

## СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ И ОБОЗНАЧЕНИЯ .....	3
ВВЕДЕНИЕ .....	4
1 Постановка задачи .....	5
2 Описание иерархии классов и структур .....	6
2.1 Класс «Form1».....	6
2.2 Класс «Form2».....	8
2.3 Класс «Form3».....	8
2.4 Класс «GraphData».....	9
2.5 Структура «nodeinfo» .....	10
2.6 Структура «edgeinfo» .....	11
3 Описание работы программы .....	12
ЗАКЛЮЧЕНИЕ .....	19
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	20
ПРИЛОЖЕНИЕ А .....	21

## ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящей пояснительной записке применяются следующие термины и определения:

.NET – это технология, которая поддерживает создание и выполнение веб-служб и приложений Windows, позволяет использовать одни и те же пространства имён, библиотеки и др. [1]

NuGet – репозиторий с библиотеками, механизм совместного использования кода, поддерживаемым Microsoft [2]

Windows forms – это платформа пользовательского интерфейса для создания классических приложений Windows. Она обеспечивает один из самых эффективных способов создания классических приложений с помощью визуального конструктора в Visual Studio. Такие функции, как размещение визуальных элементов управления путем перетаскивания, упрощают создание классических приложений [3]

Мультиграф – структура данных, состоящая из множества вершин и взвешенных дуг, допускающая наличие нескольких дуг между двумя вершинами, а также существования дуг, ведущих из вершины в ту же вершину.

В данной работе, под понятием граф, будет подразумеваться мультиграф

Форма – это визуальная поверхность Windows forms, на которой выводится информация для пользователя [3]

Элемент управления – это отдельный элемент пользовательского интерфейса, предназначенный для отображения или ввода данных [3]

## ВВЕДЕНИЕ

С текущим развитием технологического прогресса, всё больше и больше данных генерируется, обрабатывается и изучается. Любая программа, будь то мобильное приложение или банковский сервис, работает с информацией, которую необходимо как-то хранить, да ещё так, чтобы она выполнялась в определённо заданное время, при этом чрезмерно не нагружая технику и память.

Математический аппарат в сфере информационного анализа позволил человечеству решать эти проблемы с хранением и обработкой данных. Одна из наиболее распространённых моделей, используемых для таких задач, – это граф. Будь то сервис по навигации, или моделирования сетевого трафика – граф отлично подходит для решения проблем, связанных в конкретной области.

Хорошим навыком для каждого разработчика будет изучения алгоритмов и структур данных. Поэтому создание приложения, основная задача которого заключается в визуализации структур и алгоритмов, является полезным для дальнейшего развития разработчика как специалиста.

Для демонстрации эффективности такого подхода к обучению требуется написать визуализатор графа и алгоритма поиска с использованием объектно-ориентированного языка программирования C# и технологией Windows forms.

Целью данной работы является разработка объектно-ориентированного приложения, реализующего визуализацию графа. Задачами являются составление иерархии классов, а также проработка взаимодействия интерфейса и данных, демонстрация работы программы.

## **1 Постановка задачи**

Разработать программу согласно выбранному варианту с использованием методов разработки «сверху-вниз». Программа должна быть написана на языке C#, быть работоспособной. Разрешается использовать библиотеки из репозитория NuGet [4].

Программа должна построить взвешенный граф по введённому пользователем списку смежности. Пользователь задаёт количество вершин и вводит смежные для каждой из них с расстоянием между ними. Программа строит визуальное представление графа. Вершины графа должны быть подписаны.

Должна быть добавлена возможность поиска кратчайшего пути между выбранными вершинами с визуализацией найденного пути на графе. Скорость поиска должна регулироваться пользователем.

## 2 Описание иерархии классов и структур

Классы можно поделить на следующие группы:

- классы, описывающие граф;
- классы-формы, предоставляющие графический интерфейс.

### 2.1 Класс «Form1»

Форма, реализующая основной интерфейс.

Содержит методы:

- button1\_Click с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для нажатия на кнопку;
- инструкцияToolStripMenuItem\_Click с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для работы с кнопкой меню; по нажатию, открывается инструкция;
- oРазработчикахToolStripMenuItem\_Click с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для работы с кнопкой меню; по нажатию, открывается информация о разработчиках;
- trackBar1\_Changed с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для обработки изменений элемента trackBar;
- Form1\_Load с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для загрузки формы;
- picture\_Reboot с параметрами sender: object, e: EventArgs является событием для обновления изображения формы;
- numericUpDown\_Change с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для обработки изменений значений в элементе numericUpDown;
- button4\_Click с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для нажатия на кнопку;
- StartAlgAsync без параметров; асинхронный метод запуска

алгоритма [5];

- button2\_Click с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для нажатия на кнопку;
- numericUpDown1\_ValueChanged с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для обработки изменений элемента numericUpDown;
- Dispose с параметрами disposing: bool для удаления объекта из памяти;
- InitializeComponent, является методом инициализации полей.

А также класс содержит поля:

- G: GraphData – класс, представляющий собой граф;
- viewer: Microsoft.Msagl.GraphViewerGdi.GViewer – объект для отрисовки графа.

Среди них элементы управления интерфейсом:

- NumericUpDown: numericUpDown1,
- Label: label2,
- TrackBar: trackBar1,
- Label: label3,
- TextBox: textBox2,
- TextBox: textBox3,
- Button: button2,
- Label: label5,
- Button: button4,
- Label: label4,
- HelpProvider: helpProvider1,
- MenuStrip: menuStrip1,
- ToolStripMenuItem: оРазработчикахToolStripMenuItem,
- ToolStripMenuItem: инструкцияToolStripMenuItem,
- Button: button1,

- System.ComponentModel.IContainer: components.

## **2.2 Класс «Form2»**

Форма, реализующая интерфейс всплывающего окна.

Методы класса:

- button1\_Click с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для нажатия на кнопку и выполняет сохранение данных;
- button2\_Click с параметрами sender: object, e: EventArgs, является стандартным событием Windows forms для нажатия на кнопку и выполняет закрытие формы без сохранения данных;
- InitializeComponent, является методом инициализации полей и характеристик формы;
- Dispose с параметрами disposing: bool, для удаления объекта из памяти.

Поля класса:

- Label: label1,
- TextBox: textBox1,
- Button: button1,
- Button: button2,
- HelpProvider: helpProvider1,
- protected GraphData G.

## **2.3 Класс «Form3»**

Форма вывода сообщения.

Класс имеет поля и свойства:

- label1Text, служит установке и получению значения текстового поля формы;
- Label: label1, текстовое поле формы.

Методы класса:

- `InitializeComponent`, является методом инициализации поля и характеристик формы;
- `Dispose` с параметрами `disposing: bool` – для удаления сборщиком мусора.

## 2.4 Класс «`GraphData`»

Класс, реализующий граф.

Содержит методы:

- `NewNode` – метод, добавляющий новую вершину графа;
- `NewNode` – метод, принимающий параметр `Name: string`, добавляющий новую вершину графа с указанным именем;
- `PopNode` – метод, удаляющий вершину снизу списка;
- `GetData` – метод с возвращаемым значением `string`, выдающий список смежностей в виде строки;
- `Intersection` – метод, принимающий параметр `newdata: string`, выполняющий обновление данных (вычисляет пересечение множества вершин из списка смежностей и новых данных);
- `FindShortestWay` – метод, принимающий параметры `A, B: string`, `viewer: Microsoft.Msagl.GraphViewerGdi.GViewer`, `lab: System.Windows.Forms.Label`, выполняющий поиск кратчайшего расстояния между вершинами с указанными именами, отрисовывая на `GViewer`, записывая результат работы в `Label`;
- `FindNodeWithName` – метод, с возвращаемым значением типа `int`, принимающий параметр `name string`, находящий в списке позицию вершины графа с указанным именем;
- `NewEdge` – метод, принимающий параметры `posA: int`, `B: string`, `weigh: int`. Добавляет ребро между вершиной с индексом и вершиной с именем с указанным весом;
- `RemoveWhitespace` – метод с возвращаемым значением `string` с параметром `input: string`, удаляющий все знаки табуляции из строки;



– DeleteNotCommon – метод, с возвращаемым значением типа List<nodeinfo>, принимающий параметр newdata string[], удаляющий из списка смежности все вершины, не совпадающие с поступившими в виде массива строк данными;

– DeleteRemains метод с возвращаемым значением List<nodeinfo> с параметрами deletenodes: List<string>, targetnodes: List<nodeinfo>, удаляющий все рёбра из графа, ведущих из вершин, указанных списке первым параметром;

– int FindEdgeWhithWeight – метод с возвращаемым значением int с параметрами Ainf: nodeinfo, weight: int, находящий индекс ребра в списке смежностей из вершины с указанным весом;

– PaintNodeAndNeighbors – метод с параметрами pos: int, a: Microsoft.Msagl.Drawing.Color, b: Microsoft.Msagl.Drawing.Color, окрашивающий вершину с указанным индексом в первый цвет, а все смежные с ней – во второй цвет.

Содержит поля и свойства:

– List<nodeinfo> nodes – список смежностей графа;

– int nodecounts – свойство получения количества вершин графа;

– bool isalgoactive – переменная-активатор алгоритма;

– int algospeed – переменная, регулирующая скорость выполнения алгоритма;

– int AlgoSpeed – свойство, устанавливающее и выдающее значение поля algospeed.

В работе также использовались структуры nodeinfo и edgeinfo.

## 2.5 Структура «nodeinfo»

Структура представляет собой запись данных о вершине графа.

Поля структуры:

- name: string – имя вершины;
- neighbors – List<edgeinfo> - список смежностей.

## 2.6 Структура «edgeinfo»

Структура представляет собой запись данных о дуге.

Поля структуры:

- weight: int – вес дуги;
- endname: string – имя конечной для дуги вершины;

На рисунке 1 представлена диаграмма классов.

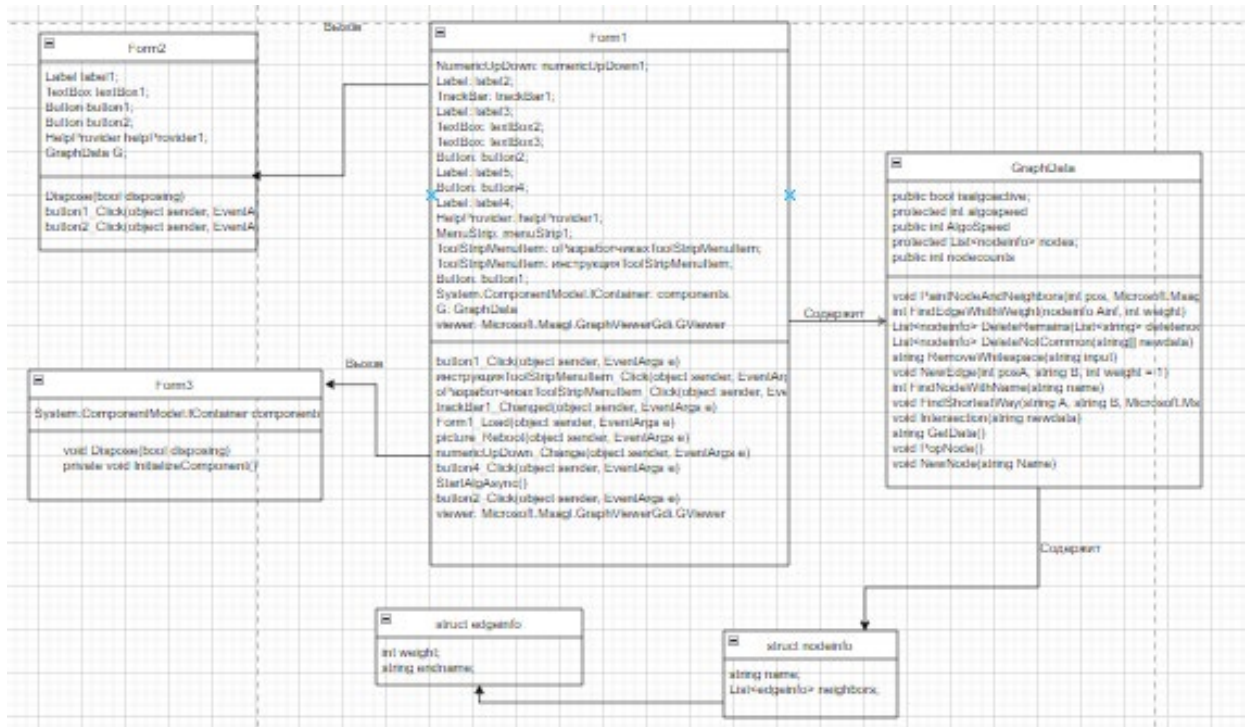


Рисунок 1 – Диаграмма классов

### 3 Описание работы программы

После запуска программы, перед пользователем открывается главное окно программы, как показано на рисунке 2. Интерфейс разделён на три части:

- 1) информационное меню,
- 2) меню построение графа,
- 3) элементы, ответственные за нахождение кратчайшего расстояния.

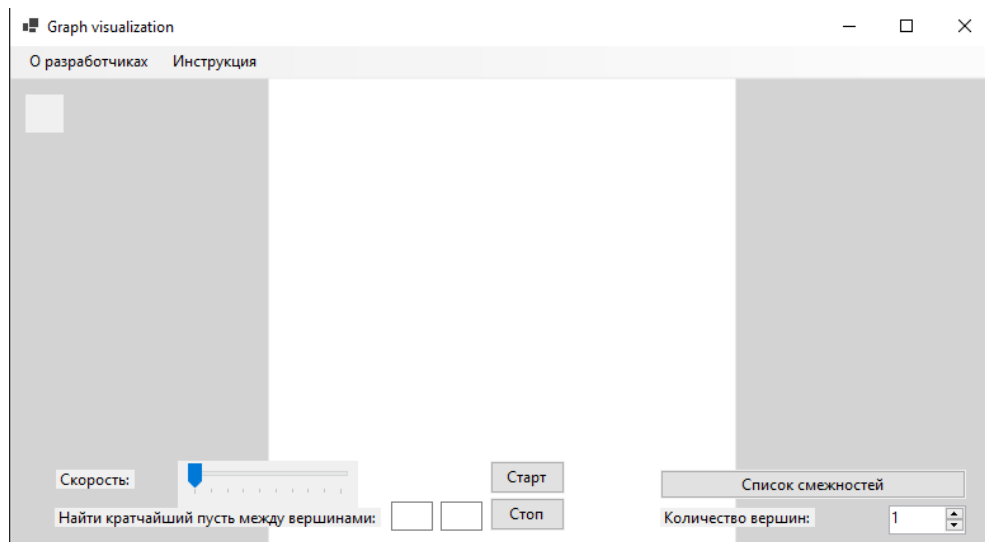


Рисунок 2 – Главное окно

Информационное меню содержит две вкладки: «О разработчиках» и «Инструкция». По нажатии на первую, открывается окно информации о разработчике, это показано на рисунке 3. По нажатии на вторую, открывается окно, информирующее пользователя о том, как пользоваться программой, это показано на рисунке 4.

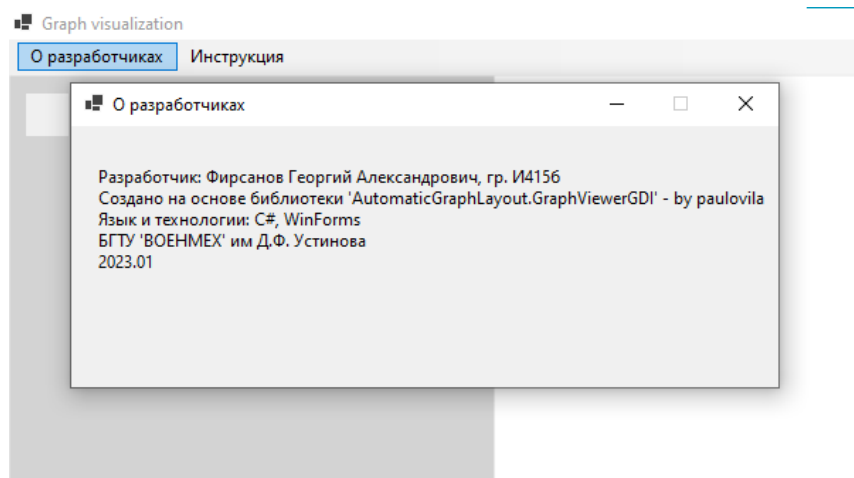


Рисунок 3 – Вкладка информации

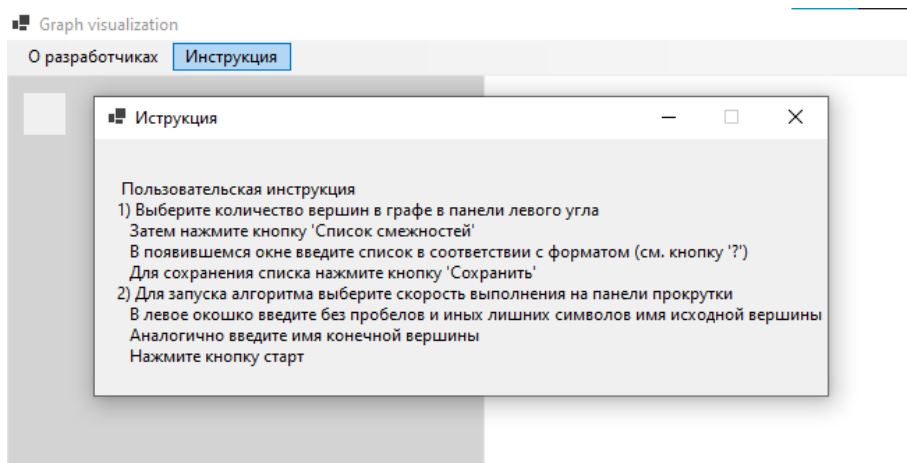


Рисунок 4 – Инструкция, расположенная в меню

Часть интерфейса, ответственное за построение графа, находится в правом нижнем углу окна программы. Пользователь может ввести количество вершин в графе в специальном окне напротив надписи «Количество вершин», может набрать нужное количество кнопками в виде стрелок. После нажатия на кнопку «Список смежностей» в окне редактирования открывается сгенерированные названия вершин, это показано на рисунке 5. В качестве названия может быть использовано любое слово, не содержащее символов «,» и «:». При неправильном вводе отображается ошибка.

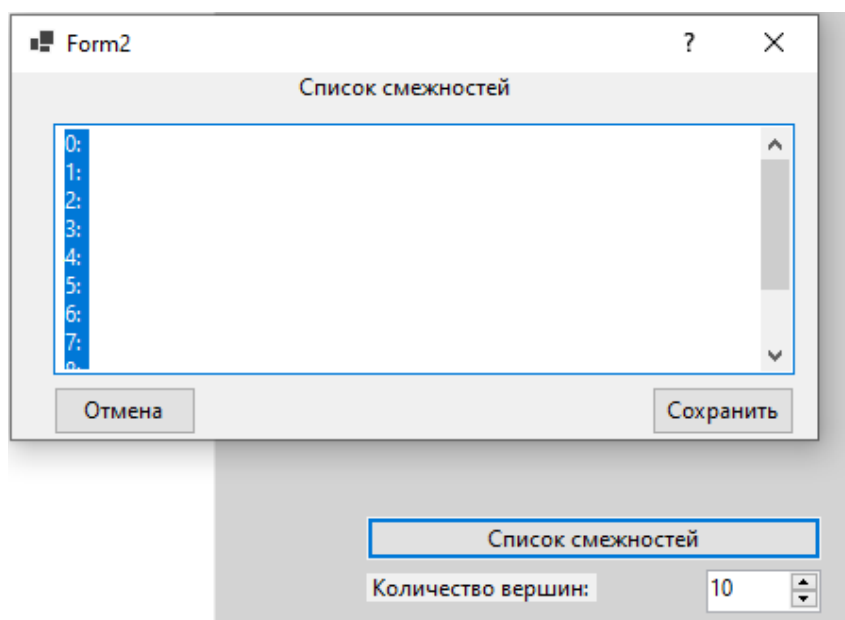


Рисунок 3 – Окно ввода списка смежностей

Пользователь может редактировать количество вершин непосредственно в редакторе списка смежностей, а может

уменьшать/увеличивать его в окне со стрелками, как на рисунке 6. Если ввести несуществующую вершину, она автоматически добавится в список после сохранения. Кнопка «Отмена» сбрасывает все изменения.

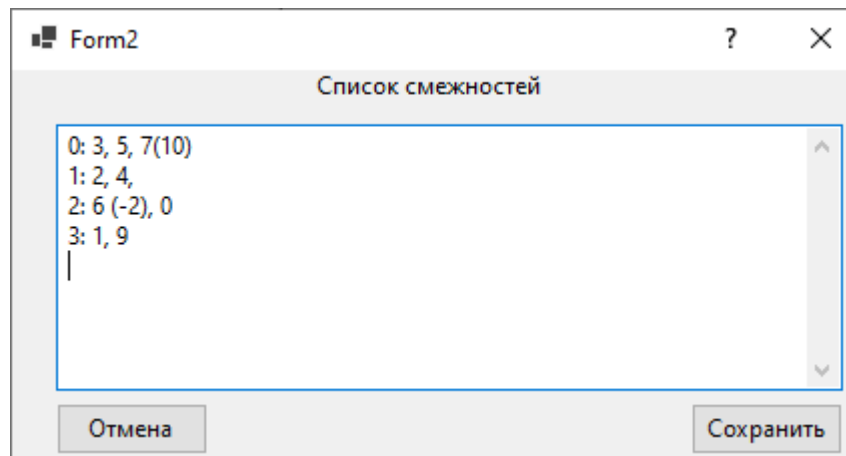


Рисунок 4 – Введённый список смежностей

После нажатия на кнопку «Сохранить», по середине окна появится граф, это отображено на рисунке 7. Вершины можно двигать, зажав левой кнопкой мыши.

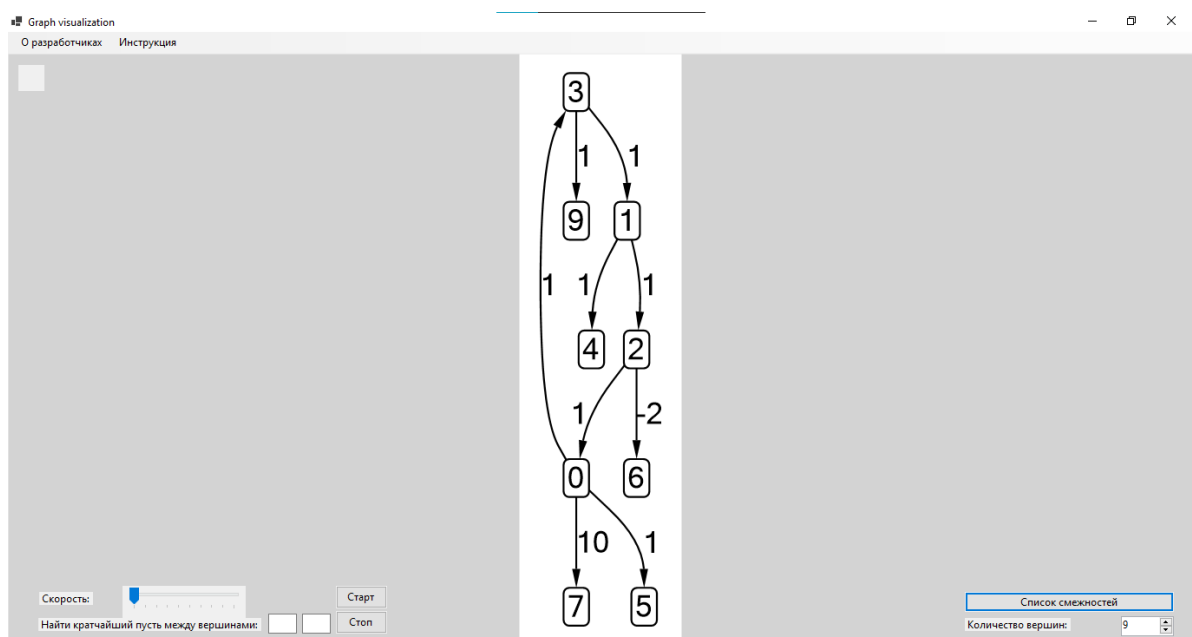


Рисунок 5 – Построенный граф

Третья часть интерфейса ответственна за настройку работы алгоритма поиска. Все клавиши находятся в левом нижнем углу окна программы и представляют собой три компонента:

- 1) бегунок скорости выполнения поиска,

- 2) окна ввода имён вершин,
- 3) кнопка запуска и кнопка остановки задачи.

Данные элементы управления видны как на рисунке 7, так и на рисунке 8.

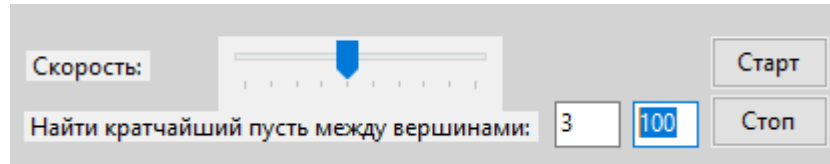


Рисунок 6 – Ввод параметров алгоритма

В случае попытки пользователем ввести несуществующую вершину в графе, программа выдаст сообщение о некорректности ввода, как показано на рисунке 9.

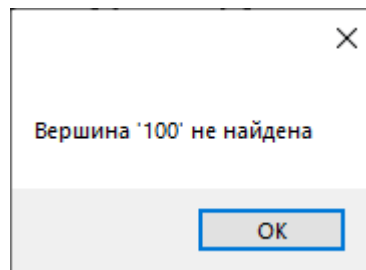


Рисунок 7 – Сообщение об ошибке

После нажатия кнопки «Старт», программа активирует поиск кратчайшего расстояния между вершинами графа. На рисунке 10 виден процесс работы, на рисунке 11 крупно показано сообщение о промежуточных действиях в вычислении расстояния.

Сам поиск сопровождается визуализацией процесса: при обходе всех вершин, активная окрашивается в тёмно-зелёный цвет. Смежные текущей вершины окрашиваются в светло-зелёный цвет, как на рисунке 12. Также выводится сообщение о текущем знании кратчайших расстояний до каждой из раскрашенных вершин.

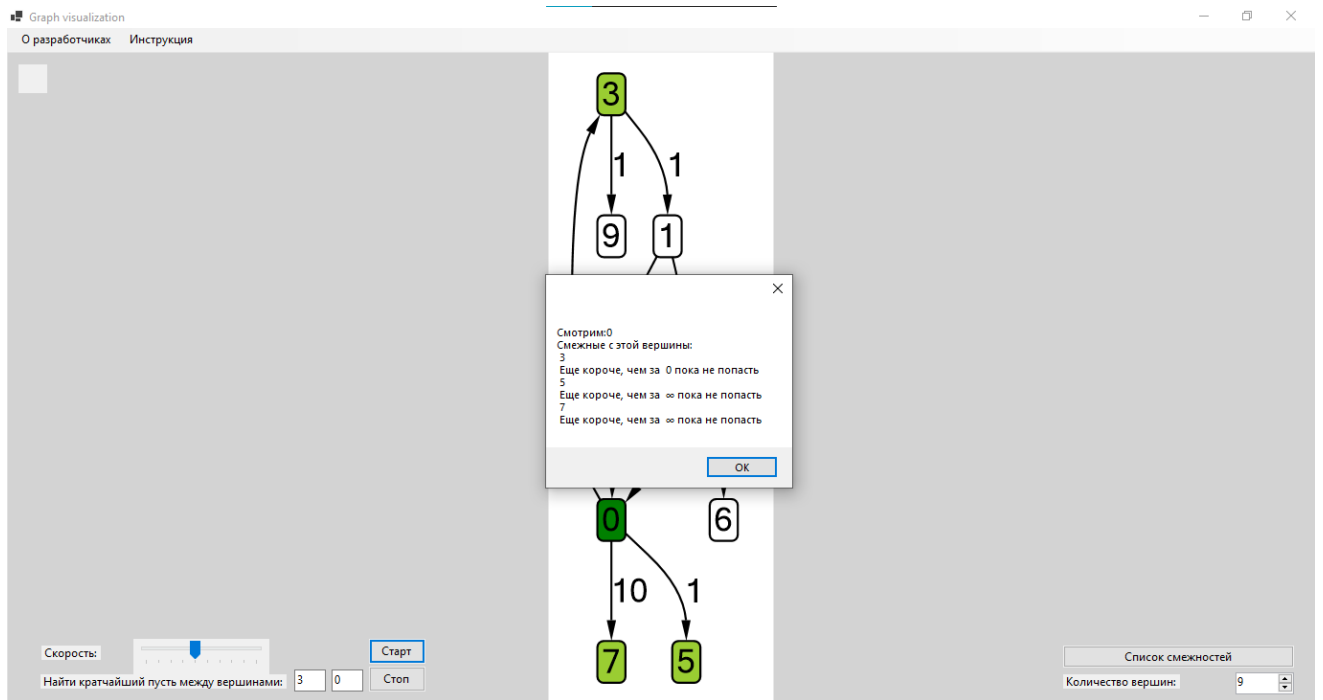


Рисунок 8 – Работа алгоритма поиска

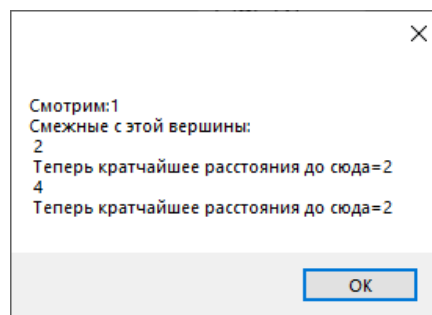


Рисунок 9 – Сообщение о промежуточных действиях алгоритм

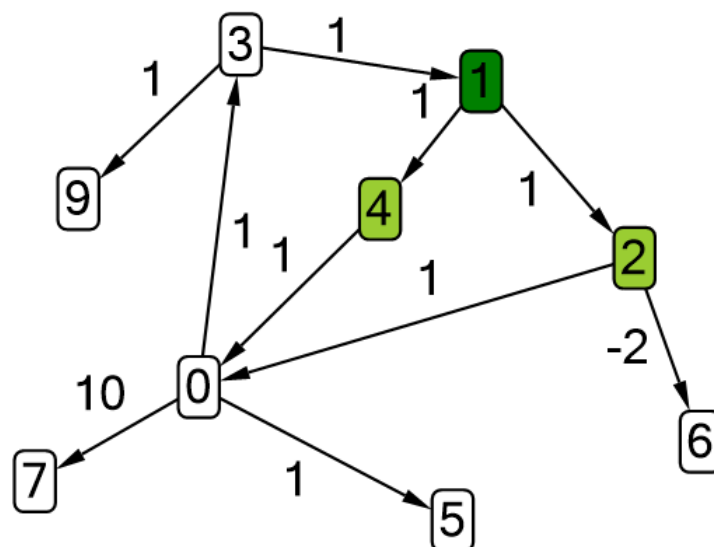


Рисунок 10 – Граф во время работы алгоритма

По результатам работы, программа выведет результат в левом верхнем углу окна сообщение, где описано, откуда, куда и сколько стоит кратчайший путь, видно на 13-м рисунке.

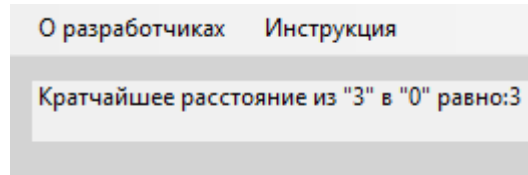


Рисунок 11 – Результат поиска

В основе поиска кратчайшего расстояния между вершинами лежит алгоритм Беллмана-Форда. Даже в случае, когда пользователь введет граф, в котором есть отрицательные дуги, алгоритм выведет результат. А если в графе есть циклы с отрицательным весом, как на рисунке 14, программа в конце работы выведет сообщение о том, что обнаружен такой цикл (рисунок 15).

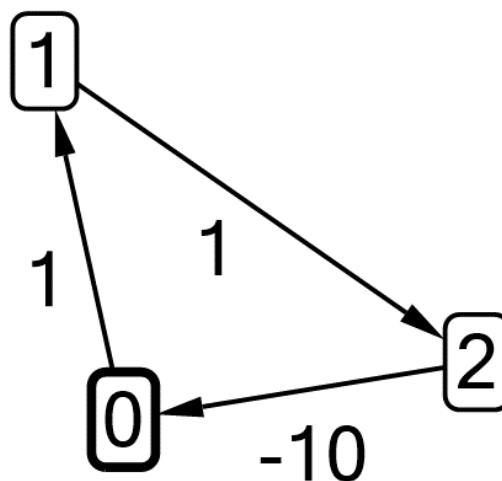


Рисунок 12 – Граф с отрицательным циклом

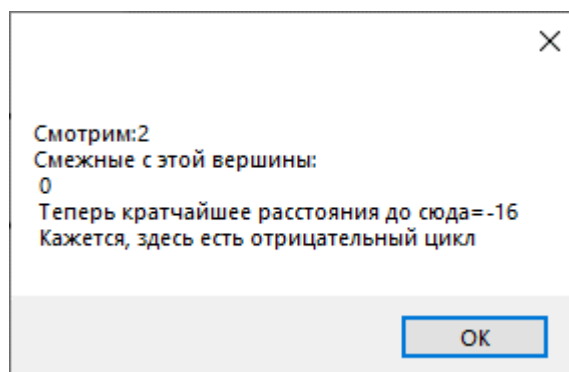


Рисунок 13 – Сообщение о нахождении отрицательного цикла



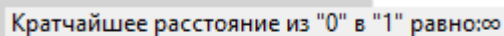
В таком случае, вывод сообщения о кратчайшем расстоянии будет вида, как на рисунке 16:



Кратчайшее расстояние из "0" в "2" равно:  $-\infty$

Рисунок 14 – Бесконечно короткое расстояние в графе

В обратном случае, когда нет никакого пути между двумя указанными вершинами, сообщение в конце будет таким:



Кратчайшее расстояние из "0" в "1" равно:  $\infty$

Рисунок 15 – Бесконечное расстояние между несвязанными вершинами

Рядом с кнопкой старта поиска находится кнопка «Стоп», при нажатии которой поиск прекращается.

## **ЗАКЛЮЧЕНИЕ**

В программе реализован объектно-ориентированный подход к написанию программы, реализован пользовательский интерфейс.

Была выполнена реализация визуализацию графа. Была составлена иерархия классов, а также проработка взаимодействия интерфейса и данных. Помимо просто визуализации, программа способна обрабатывать данные: можно не только получать значения, а видеть, как они были получены, при чем с разной скоростью, которую пользователь способен регулировать сам под свои нужды. Пользователь может вводить список смежностей. Программа соответствует всем поставленным целям и задачам и работает исправно.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Руководство по программированию в Windows forms. – URL: <https://metanit.com/sharp/windowsforms/> (дата обращения: 13.01.2023).
2. Официальная документация Microsoft Введение в NuGet. – URL: <https://learn.microsoft.com/ru-ru/nuget/what-is-nuget> (дата обращения: 14.01.2023).
3. Windows forms Руководство по классическим приложениям (Windows forms, .NET). – URL: <https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-6.0> (дата обращения: 9.01.2023).
4. Документация по библиотеке MSAGL. – URL: <https://github.com/Microsoft/automatic-graph-layout> (дата обращения: 14.01.2023).
5. Асинхронное программирование. – URL: <https://metanit.com/sharp/tutorial/13.3.php> (дата обращения: 16.01.2023).

## **ПРИЛОЖЕНИЕ А**

### **Текст программы**

Исходные тексты программы располагаются на прилагаемом электронном носителе.