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

Баранов А.А.

Вариант №7

09.12.2024

(дата выполнения)

Проверил: к.т.н., доц. Сафронов А.И.

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

(дата приёмки)

Москва – 2024 г.

Оглавление

[1. Формулировка цели работы 3](#_Toc184610187)

[2. Описание задачи 3](#_Toc184610188)

[4. Схемы алгоритмов методов в составе решения, отмеченных на сети Петри в качестве «эффектов» (метка ) 5](#_Toc184610189)

[5. Подбор тестовых примеров 11](#_Toc184610190)

[6. Листинг (код) программного обеспечения 13](#_Toc184610191)

[7. Графический пользовательский интерфейс программного обеспечения 21](#_Toc184610192)

[8. Подтверждение соответствия графического пользовательского интерфейса к требованиям оформления 22](#_Toc184610193)

[9. Расчет тестовых примеров с использованием составленного программного обеспечения 24](#_Toc184610194)

[10. Формулировка вывода о проделанной работе 25](#_Toc184610195)

## **1. Формулировка цели работы**

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

## **2. Описание задачи**

В интегрированной среде разработки *Microsoft Visual Studio* разработать программу в режиме *Windows Forms Application* на языке *Visual C#*, представляющую собой средство взаимодействия с электронными таблицами *Microsoft Office Excel*. Загружать данные, представленные на номограмме, в программное обеспечение из *csv*-файла. Предоставить пользователю возможность просмотра как сразу всех рядов данных, так и одного, указанного ряда данных. Предоставить пользователю возможность просмотра нескольких указанных (не всех) рядов данных. Обязательно формировать легенду для номограммы. Допускается реализация не с чёрно-белым представлением, а с цветным представлением рядов данных.

**3. Сеть Петри запрограммированного технологического процесса**

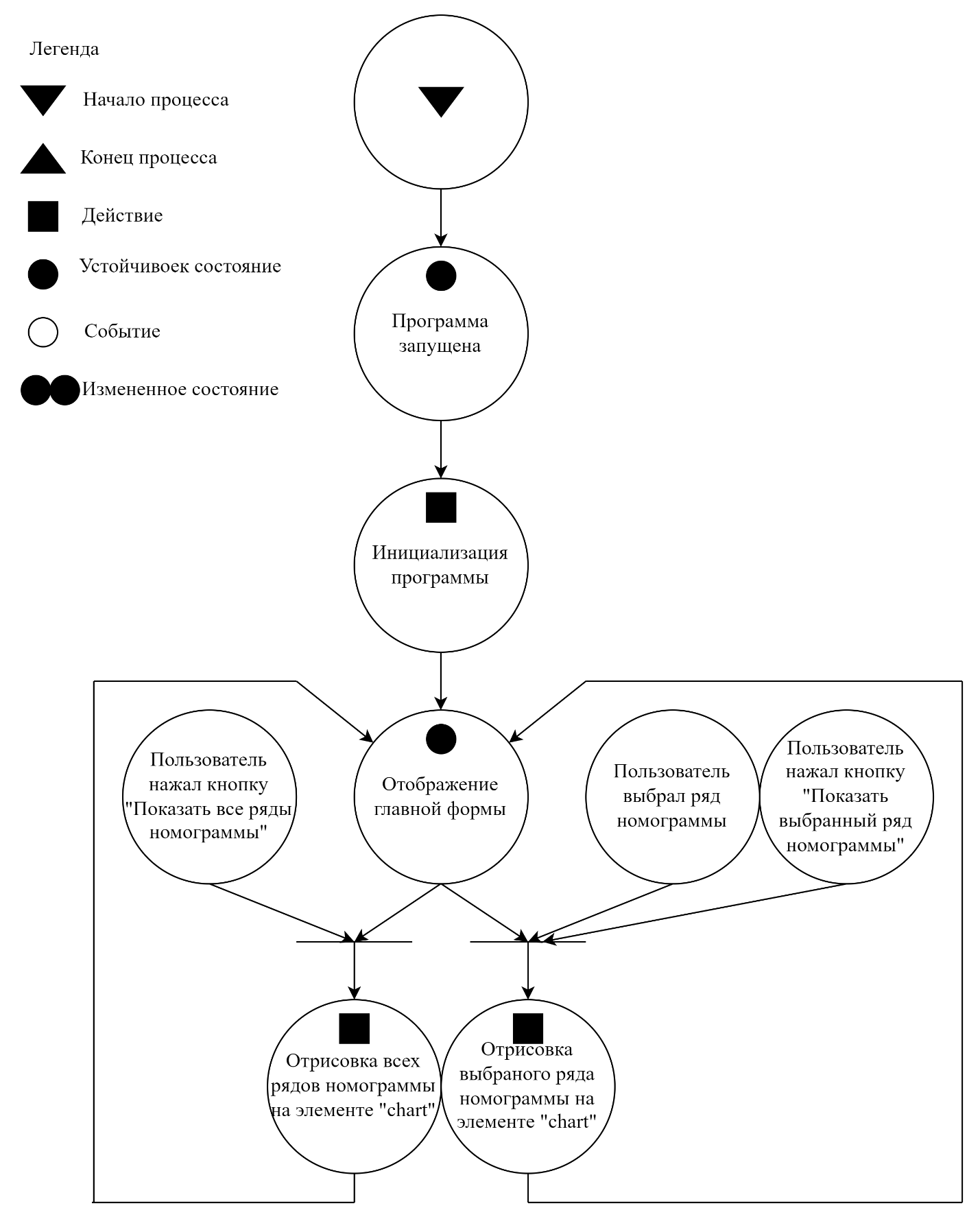


Рисунок 1 – Сеть Петри запрограммированного технологического процесса

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

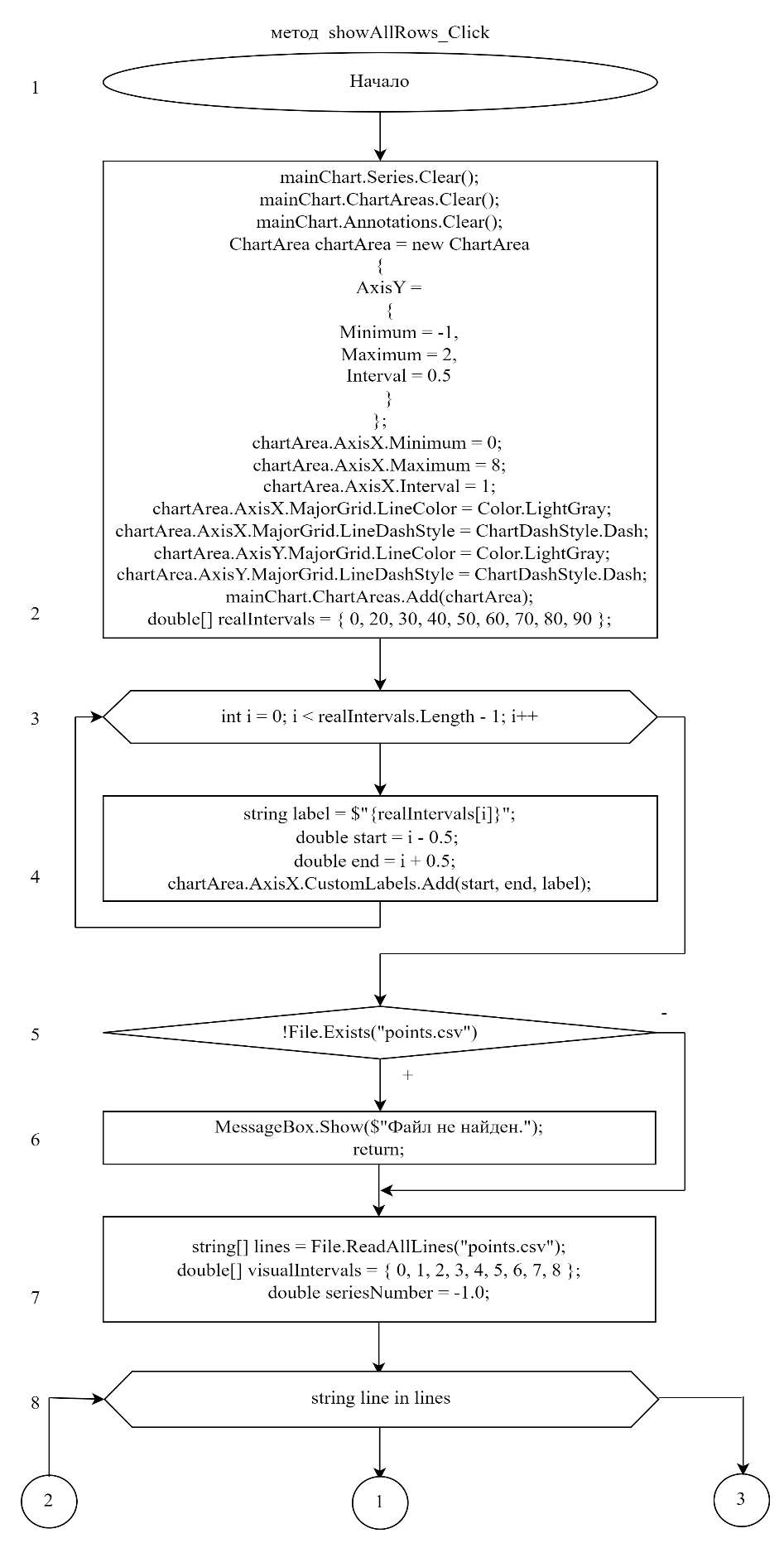


Рисунок 2 – Блок-схема метода *showAllRows\_Click*



Рисунок 3 – Блок-схема метода *showAllRows\_Click*



Рисунок 4 – Блок-схема метода *showSelectedRow\_Click*



Рисунок 5 – Блок-схема метода *showSelectedRow\_Click*

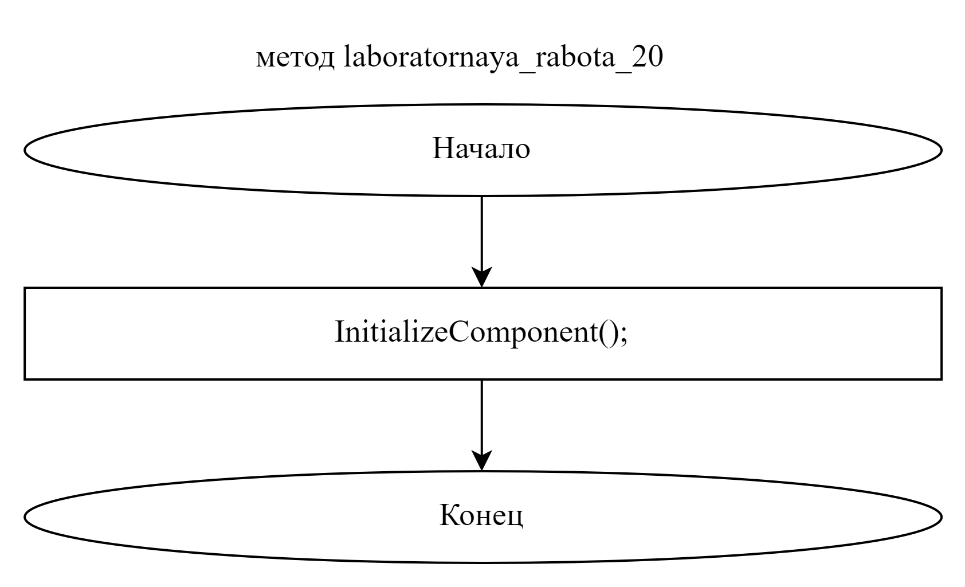


Рисунок 6 – Блок-схема метода *showSelectedRow\_Click*

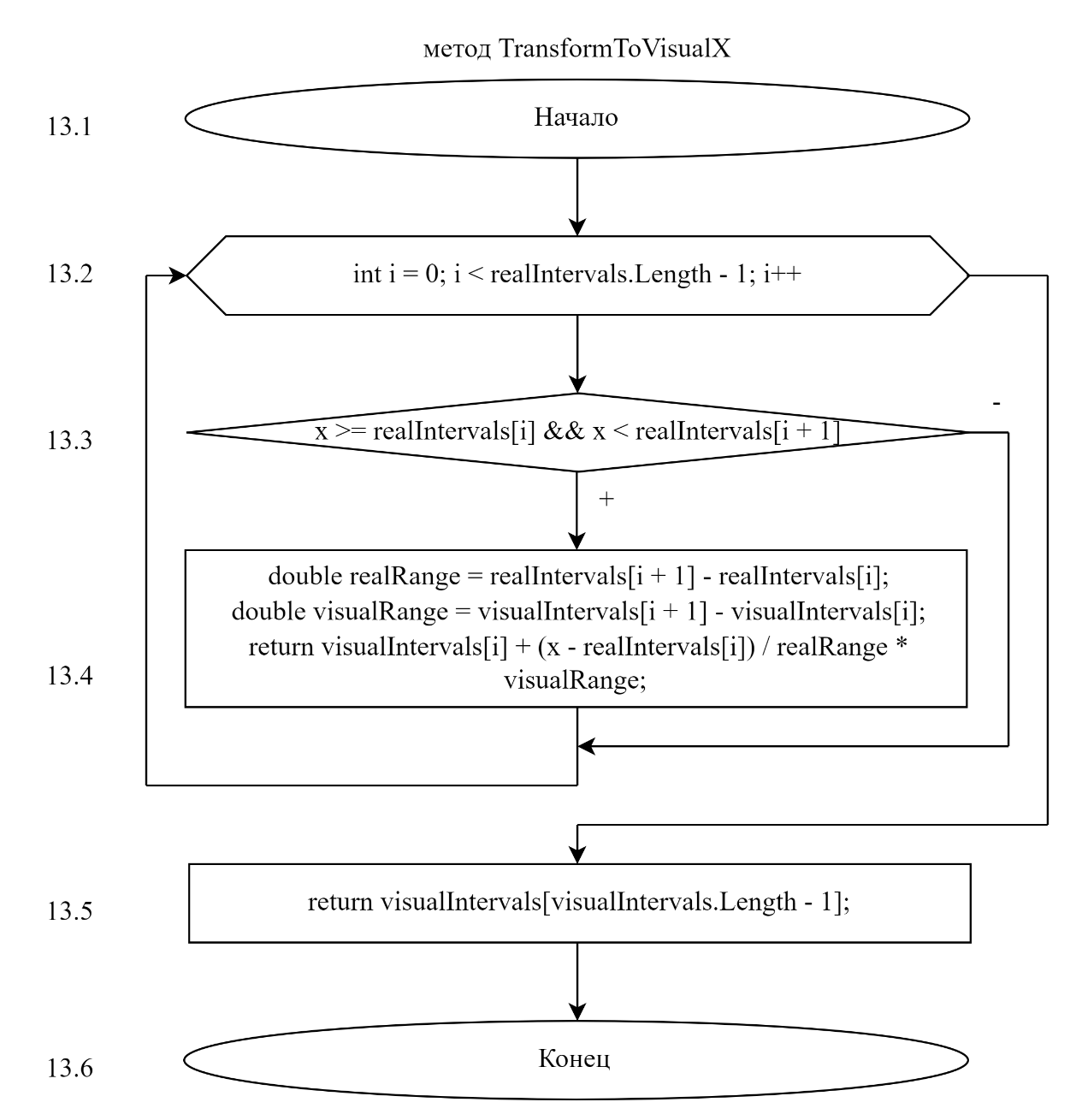


Рисунок 7– Блок-схема метода *TransformToVisualX*

## **5. Подбор тестовых примеров**

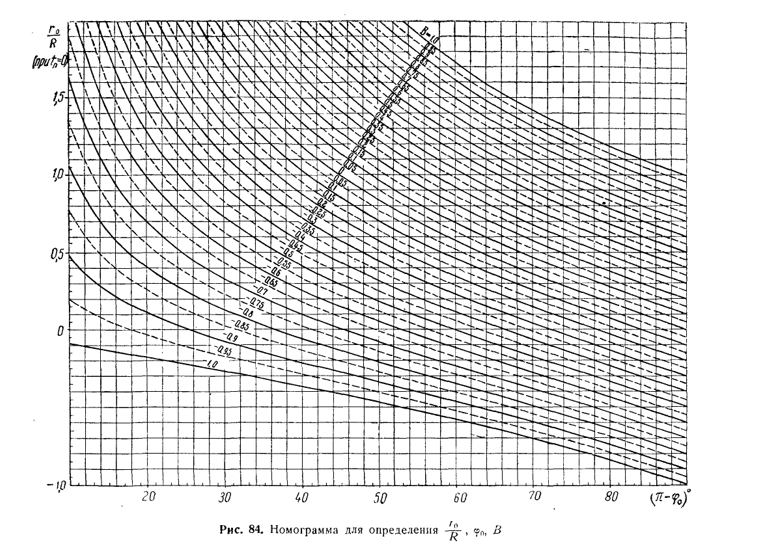


Рисунок 8 – Индивидуальный вариант задания

**Первый тестовый пример**

При нажатии на кнопку «Показать все ряды номограммы», на элементе экранной формы *«chart»* должен появиться график, схожий с рисунком 8. График строится по то точкам, записанным в *CSV*-файле «*points.csv*» (рисунок 9). По договоренности с преподавателем решено оставить 21 ряд номограммы.

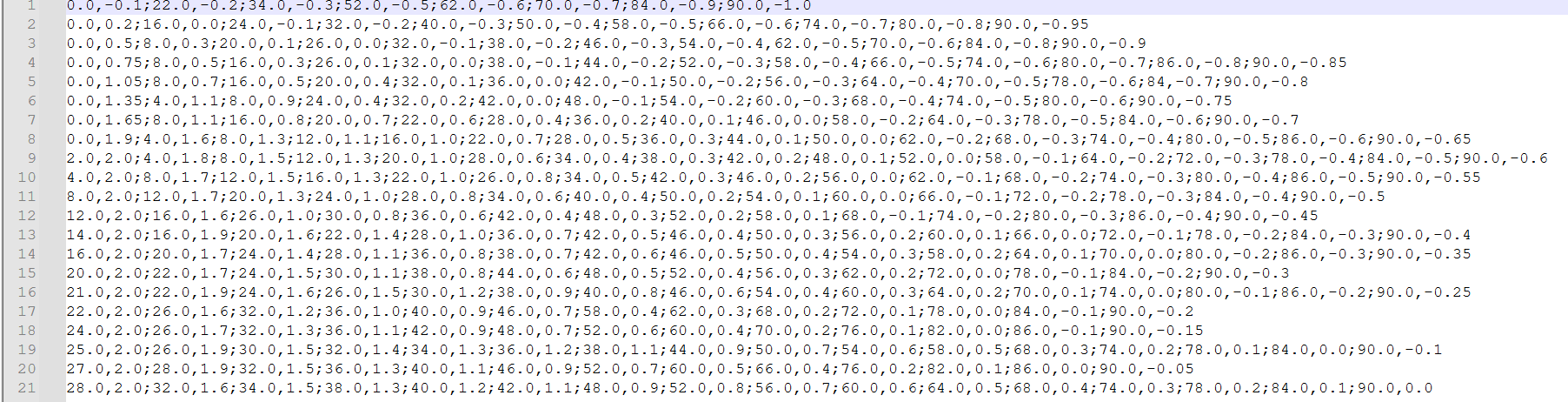


Рисунок 9 – Содержание файла «*points.csv*»

**Второй тестовый пример**

При нажатии на кнопку «Показать выбранный ряд номограммы» и выборе «-0.95» в выпадающем списке, на элементе экранной формы *«chart»* должен появиться второй снизу ряд номограммы, схожий с таким же на рисунке 7. График строится по то точкам из второго ряда *CSV*-файла «*points.csv*» (рисунок 9).

## **6. Листинг (код) программного обеспечения**

using System;

using System.Drawing;

using System.Globalization;

using System.IO;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

namespace laboratornaya\_rabota\_20

{

public partial class laboratornaya\_rabota\_20 : Form

{

public laboratornaya\_rabota\_20()

{

InitializeComponent();

for (double i = -1.0; i <= 0.05; i+=0.05)

{

lineSelector.Items.Add(Math.Round(i, 2).ToString());

}

}

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

{

mainChart.Series.Clear();

mainChart.ChartAreas.Clear();

mainChart.Annotations.Clear();

ChartArea chartArea = new ChartArea

{

AxisY =

{

Minimum = -1,

Maximum = 2,

Interval = 0.5

}

};

chartArea.AxisX.Minimum = 0;

chartArea.AxisX.Maximum = 8;

chartArea.AxisX.Interval = 1;

chartArea.AxisX.MajorGrid.LineColor = Color.LightGray;

chartArea.AxisX.MajorGrid.LineDashStyle = ChartDashStyle.Dash;

chartArea.AxisY.MajorGrid.LineColor = Color.LightGray;

chartArea.AxisY.MajorGrid.LineDashStyle = ChartDashStyle.Dash;

mainChart.ChartAreas.Add(chartArea);

double[] realIntervals = { 0, 20, 30, 40, 50, 60, 70, 80, 90 };

for (int i = 0; i < realIntervals.Length - 1; i++)

{

string label = $"{realIntervals[i]}";

double start = i - 0.5;

double end = i + 0.5;

chartArea.AxisX.CustomLabels.Add(start, end, label);

}

if (!File.Exists("points.csv"))

{

MessageBox.Show($"Файл не найден.");

return;

}

string[] lines = File.ReadAllLines("points.csv");

double[] visualIntervals = { 0, 1, 2, 3, 4, 5, 6, 7, 8 };

double seriesNumber = -1.0;

foreach (string line in lines)

{

Series series = new Series

{

ChartType = SeriesChartType.Line,

BorderWidth = 1,

Color = Color.Black

};

string[] points = line.Split(';');

foreach (string point in points)

{

string[] coordinates = point.Split(',');

if (coordinates.Length == 2 &&

double.TryParse(coordinates[0], NumberStyles.Float, CultureInfo.InvariantCulture, out double x) &&

double.TryParse(coordinates[1], NumberStyles.Float, CultureInfo.InvariantCulture, out double y))

{

double transformedX = TransformToVisualX(x, realIntervals, visualIntervals);

series.Points.AddXY(transformedX, y);

}

}

mainChart.Series.Add(series);

DataPoint thirdPoint = series.Points[2];

TextAnnotation annotation = new TextAnnotation

{

Text = $"{Math.Round(seriesNumber, 2)}",

ForeColor = Color.Red,

Font = new Font("Arial", 7),

AnchorDataPoint = thirdPoint,

Y = thirdPoint.YValues[0] + 0.1

};

mainChart.Annotations.Add(annotation);

seriesNumber+=0.05;

}

}

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

{

mainChart.Series.Clear();

mainChart.ChartAreas.Clear();

mainChart.Annotations.Clear();

ChartArea chartArea = new ChartArea

{

AxisY =

{

Minimum = -1,

Maximum = 2,

Interval = 0.5

}

};

chartArea.AxisX.Minimum = 0;

chartArea.AxisX.Maximum = 8;

chartArea.AxisX.Interval = 1;

chartArea.AxisX.MajorGrid.LineColor = Color.LightGray;

chartArea.AxisX.MajorGrid.LineDashStyle = ChartDashStyle.Dash;

chartArea.AxisY.MajorGrid.LineColor = Color.LightGray;

chartArea.AxisY.MajorGrid.LineDashStyle = ChartDashStyle.Dash;

mainChart.ChartAreas.Add(chartArea);

double[] realIntervals = { 0, 20, 30, 40, 50, 60, 70, 80, 90 };

for (int i = 0; i < realIntervals.Length - 1; i++)

{

string label = $"{realIntervals[i]}";

double start = i - 0.5;

double end = i + 0.5;

chartArea.AxisX.CustomLabels.Add(start, end, label);

}

if (!File.Exists("points.csv"))

{

MessageBox.Show($"Файл не найден.");

return;

}

string[] lines = File.ReadAllLines("points.csv");

int index = lineSelector.SelectedIndex;

string line = lines[index];

Series series = new Series

{

ChartType = SeriesChartType.Line,

BorderWidth = 1,

Color = Color.Black

};

double[] visualIntervals = { 0, 1, 2, 3, 4, 5, 6, 7, 8 };

string[] points = line.Split(';');

foreach (string point in points)

{

string[] coordinates = point.Split(',');

if (coordinates.Length == 2 &&

double.TryParse(coordinates[0], NumberStyles.Float, CultureInfo.InvariantCulture, out double x) &&

double.TryParse(coordinates[1], NumberStyles.Float, CultureInfo.InvariantCulture, out double y))

{

double transformedX = TransformToVisualX(x, realIntervals, visualIntervals);

series.Points.AddXY(transformedX, y);

}

}

mainChart.Series.Add(series);

DataPoint thirdPoint = series.Points[2];

TextAnnotation annotation = new TextAnnotation

{

Text = lineSelector.Items[lineSelector.SelectedIndex].ToString(),

ForeColor = Color.Red,

Font = new Font("Arial", 7),

AnchorDataPoint = thirdPoint,

Y = thirdPoint.YValues[0] + 0.1

};

mainChart.Annotations.Add(annotation);

}

private double TransformToVisualX(double x, double[] realIntervals, double[] visualIntervals)

{

for (int i = 0; i < realIntervals.Length - 1; i++)

{

if (x >= realIntervals[i] && x < realIntervals[i + 1])

{

double realRange = realIntervals[i + 1] - realIntervals[i];

double visualRange = visualIntervals[i + 1] - visualIntervals[i];

return visualIntervals[i] + (x - realIntervals[i]) / realRange \* visualRange;

}

}

return visualIntervals[visualIntervals.Length - 1];

}

}

}

# 

# **7. Графический пользовательский интерфейс программного обеспечения**

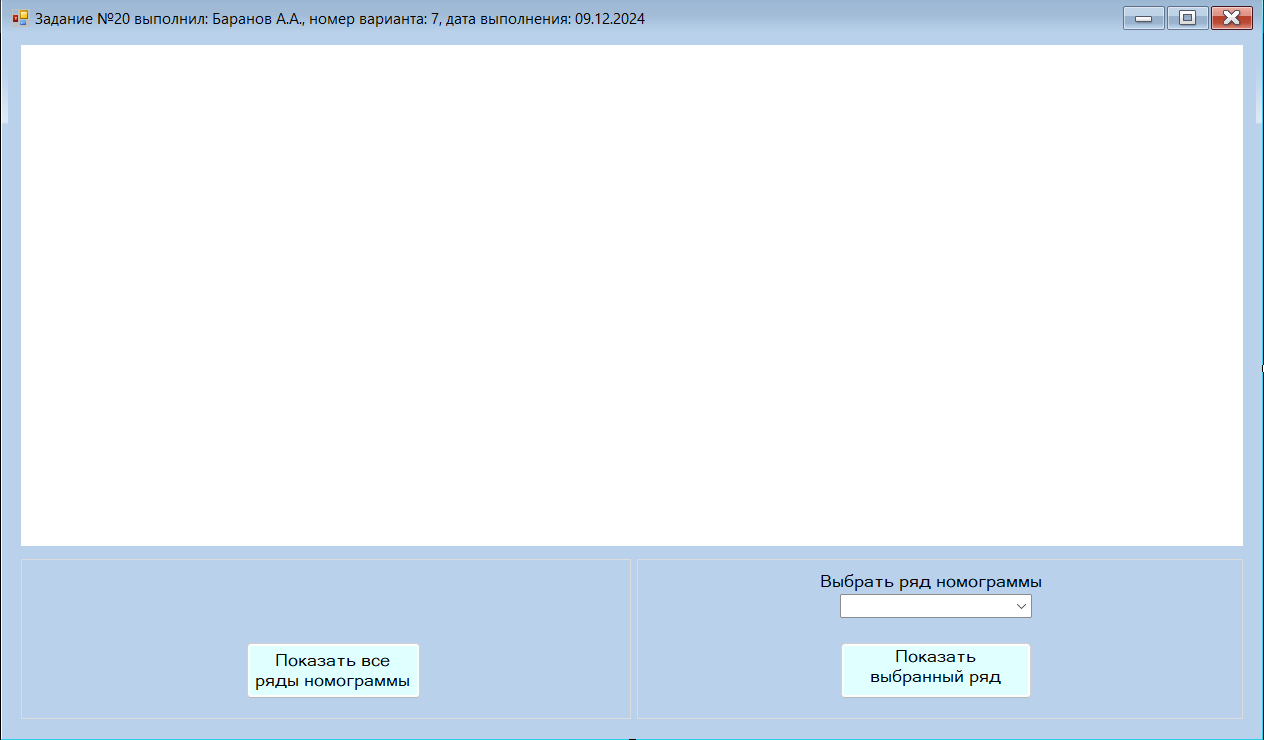


Рисунок 10 – Интерфейс главной формы при запуске

На главной форме располагается элемент экранной формы *«chart»*, кнопки «Показать все ряды номограммы» и «Показать выбранный ряд номограммы», а также выпадающий список для выбора ряда номограммы.

# **8. Подтверждение соответствия графического пользовательского интерфейса к требованиям оформления**

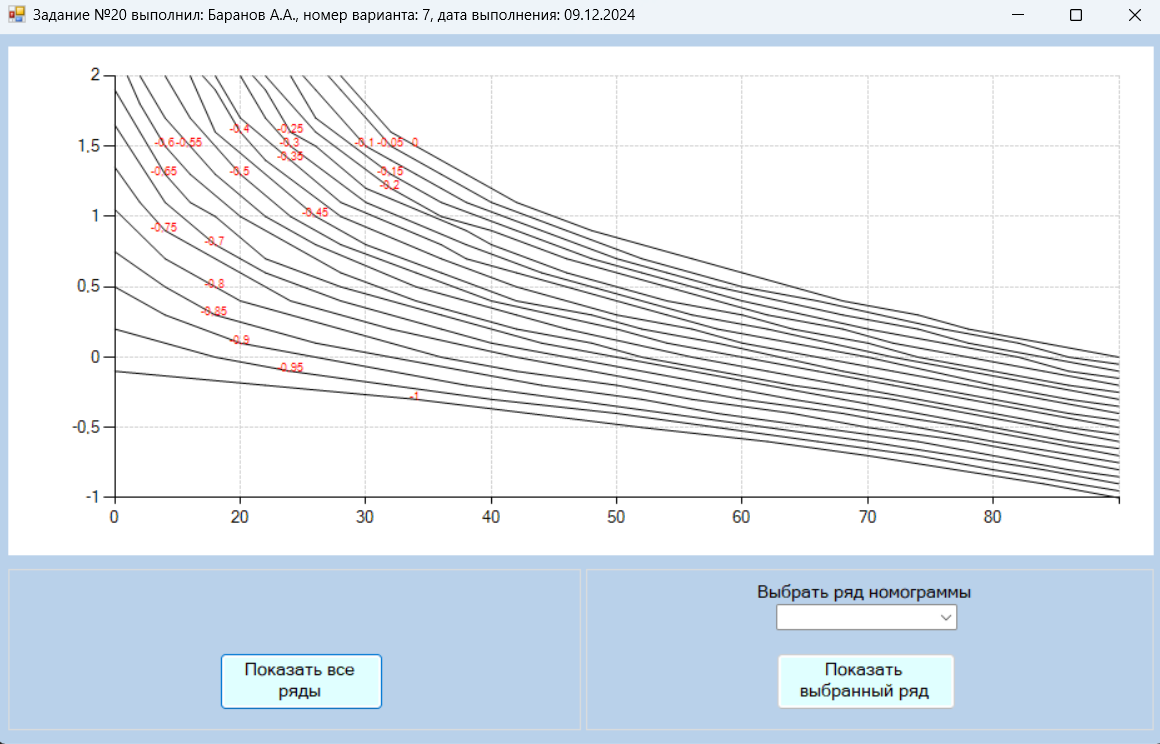


Рисунок 11 – Интерфейс главной формы при нажатии кнопки «Показать все ряды номограммы»

При нажатии на кнопку «Показать все ряды номограммы», элемент экранной формы *«chart»* отражает заданный по индивидуальному заданию рисунок.

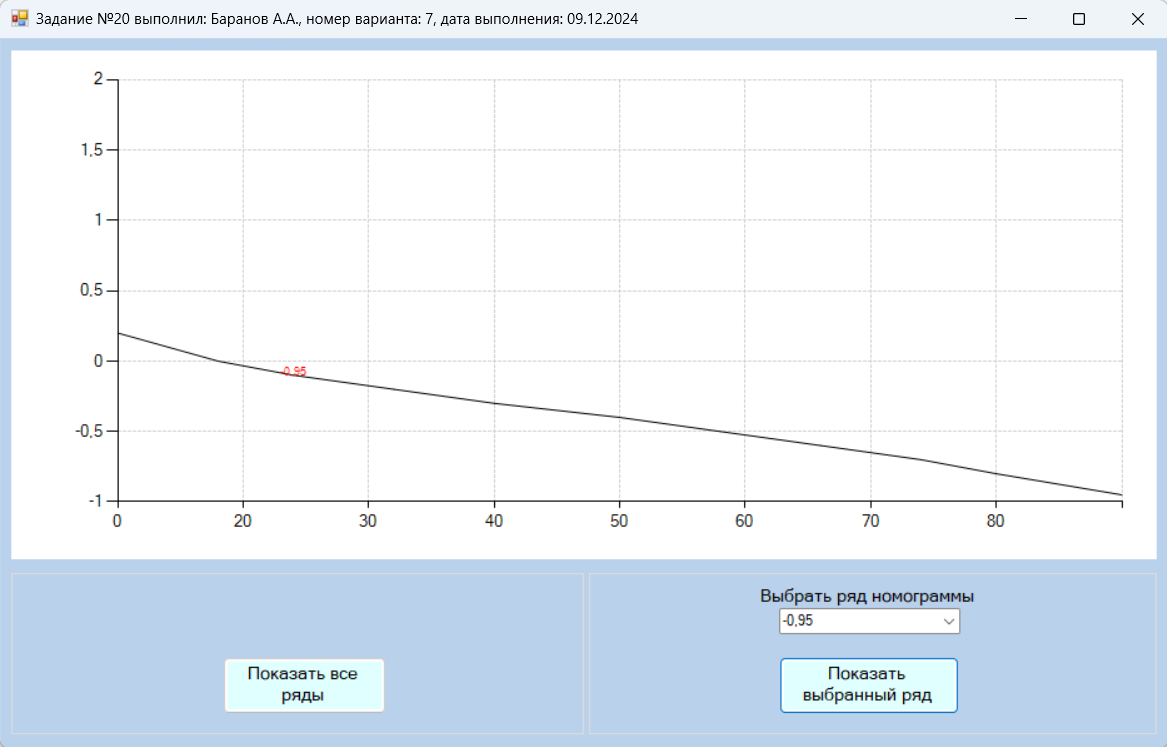


Рисунок 12 – Интерфейс главной формы при нажатии кнопки «Показать выбранный ряд номограммы»

При нажатии на кнопку «Показать все ряды номограммы», элемент экранной формы *«chart»* отражает заданный ряд номограммы.

## **9. Расчет тестовых примеров с использованием составленного программного обеспечения**

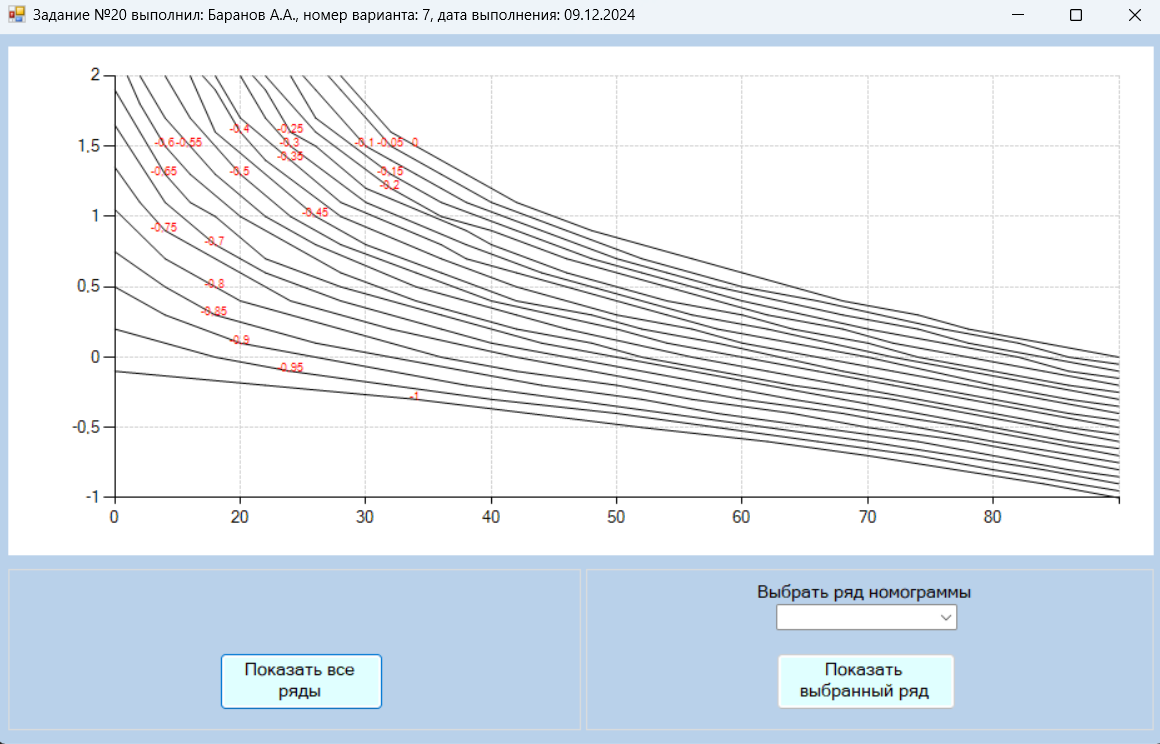


Рисунок 13 – Расчет первого тестового примера

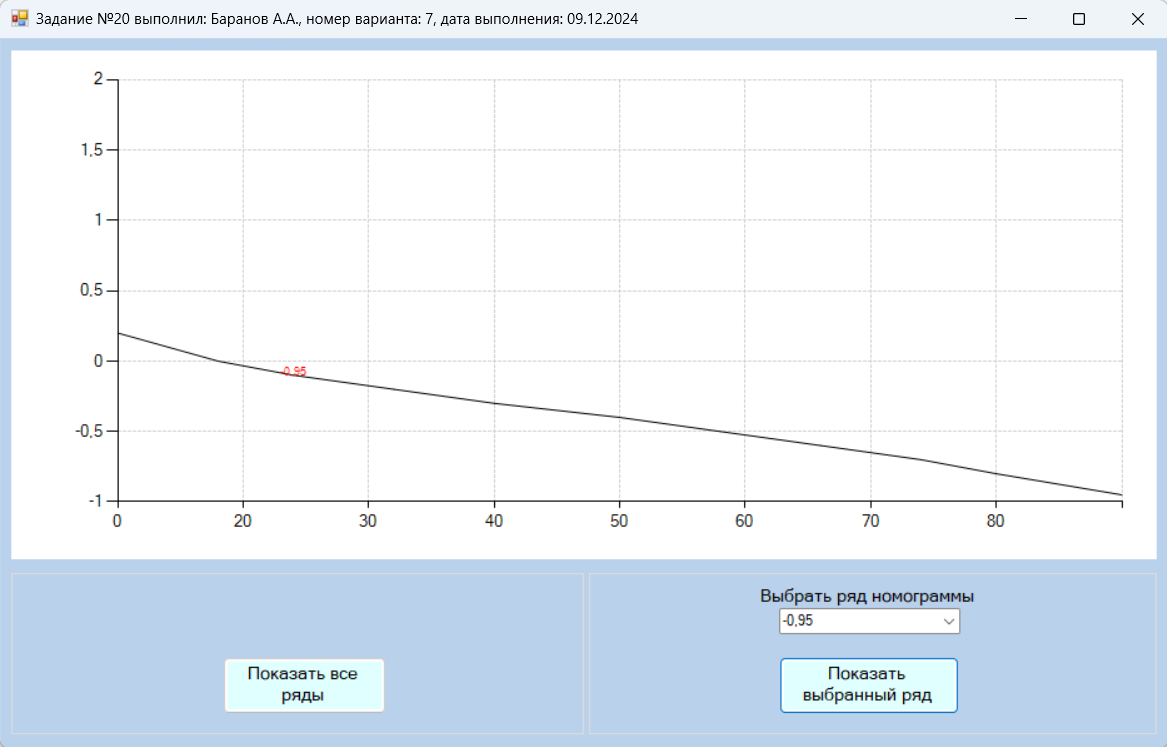


Рисунок 14 – Расчет второго тестового примера

## **10. Формулировка вывода о проделанной работе**

В процессе работы освоил работу с элементом экранной формы *«chart»*, закрепил для себя подключение и использование внешних модулей для работы с диаграммами, закрепил для себя методы работы с файлами, а так же использовал внешний модуль *«Globalization»,* который помог в обработке десятичных чисел из файла «points.csv».