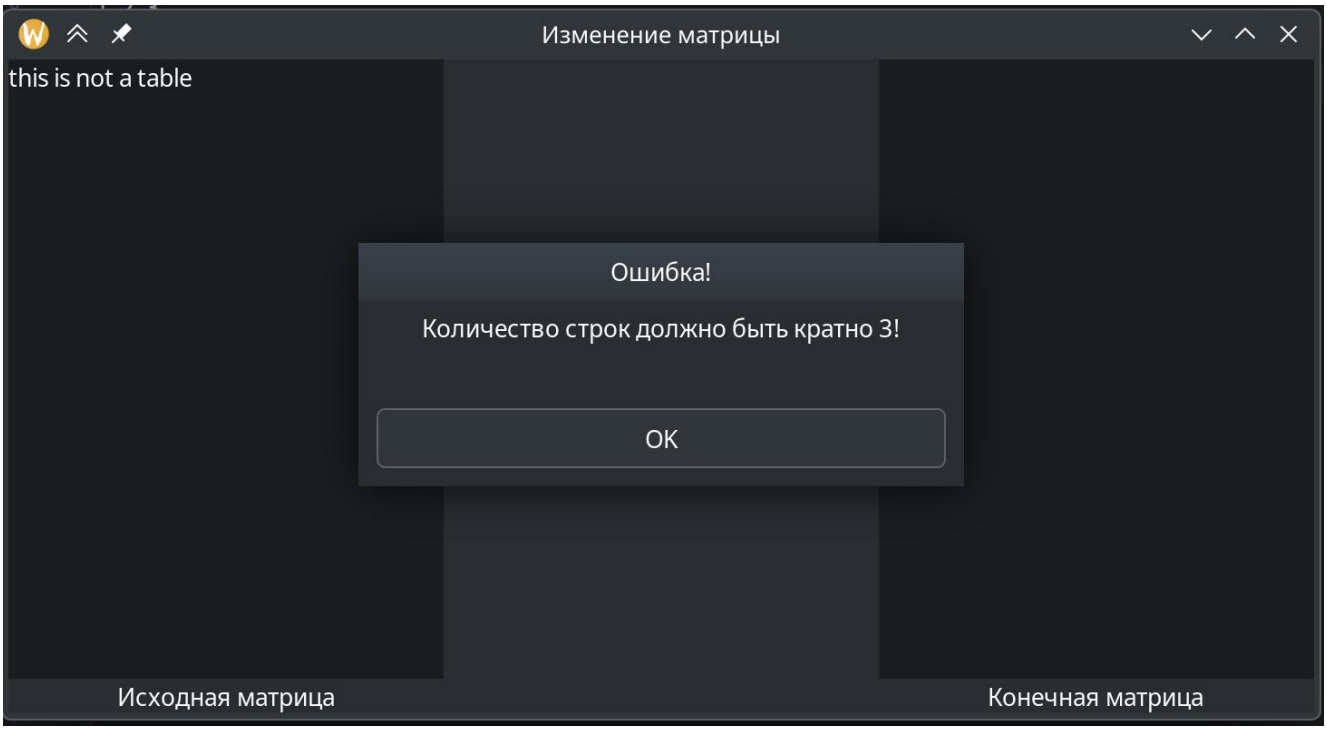

Рисунок 8 — Окно обработки ошибок

Вывод: в данной лабараторной работе был разработан графический интерфейс для прошлой лабараторной работы с использованием библиотеки GTK# и языка C#.