Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«Российский университет транспорта»

(ФГАОУ ВО РУТ(МИИТ), РУТ (МИИТ)

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа № 20

по дисциплине: «Программирование и основы алгоритмизации»

на тему: «Диаграммы»

Выполнил: ст. гр. ТУУ-211

Грачева Наталья Сергеевна

Вариант №5

26.12.2024

(дата выполнения)

Проверил: к.т.н., доц. Сафронов А.И.

(дата приёмки)

Москва – 2024 г.

**1. Цель работы**

Освоить навыки подключения внешних модулей и их использования в разрабатываемых программах; освоить навыки программируемой настройки элемента экранной формы *«Chart»*.

**2. Формулировка задачи**



Загружать данные, представленные на круговых диаграммах, в программное обеспечение из *csv*-файла. Представлены шесть круговых диаграмм. Точное воспроизведение цветовой схемы диаграмм не требуется. Цветовая схема подбирается автором самостоятельно среди доступных средств фреймворка *.NET Framework*, но необходимо предоставить пользователю возможность просмотра как данных с каждой диаграммы в отдельности, так и данных со всех диаграмм сразу (согласно проиллюстрированному порядку). Предусмотреть пользовательскую возможность для последовательного подключения к уже демонстрируемым диаграммам указываемых пользователем, причём порядок размещения определяется выбором очередного названия ряда данных. В этом случае порядок вывода диаграмм на экран будет отличаться от представленного на исходной иллюстрации.

**3. Сети Петри запрограммированного технологического процесса**

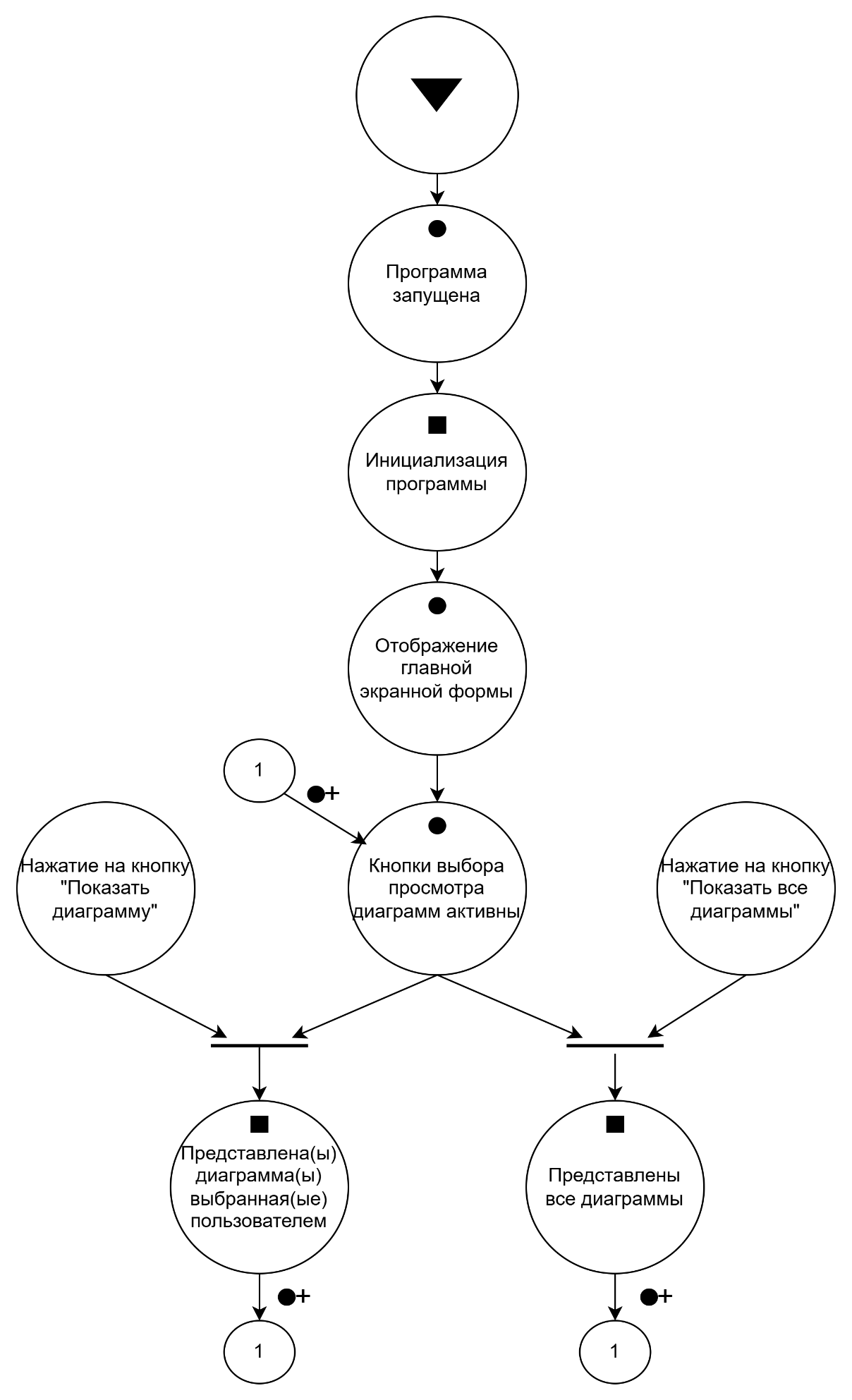


Рисунок 1 – Сеть Петри

**4. Схемы алгоритмов методов в составе решения, отмеченных на сети Петри в качестве «эффектов» (метка )**

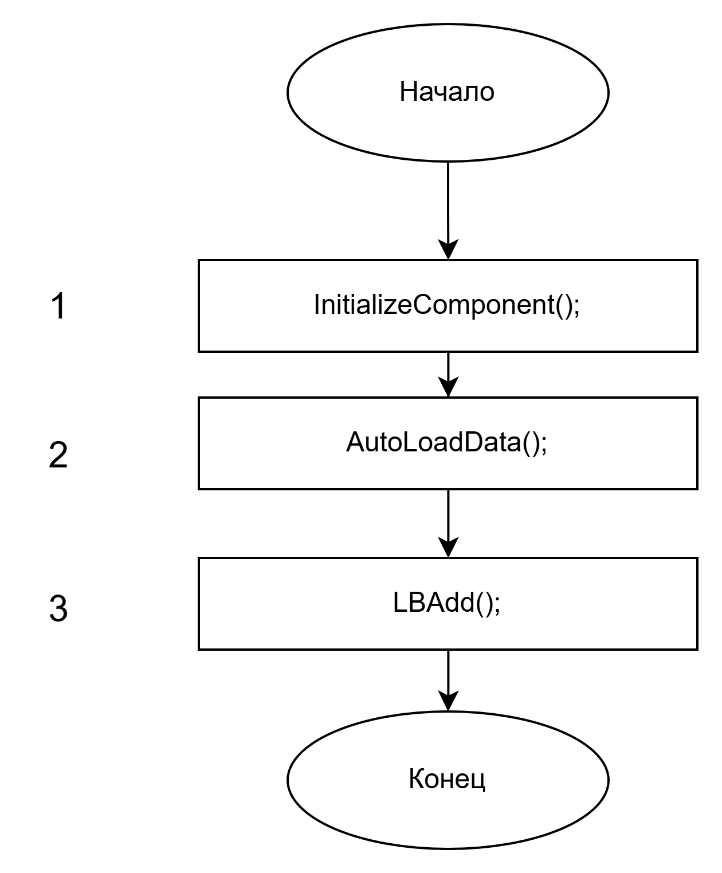


Рисунок 2 – Инициализация программы

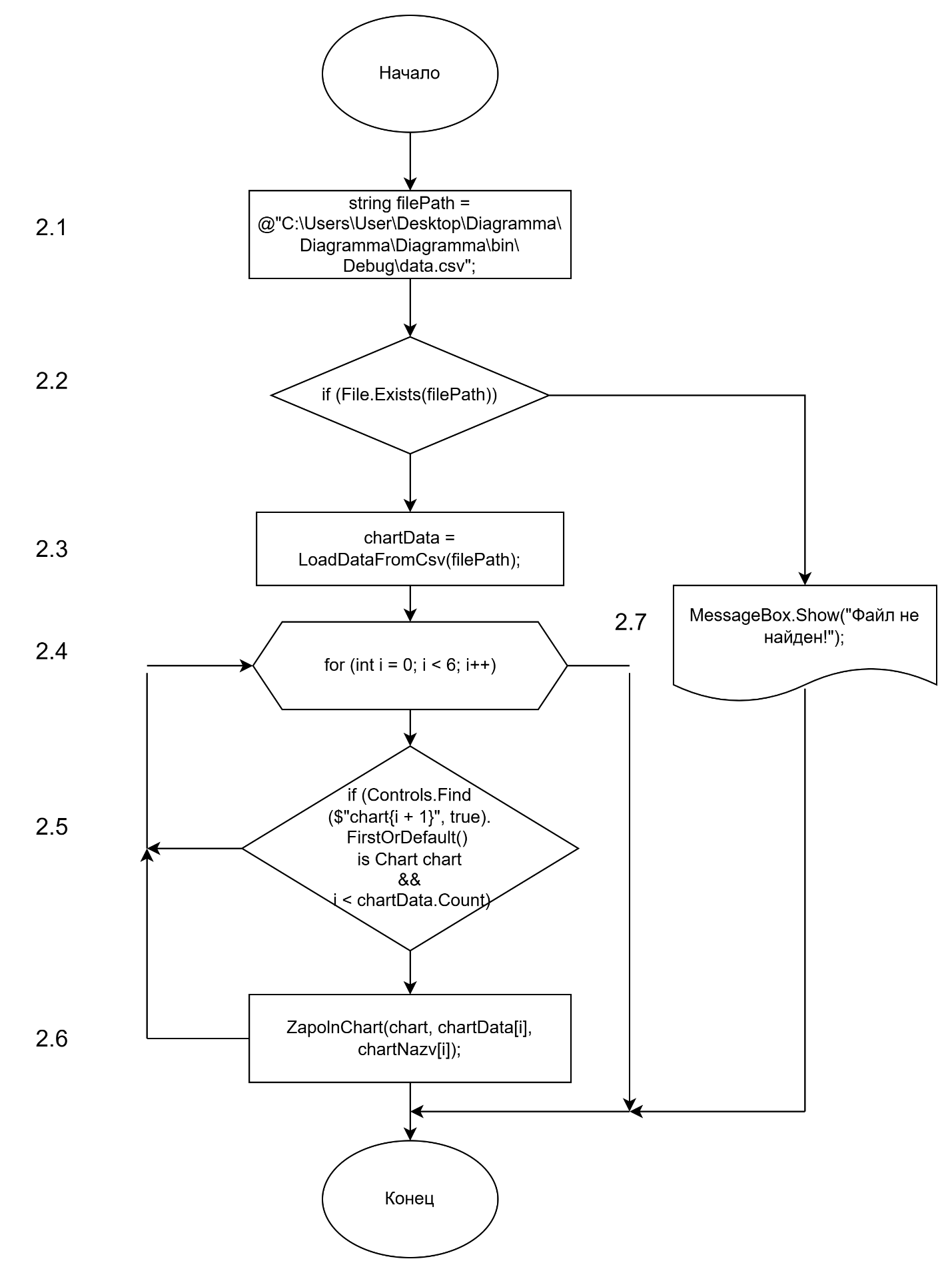


Рисунок 3 – Блок-схема методаAutoLoadData

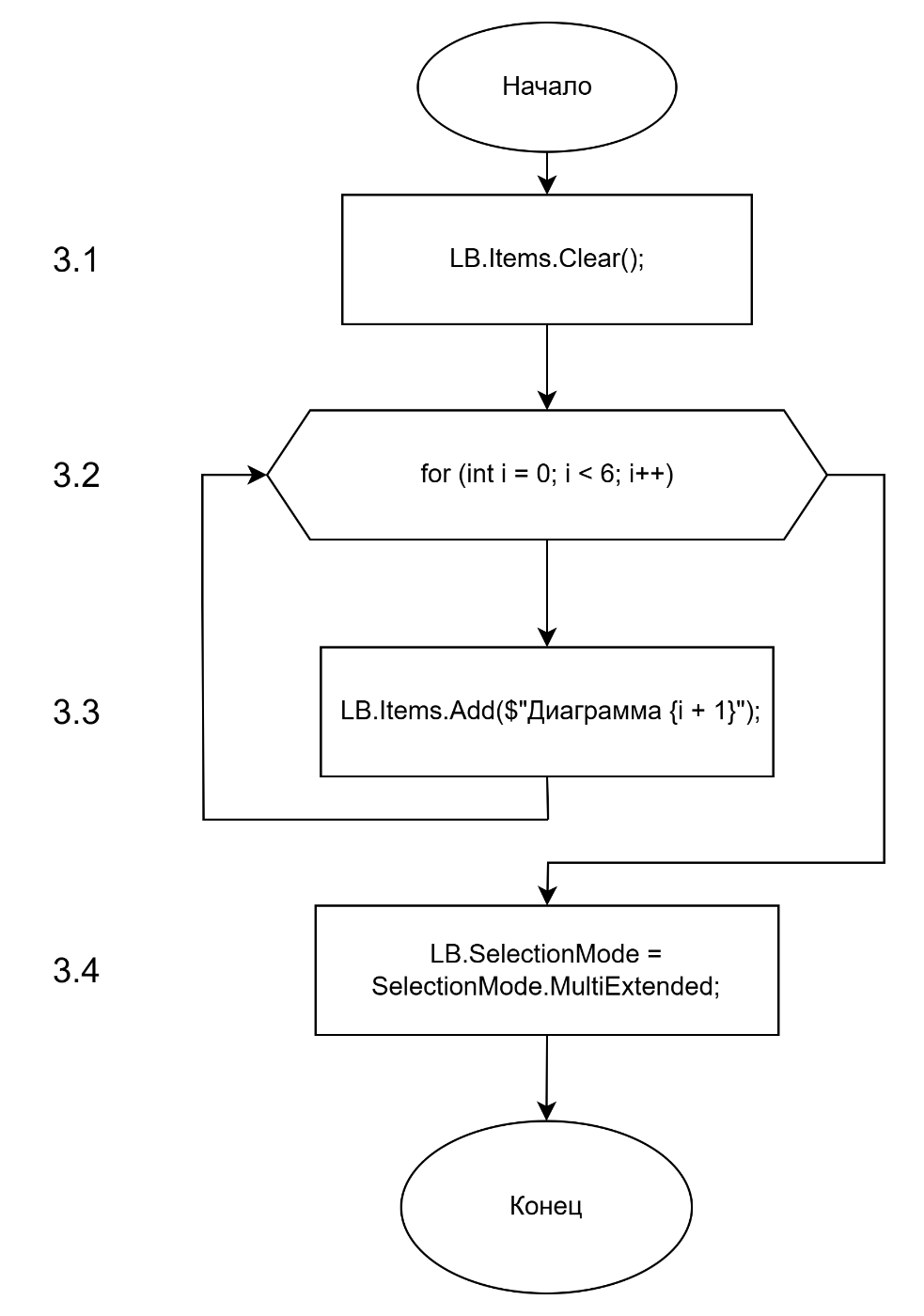


Рисунок 4 – Блок-схема методаLBAdd

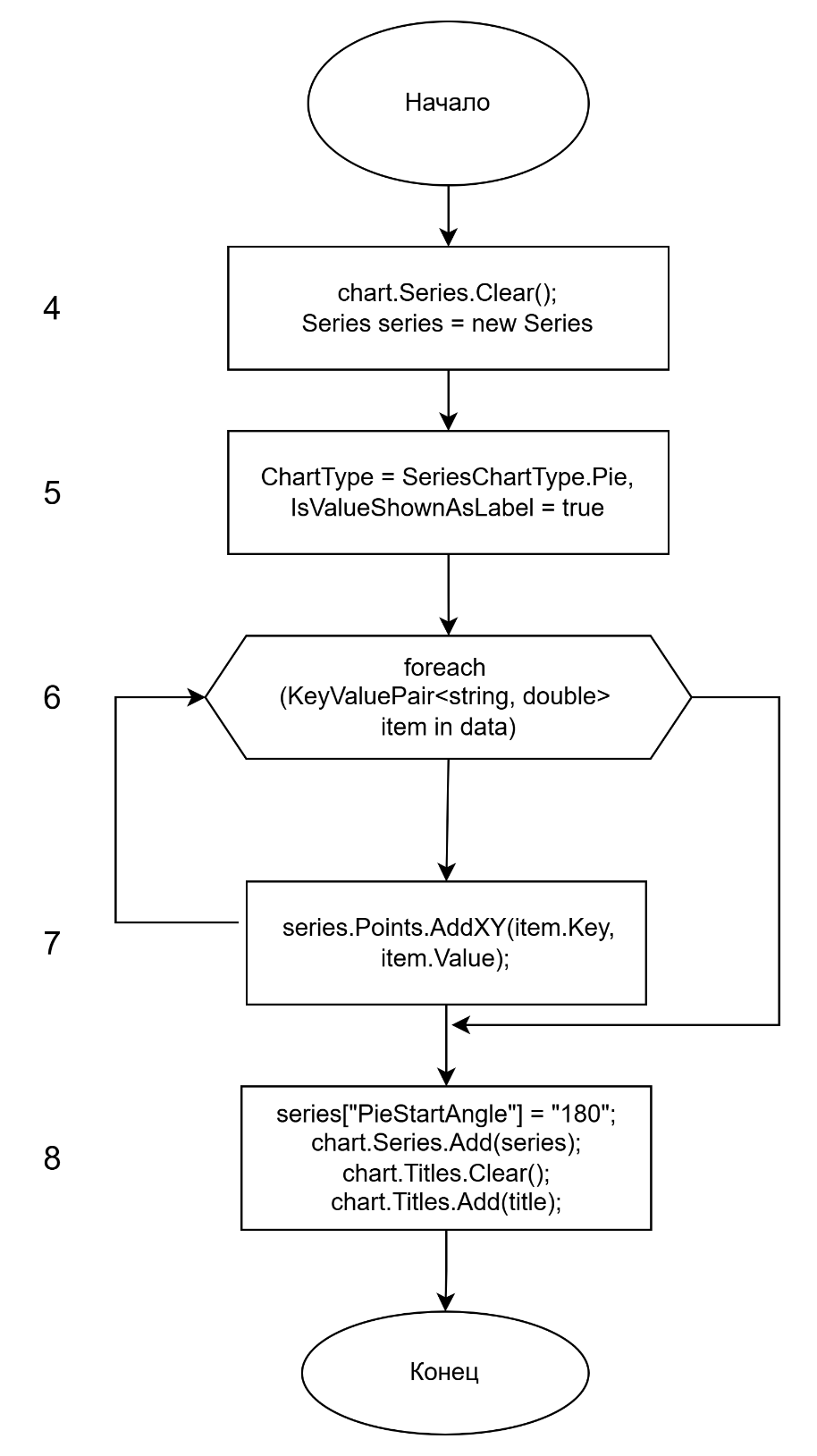
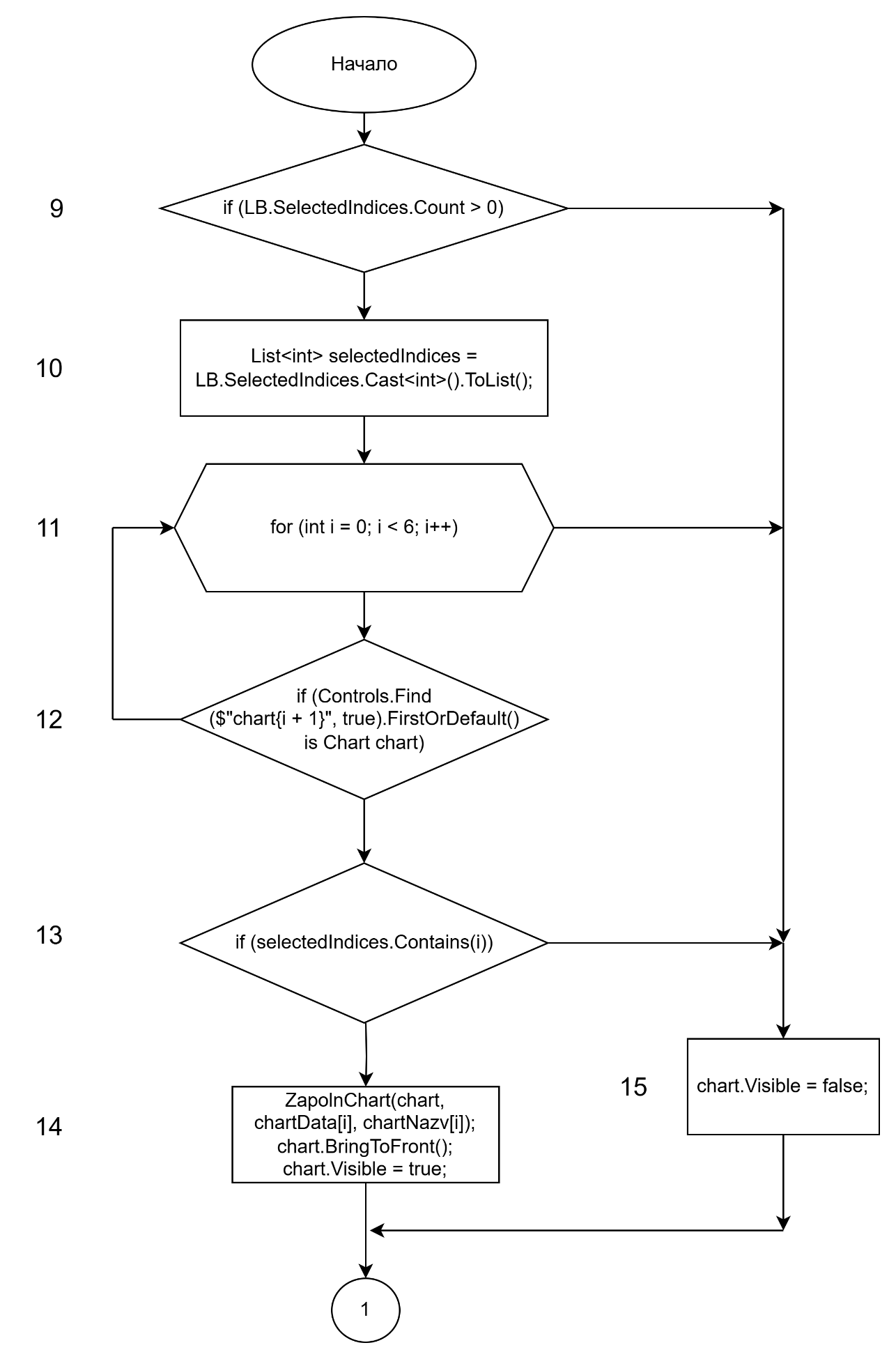


Рисунок 5 – Блок-схема методаZapolnChart



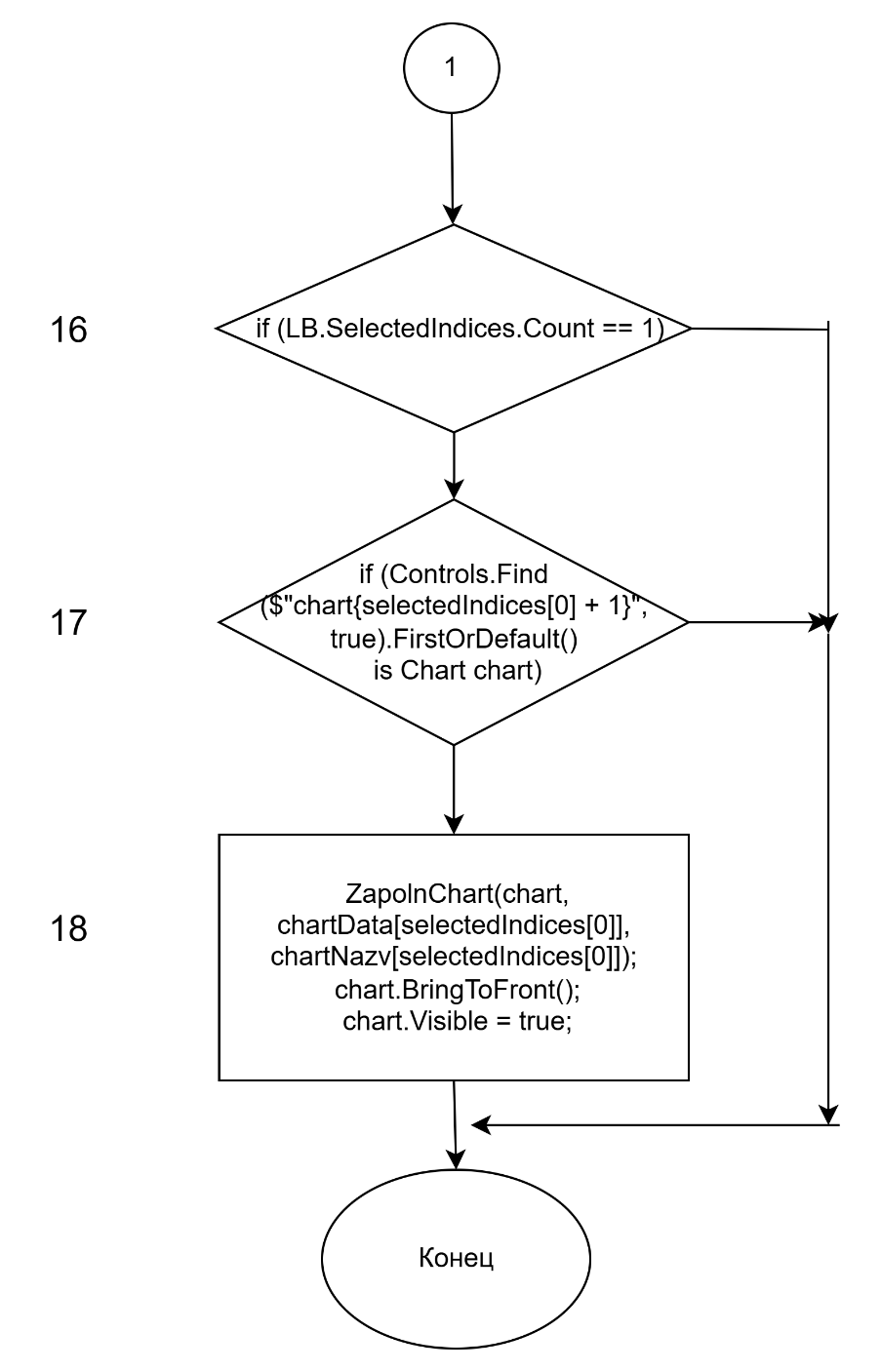


Рисунок 6 – Блок-схема метода ChangeBtn\_Click

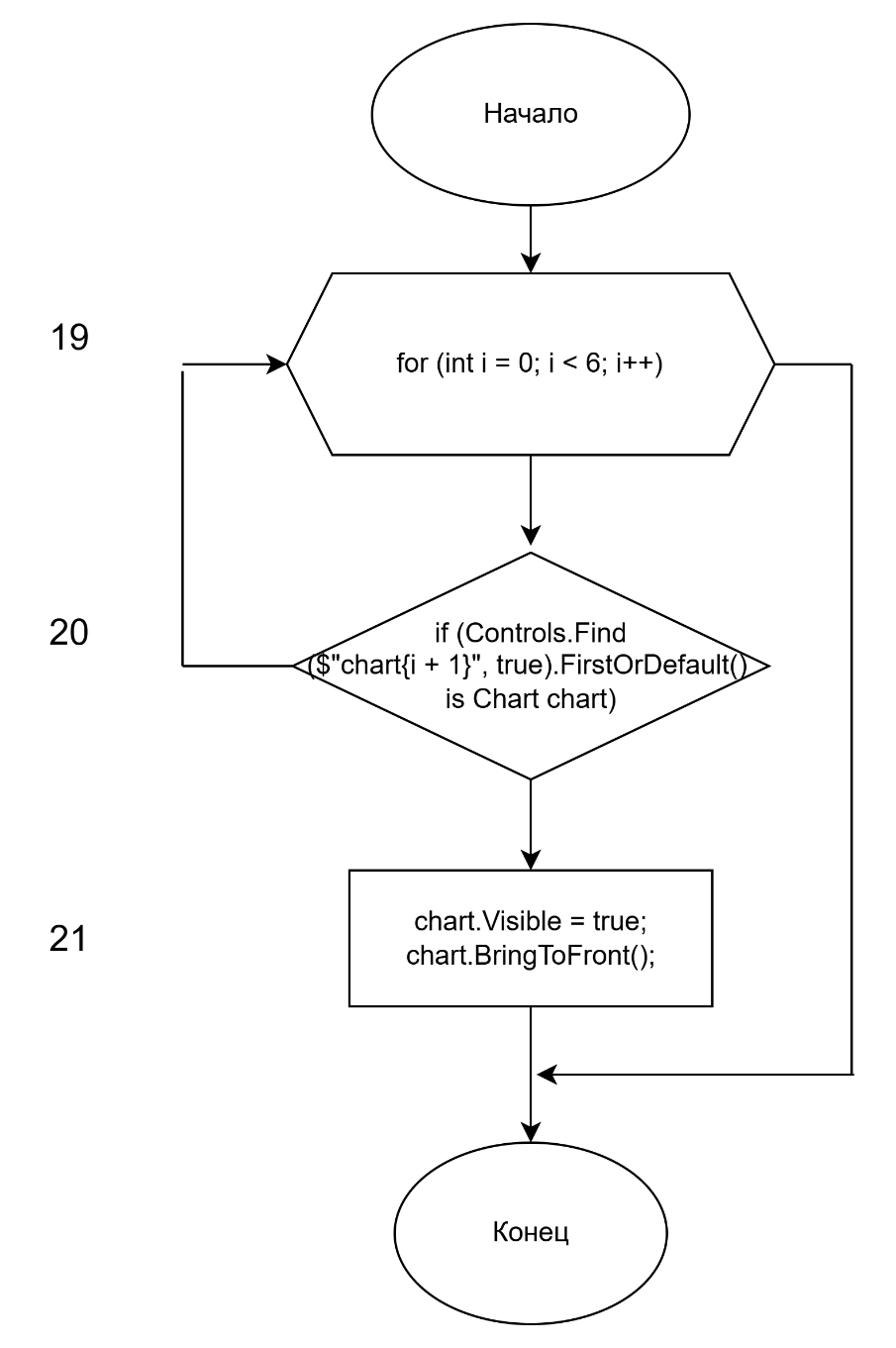


Рисунок 7 – Блок-схема метода ShowAllBtn\_Click

**5. Подбор тестовых примеров**

Пример 1:

Нажатие на кнопку «Показать диаграмму» предоставляет пользователю диаграмму, которую он выбрал.

Пример 2:

Нажатие на кнопку «Показать диаграмму» предоставляет пользователю выбора нескольких диаграмм.

Пример 3:

Нажатие на кнопку «Показать все диаграммы» показывает пользователю все диаграммы.

**6. Листинг (Код) программы**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

namespace Diagramma

{

public partial class MyDiagramma : Form

{

private List<List<KeyValuePair<string, double>>> chartData = new();

private List<string> chartNazv = new();

public MyDiagramma()

{

InitializeComponent();

AutoLoadData();

LBAdd();

}

private void AutoLoadData()

{

string filePath = @"C:\Users\User\Desktop\Diagramma\Diagramma\Diagramma\bin\Debug\data.csv";

if (File.Exists(filePath))

{

chartData = LoadDataFromCsv(filePath);

for (int i = 0; i < 6; i++)

{

if (Controls.Find($"chart{i + 1}", true).FirstOrDefault() is Chart chart && i < chartData.Count)

{

ZapolnChart(chart, chartData[i], chartNazv[i]);

}

}

}

else

{

MessageBox.Show("Файл не найден!");

}

}

private void LBAdd()

{

LB.Items.Clear();

for (int i = 0; i < 6; i++)

{

LB.Items.Add($"Диаграмма {i + 1}");

}

LB.SelectionMode = SelectionMode.MultiExtended;

}

private List<List<KeyValuePair<string, double>>> LoadDataFromCsv(string filePath)

{

List<List<KeyValuePair<string, double>>> data = new List<List<KeyValuePair<string, double>>>();

List<string> titles = new List<string>();

string[] lines = File.ReadAllLines(filePath);

for (int i = 0; i < lines.Length; i++)

{

string[] parts = lines[i].Split(',');

titles.Add(parts[0]);

List<KeyValuePair<string, double>> chartData = new List<KeyValuePair<string, double>>()

{

new("HCO3-", double.Parse(parts[1])),

new("Cl-", double.Parse(parts[2])),

new("Ca2+", double.Parse(parts[3])),

new("Mg2+", double.Parse(parts[4])),

new("Другие элементы", double.Parse(parts[5]))

};

data.Add(chartData);

}

chartNazv = titles;

return data;

}

private void ZapolnChart(Chart chart, List<KeyValuePair<string, double>> data, string title)

{

chart.Series.Clear();

Series series = new Series

{

ChartType = SeriesChartType.Pie,

IsValueShownAsLabel = true

};

foreach (KeyValuePair<string, double> item in data)

{

series.Points.AddXY(item.Key, item.Value);

}

series["PieStartAngle"] = "180";

chart.Series.Add(series);

chart.Titles.Clear();

chart.Titles.Add(title);

}

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

{

if (LB.SelectedIndices.Count > 0)

{

List<int> selectedIndices = LB.SelectedIndices.Cast<int>().ToList();

for (int i = 0; i < 6; i++)

{

if (Controls.Find($"chart{i + 1}", true).FirstOrDefault() is Chart chart)

{

if (selectedIndices.Contains(i))

{

ZapolnChart(chart, chartData[i], chartNazv[i]);

chart.BringToFront();

chart.Visible = true;

}

else

{

chart.Visible = false;

}

}

}

if (LB.SelectedIndices.Count == 1)

{

if (Controls.Find($"chart{selectedIndices[0] + 1}", true).FirstOrDefault() is Chart chart)

{

ZapolnChart(chart, chartData[selectedIndices[0]], chartNazv[selectedIndices[0]]);

chart.BringToFront();

chart.Visible = true;

}

}

}

}

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

{

for (int i = 0; i < 6; i++)

{

if (Controls.Find($"chart{i + 1}", true).FirstOrDefault() is Chart chart)

{

chart.Visible = true;

chart.BringToFront();

}

}

}

}

}

**7. Графический пользовательский интерфейс программного обеспечения и его описание**

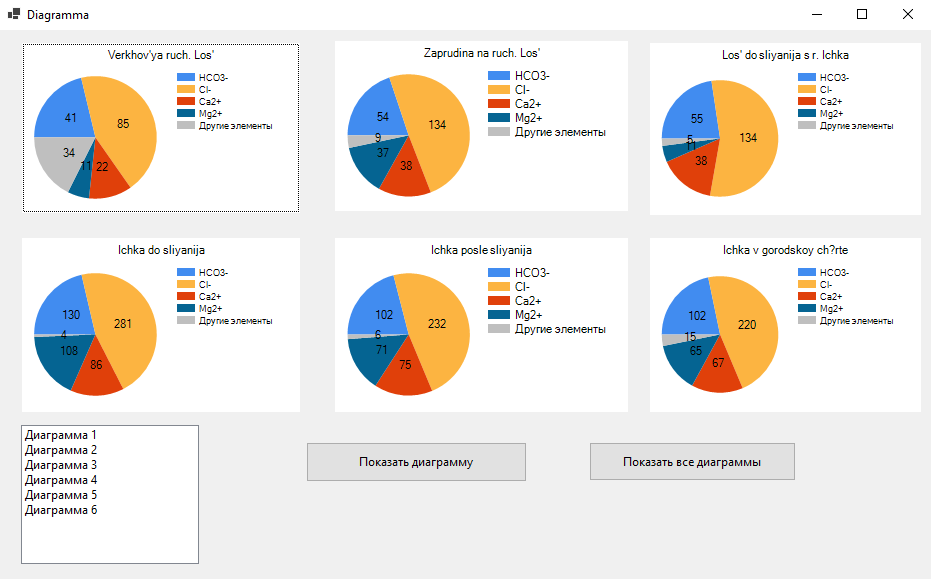
****

Рисунок 7 – Графический пользовательский интерфейс программного обеспечения

Описание:

На форме присутствуют две кнопки: «Показать диаграмму» и «Показать все диаграммы».

Нажатие на кнопку «Показать диаграмму» предоставляет пользователю выбрать одну или несколько диаграмм для просмотра.

Нажатие на кнопку «Показать все диаграммы» предоставляет пользователю все диаграммы.

**8. Подтверждение соответствия графического пользовательского интерфейса требованиям к оформлению**

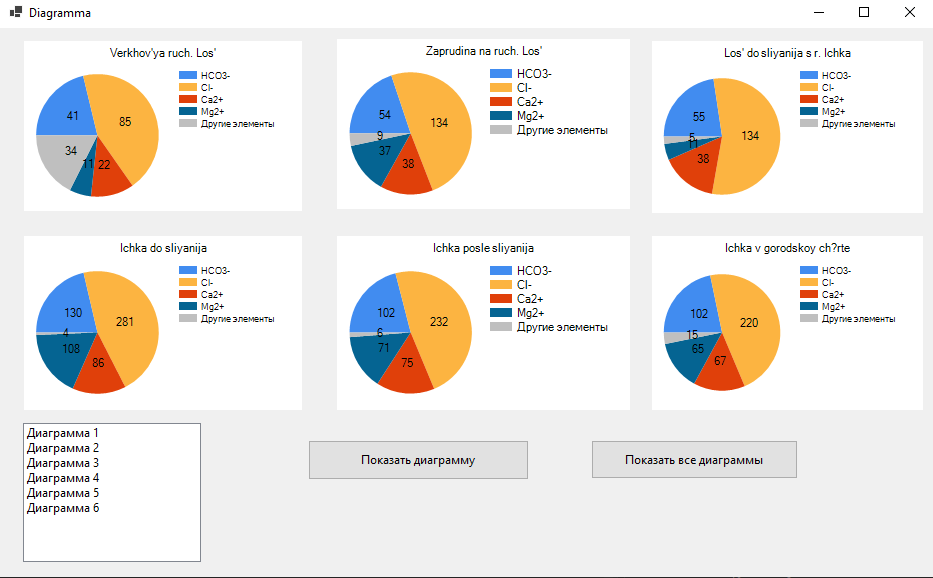


Рисунок 8 – Графический пользовательский интерфейс программного обеспечения

Представлены шесть круговых диаграмм. Точное воспроизведение цветовой схемы диаграмм не требуется. Пользователю предоставлена возможность выбора как одной диаграммы, так и нескольких, а также просмотра всех диаграмм.

**9. Расчёт тестовых примеров** **с использованием составленного программного обеспечения**

Пример 1:

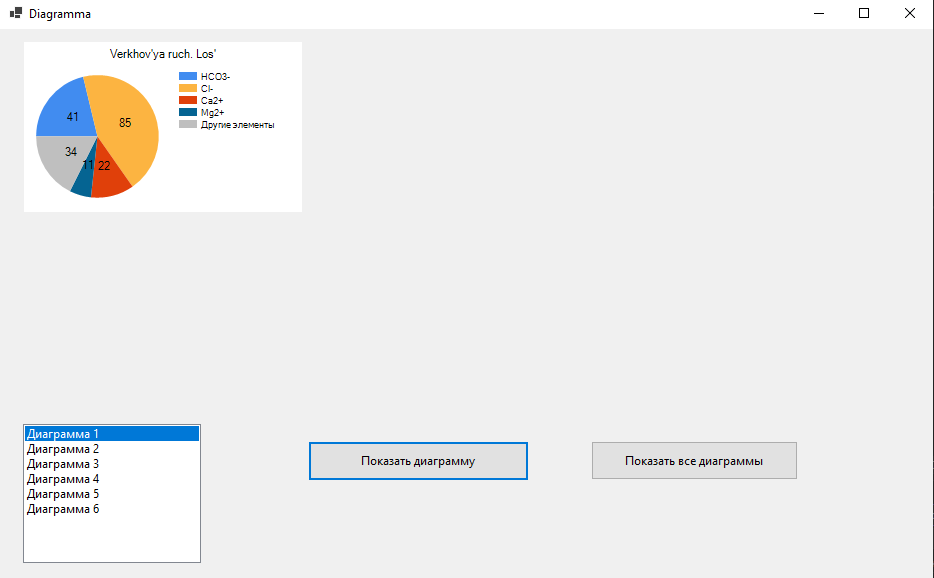
****

Рисунок 9 – Расчёт тестового примера №1 с использованием составленного программного обеспечения

Пример 2:

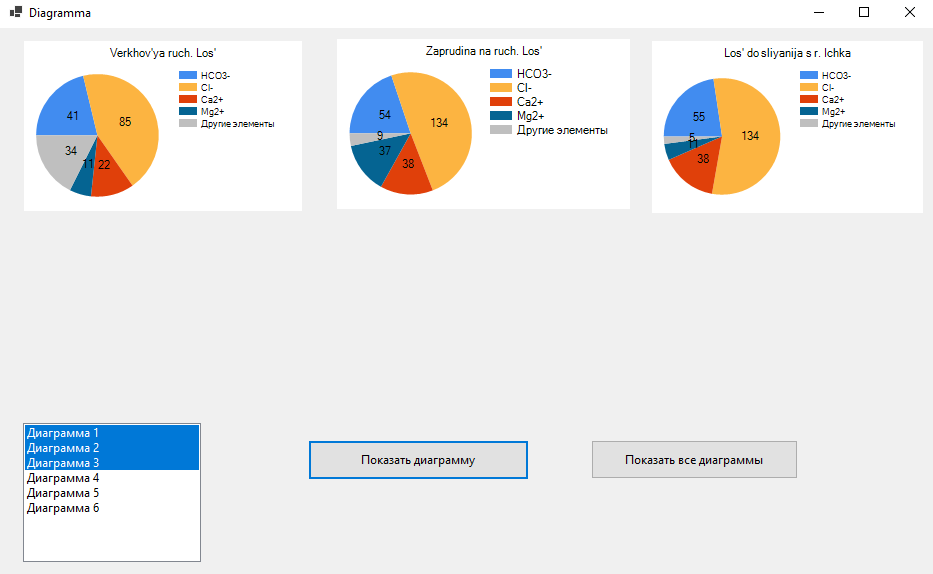
****

Рисунок 10 – Расчёт тестового примера №2 с использованием составленного программного обеспечения.

Пример 3:

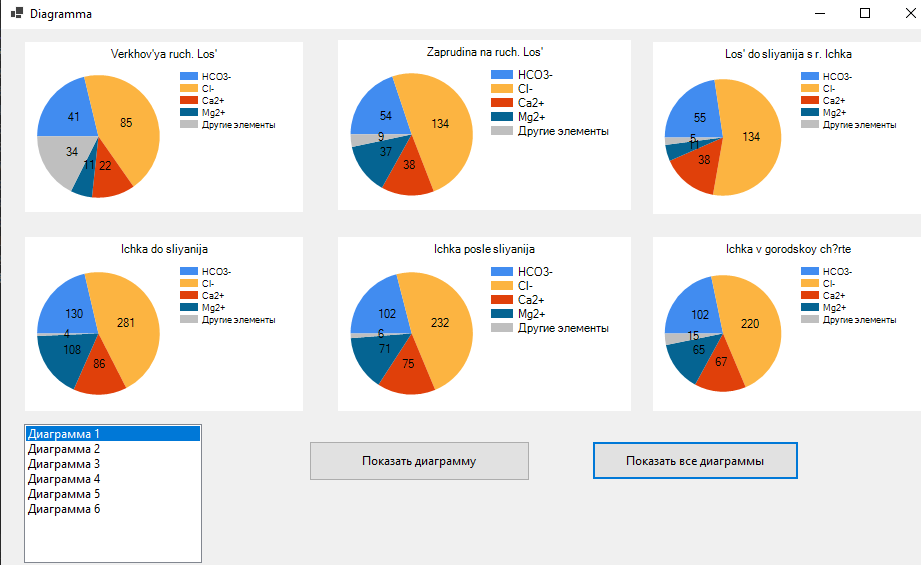


Рисунок 11 – Расчёт тестового примера №3 с использованием составленного программного обеспечения.

**10. Вывод**

В процессе работы освоила работу с элементом экранной формы «*Сhart*», закрепила для себя подключение и использование внешних модулей для работы с диаграммами. В процессе работы я также изучила различные типы диаграмм и их настройки, что расширило мои навыки визуализации данных. Также научилась обрабатывать и представлять данные. Данная работа позволила закрепить теоретические знания и применить их на практике.