

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения

Специальность программное обеспечение вычислительной техники и автоматизированных систем (230105)

Кафедра автоматики и компьютерных систем

ДИПЛОМНЫЙ ПРОЕКТ

Тема работы
Разработка информационной системы «Студграф»

УДК 004.415:51-022.334

Студент

Группа	ФИО	Подпись	Дата
3-8001	Горобченко Сергей Николаевич		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Паньшин Г.Л.			

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Конотопский В.Ю.	К.Э.Н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Невский Е.С.			

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АиКС	Фадеев А.С.	К.Т.Н.		

Министерство образования и науки Российской Федерации
 федеральное государственное автономное образовательное учреждение
 высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
 ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения
 Направление подготовки (специальность) – 230105 «Программное обеспечение
 вычислительной техники и автоматизированных систем»
 Кафедра автоматики и компьютерных систем

УТВЕРЖДАЮ:
 Зав. кафедрой АиКС Фадеев А.С.

 (Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

дипломной работы

Студенту:

Группа	ФИО
3-8001	Горобченко Сергей Николаевич

Тема работы:

Разработка информационной системы «Студграф»	
Утверждена приказом директора (дата, номер)	№ 2917/с от 15.04.2016 г.

Срок сдачи студентом выполненной работы:	01.06.2016 г.
--	---------------

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе	Проектировка и реализация программы, позволяющей задавать неориентированные графы визуальным способом и выполнять следующие алгоритмические задачи над графами: поиск кратчайшего пути между двумя выбранными вершинами; решение задачи коммивояжера любым алгоритмом
Перечень подлежащих исследованию, проектированию и разработке вопросов	Изучение методов представления графов, изучение алгоритмов поиска кратчайших путей в графе, изучение вопросов визуализации с применением эффекта сглаживания anti-aliasing.
Перечень графического материала	– графическое представление графа – диаграмма классов программы
Консультанты по разделам выпускной квалификационной работы	
Раздел	Консультант

Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Конотопский В.Ю.
Социальная ответственность	Невский Е.С.
Названия разделов, которые должны быть написаны на русском и иностранном языках:	
Обзор литературы, заключение	

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	01.02.2015 г.
---	---------------

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Паньшин Г.Л.			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8001	Горобченко Сергей Николаевич		

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения

Направление подготовки (специальность) – 230105 «Программное обеспечение вычислительной техники и автоматизированных систем»

Кафедра автоматике и компьютерных систем

Период выполнения – осенний/весенний семестр 2015/2016 учебного года

Форма представления работы:

дипломная работа

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	01.06.2016 г.
--	---------------

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
1.06.2016 г.	Основная часть	70
24.05.2016 г.	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	15
28.05.2016 г.	Социальная ответственность	15

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Паньшин Г.Л.			

СОГЛАСОВАНО:

Зав. Кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АиКС	Фадеев А.С.			

**ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСООБЪЕКТИВНОСТЬ И
РЕСУРСОСБЕРЕЖЕНИЕ»**

Студенту:

Группа	ФИО
3-8001	Горобченко Сергей Николаевич

Институт	Электронного обучения	Кафедра	АиКС
Уровень образования	Специалитет	Направление/специальность	230105

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:

1. Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих	
2. Нормы и нормативы расходования ресурсов	
3. Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования	

Перечень вопросов, подлежащих исследованию, проектированию и разработке:

1. Оценка коммерческого потенциала инженерных решений (ИР)	
2. Формирование плана и графика разработки и внедрения ИР	
3. Обоснование необходимых инвестиций для разработки и внедрения ИР	
4. Составление бюджета инженерного проекта (ИП)	
5. Оценка ресурсной, финансовой, социальной, бюджетной эффективности ИР и потенциальных рисков	

Перечень графического материала (с точным указанием обязательных чертежей)

1. ФСА диаграмма
2. Оценка перспективности нового продукта
3. График разработки и внедрения ИР
4. Инвестиционный план. Бюджет ИП
5. Основные показатели эффективности ИП
6. Риски ИП

Дата выдачи задания для раздела по линейному графику	
---	--

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Конотопский В.Ю.	к. э. н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8001	Горобченко Сергей Николаевич		

**ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»**

Студенту:

Группа	ФИО
3-8001	Горобченко Сергей Николаевич

Институт	Электронного обучения	Кафедра	
Уровень образования	Специалитет	Направление/специальность	230105

Исходные данные к разделу «Социальная ответственность»:	
1. Социальная ответственность	Предназначение раздела социальная ответственность
2. Анализ возможных сбоев и их последствий	Описание вероятных сбоев и их возможных последствий. Сбой в процессе визуализации алгоритмов. Сбой в связи с возможной утечкой памяти, так как в программе имеются неуправляемые ресурсы, обёрнутые в управляемую оболочку. Сбой в связи с подмены системных библиотек сторонним программным обеспечением, в том числе вирусами. Вероятные последствия сбоя в программе.
3. Встроенная система защиты от сбоев	Нужно предусмотреть максимальное количество факторов, влияющих на возникновение сбоев и постараться защитить систему от их влияния. Отключения доступности элементов управления, когда работа программы при их доступности имеет неопределенный характер. Включение условных операторов в программный код с целью проверки неожиданных случаев. Валидация данных для отсеечения ввода некорректных данных.
4. Мероприятия по защите здоровья и безопасности пользователя	Защита здоровья и безопасности пользователя включает предоставление продуктов и услуг, являющихся безопасными и не представляющих неприемлемого риска причинения ущерба при использовании или потреблении. Защита от электромагнитных полей ПЭВМ.

	Меры предосторожности при работе с ПЭВМ.
5. Мероприятия по защите оборудования	<p>Поскольку разрабатываемый проект представляет собой программный комплекс, то воздействие факторов, влияющих на физическую платформу, на базе которой будет разворачиваться система, невозможно предотвратить средствами самой системы. Для защиты необходимо обеспечить платформу физическими средствами, которые позволят игнорировать данные факторы.</p> <p>Для надежной защиты компьютера от сбоев с электропитанием рекомендуется использовать специальный прибор – источник бесперебойного питания.</p> <p>Для блокирования (парирования) случайных угроз безопасности информации в компьютерных системах рекомендуется производить дублирование информации на жесткий магнитный диск.</p>
Перечень графического материала:	
Графические изображения	Изображение защиты кабельной линии компьютера.

Дата выдачи задания для раздела по линейному графику	23.05.16
--	----------

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Невский Е.С.			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8001	Горобченко Сергей Николаевич		

РЕФЕРАТ

Выпускная квалификационная работа 76 стр., 13 рис., 12 табл., 16 источников, 2 приложения.

Ключевые слова: ТЕОРИЯ ГРАФОВ, ВИЗУАЛИЗАЦИЯ АЛГОРИТМА ЛИ, ВИЗУАЛИЗАЦИЯ ЗАДАЧИ КОММИВОЯЖЁРА, С#.

Объектом исследования является ввод графа интуитивно-понятным способом, визуализация решения задачи поиска кратчайшего пути в графе, а также визуализация решения задачи коммивояжёра.

Целью данной работы является разработка программы, позволяющей задавать неориентированные графы визуальным способом и выполнять следующие алгоритмические задачи над графами:

1. Поиск кратчайшего пути между двумя выбранными вершинами;
2. Решение задачи коммивояжёра любым алгоритмом.

В процессе исследования проводились декомпозиция задачи на подзадачи, сбор и анализ информации по выбранной тематике с последующим синтезом при проектировании, прототипирование (макетирование), тестирование и отладка.

В результате исследования была достигнута поставленная цель – создан программный продукт, позволяющий задавать неориентированные графы и выполнять визуализацию алгоритмических задач над графами.

Основные конструктивные, технологические и технико-эксплуатационные характеристики, такие как высокое быстродействие, нетребовательность к ресурсам памяти и центрального процессора, соответствуют предъявленным требованиям.

Степень внедрения: продукт является завершённым.

Область применения: в учебных целях.

Экономическая эффективность работы обоснована с точки зрения ресурсосбережения.

В будущем планируется дополнить программу визуализацией новых алгоритмических задач над графами.

Определения, обозначения, сокращения, нормативные ссылки

В данной работе применены следующие термины с соответствующими определениями.

Информационная система: Программное средство, предназначенное для удовлетворения информационных потребностей в рамках определенной предметной области.

граф: Совокупность непустого множества вершин и наборов пар вершин (связей между вершинами).

алгоритм волновой трассировки (алгоритм Ли, волновой алгоритм): Алгоритм поиска пути, алгоритм поиска кратчайшего пути на планарном графе. Принадлежит к алгоритмам, основанным на методах поиска в ширину.

вершина: Фундаментальная единица, образующая графы — неориентированный граф состоит из множества вершин и множества рёбер (неупорядоченных пар вершин), в то время как ориентированный граф состоит из множества вершин и множества дуг (упорядоченных пар вершин).

задача коммивояжёра: Задача, заключающаяся в отыскании самого выгодного маршрута, проходящего через указанные города хотя бы по одному разу с последующим возвратом в исходный город.

Оглавление

Введение.....	12
1 Обзор литературы	14
2 Literature review	15
3 Объект и методы исследования	16
3.1 Обзор методов представления графов	18
4 Реализация информационной системы	22
4.1 Выбор инструментария.....	22
4.2 Интерфейс пользователя	22
4.3 Описание программной реализации	28
5 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение.....	40
5.1 Организация и планирование работ	40
5.2 Расчет сметы затрат на выполнение проекта	44
5.3 Оценка экономической эффективности проекта	49
6 Социальная ответственность	52
6.1 Анализ возможных сбоев и их последствий	52
6.2 Методы и технологии встроенной защиты от ошибок	55
6.3 Мероприятия по защите здоровья и безопасности пользователя	56
6.4 Мероприятия по защите оборудования	59
Заключение	62
Conclusion.....	63
Список использованных источников	64
Приложение А	66
Приложение Б	68

Введение

Целью данной работы является разработка программы, позволяющей задавать неориентированные графы визуальным способом и выполнять следующие алгоритмические задачи над графами:

1. Поиск кратчайшего пути между двумя выбранными вершинами;
2. Решение задачи коммивояжёра любым алгоритмом.

Основные задачи:

- Спроектировать систему программных классов;
- Реализовать алгоритм Ли;
- Реализовать решение задачи коммивояжёра методом ветвей и границ;
- Спроектировать систему программных классов.

В настоящее время методы визуализации графов имеют множество применений. Графы и графовые модели используются для задач, оперирующих информацией, представленной в виде объектов и связей между ними. Например, синтаксические деревья в трансляторах, раскраска графов при конструировании электрических схем, поиск кратчайшего пути в компьютерных играх или в геоинформационных системах, так же визуализацию графов можно применять в учебных целях для наглядного изучения теории графов. Системы, где используют графическое представление, редко бывают статичными. Другими словами, в таких системах протекают некоторые процессы, которые могут изменить связи или иным способом перестраивать структуру модели. К статичным системам можно отнести большинство диаграмм, например, диаграммы, используемые в рамках визуального моделирования, и блок схемы. Для представления графа в динамической форме необходима генерация изображений для представления промежуточных состояний графовых моделей.

Несомненно, главным применением является применение в области улучшения качества преподавания обучающихся в старших классах, а также студентов.

В настоящее время преподавание осуществляется, в основном, без возможности визуализации преподаваемых прикладных механизмов, что затрудняет и замедляет обучение учащихся. Необходимость во внедрении соответствующих программных систем очевидна.

На данном этапе разработки система содержит лишь визуализацию некоторых алгоритмов, применяемых к графам. Такие функциональные рамки связаны с временными ресурсами, имеющимися в распоряжении разработчика. В дальнейшем предполагается значительно увеличить число поддерживаемых программным обеспечением функций, разработав алгоритмы визуализации других часто встречающихся задач, необязательно относящихся только к дисциплине «Теории графов», но и относящихся к таким дисциплинам как «Дискретная математика», «Структура и алгоритмы обработки данных», и других, что существенно повысит успеваемость студентов и сделает процесс обучения более комфортным.

Некоторые другие возможные прикладные применения разработанной программной системы кратко перечислены ниже:

1. Возможность мгновенного воспроизведения ранее забытых алгоритмов в памяти пользователя посредством просмотра визуализальной трансляции решения типовых задач;

2. Закрепление полученных знаний;

3. Непосредственно решение типовых задач с наглядным воспроизведением результата;

4. Прокладывание маршрута от пункта отправления до пункта назначения по кратчайшему пути (алгоритм Ли);

5. Поиск наиболее выгодного маршрута для посредника, развозящего товары конечным потребителям (задача коммивояжера).

Перечисленные выше преимущества делают данную разработку важным и полезным инструментом.

1 Обзор литературы

При разработке данной системы была необходимость в получении знаний о предметной области (графы; алгоритм Ли; задача коммивояжёра) и знаний, связанных с разработкой и реализацией некоторых частей программного комплекса.

При написании данной работы были использованы научная и учебно-методическая литература, а также нормативно-законодательные акты Российской Федерации.

Основными источниками, раскрывающими теоретические основы выбранной предметной области, явились работы Домнина Н.М., Касьянова В.В., а также такие веб-ресурсы, как «Википедия», «Интуит», некоторые статьи, размещенные на сайте Механико-математического факультета МГУ имени М.В. Ломоносова, пособие (состоящее из двух частей) по дисциплине «Теория графов» преподавателя Томского Политехнического Университета Буркатовской Ю.Б., где кратко и ясно излагается теоретический материал, необходимый для выполнения выпускной квалификационной работы.

Основными источниками, необходимыми для усиления навыков в программировании с целью реализации данной разработки, явились работы Шилдта Г., Уотсона К., Петцольда Ч., Жаркова В., а также такие веб-ресурсы, как «Профессорвеб», «Интуит».

2 Literature review

To develop the program system “Studgraph” we were needed to obtain knowledge of the subject area (graphs, the algorithm Lee, traveling salesman problem) and knowledge related to the designing and implementation of some parts of the software.

In order to achieve the goal we were used scientific and educational-methodical literature, and also legislative and regulatory acts of the Russian Federation.

The main sources that reveal the theoretical foundations of the chosen subject area, were the work of Domnin NM, Kasyanov VV, and such web resources as a “Wikipedia”, “Intuit”, some articles posted on the website of the Faculty of Mechanics and Mathematics of Lomonosov Moscow State University, methodical manual (that consists two parts) for discipline “Graph theory” written by the teacher Yuliya Burkatovskaya of the Tomsk Polytechnic University, where the theoretical material presented briefly and clearly and is required to perform the final qualifying work.

The main sources necessary for doing programming skills better in order to implement this development were the work of Shildt G., Watson K., Charles Petzold, Vladimir Zharkov, and web resources such as “Professorweb” and “Intuit”.

3 Объект и методы исследования

Техническое задание на программу «Студграф», полученное в период прохождения преддипломной практики, приведено в приложении А расчетно-пояснительной записки.

Разрабатываемый продукт выполнен на основе идеи и функциональных возможностей программы «Студграф» версии, созданной при прохождении преддипломной практики, с учётом недостатков архитектурно-значимых решений, принятых при разработке последней, а также с учётом необходимости в дополнении исходной программы реализацией решения задачи коммивояжёра методом ветвей и границ.

В программе «Студграф» исходной версии были получены следующие результаты:

- Реализован алгоритм Ли с подробным визуальным решением задачи поиска кратчайшего пути;
- Реализован ввод неориентированного ненагруженного графа визуальным способом;
- Возможность задания входных данных для алгоритма Ли в графическом интерфейсе пользователя;
- Автоматическое назначение имен вершинам графа;
- Разработан дизайн современного типа.

Исходная версия программы представлена на рисунке 1.

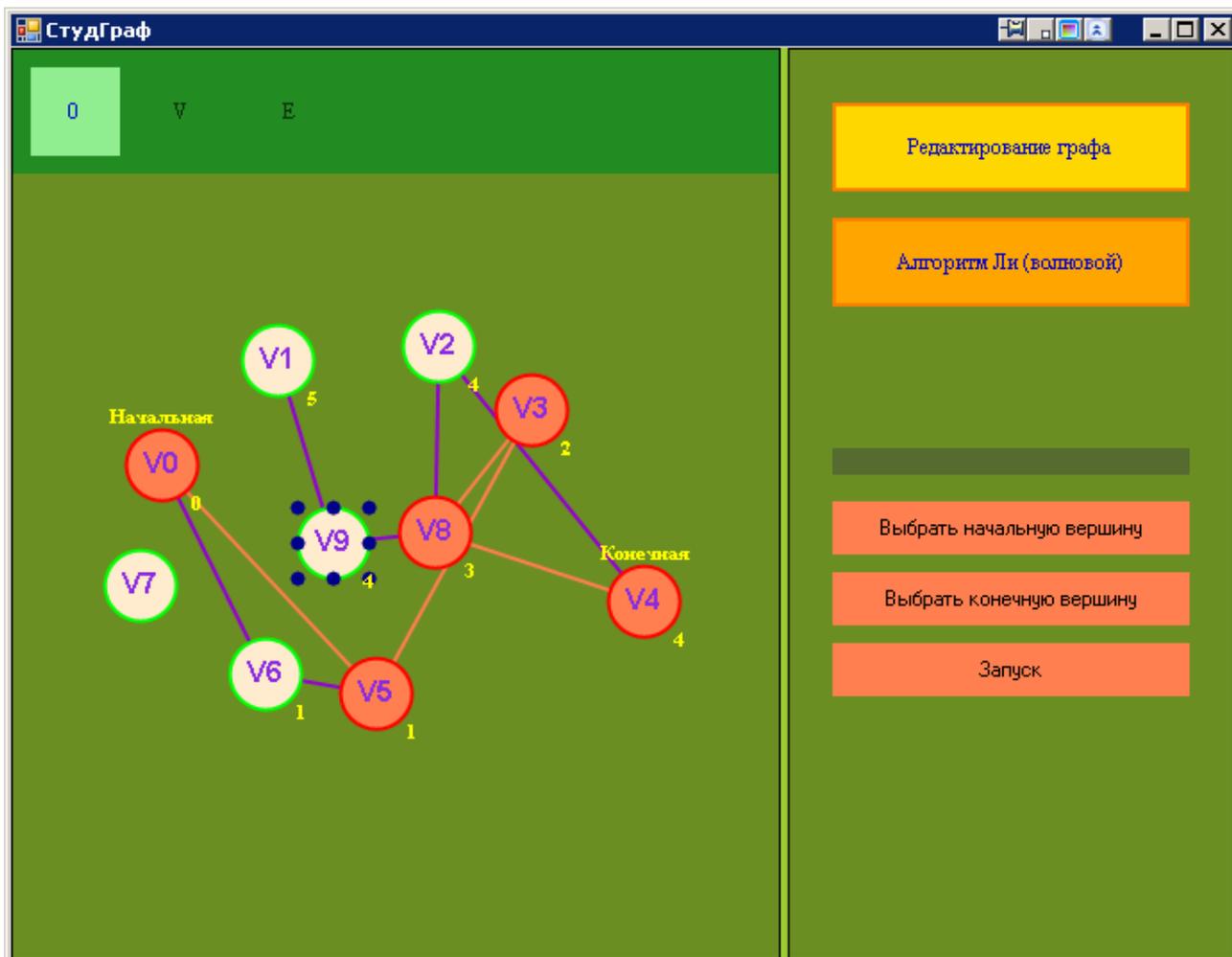


Рисунок 1 – Программа «Студграф» исходная версия

Проанализировав уже имеющуюся программу, были выявлены следующие её недостатки (которые были учтены при разработке нового программного продукта):

- Редкие сбои при выполнении программы;
- Не продумана возможность дополнения новыми алгоритмами;
- Наличие только одной задачи, которую программа способна решать и визуализировать её решение.

Объектом исследования настоящей разработки является ввод графа интуитивно-понятным способом, визуализация решения задачи поиска кратчайшего пути в графе, а также визуализация решения задачи коммивояжёра.

Необходимо разработать программный продукт, позволяющий задавать графы, решать задачу коммивояжёра, находить кратчайший путь в графе с визуализацией решения данных задач.

При этом разработанный программный продукт учитывает все недостатки, выявленные в процессе анализа программы «Студграф» исходной версии. При решении поставленной задачи были применены следующие методы:

- Декомпозиция системы с объектно-ориентированным основанием и направлением «сверху вниз»;
- Прототипирование интерфейса;
- Принцип безопасного программирования – способ написания кода, снижающий вероятность внесения некоторых типовых ошибок;
- Тестирование и отладка;
- Анализ исследуемой предметной области с последующим синтезом в программный продукт.

Такие методы позволяют разработчику создавать крупные программные продукты и дают возможность сопровождения данного продукта.

3.1 Обзор методов представления графов

Знакомство со способами представления и обработки графов весьма поучительно. С одной стороны, графы являются достаточно наглядными объектами. С другой стороны, машинное представление графов допускает большое разнообразие. Сложность получения ответа на тот или иной вопрос относительно данного графа зависит, естественно, от способа представления графа.

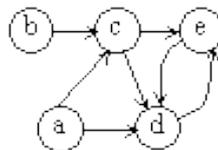
3.1.1 Матричный способ

Очень распространенным методом представления графов является матричный способ. Для представления ненаправленных графов обычно используют матрицы смежности, а для ориентированных графов матрицы инцидентности. Обе матрицы имеют размерность $n * n$, где n - число вершин в графе. Вершины графа последовательно нумеруются.

Матрица смежности имеет значение ноль в позиции $m(i,j)$, если не существует ребра, связывающего вершину i с вершиной j , или имеет единичное значение в позиции $m(i,j)$, если такое ребро существует.

Правила построения матрицы инцидентности аналогичны правилам построения матрицы смежности. Разница состоит в том, что единица в позиции $m(i,j)$ означает выход дуги из вершины i и вход дуги в вершину j .

На рисунке 2 изображено матричное представление графа.



Матрица смежности

	a	b	c	d	e
a	0	0	1	1	0
b	0	0	1	0	0
c	0	1	0	1	1
d	1	0	1	0	1
e	1	0	1	1	0

Матрица инцидентности

	a	b	c	d	e
a	0	0	1	1	0
b	0	0	1	0	0
c	0	0	0	1	1
d	0	0	0	0	1
e	0	0	0	1	0

Рисунок 2 – Матричное представление графа

3.1.2 Списанием вершин и списком ребер

Граф можно представить в виде списочной структуры, состоящей из списков двух типов списка вершин и списков ребер. Элемент списка вершин содержит поле данных и два указателя. Один указатель связывает данный элемент с элементом другой вершины. Другой указатель связывает элемент списка вершин со списком ребер, связанных с данной вершиной. Для ориентированного графа используют список дуг, исходящих из вершины. Элемент списка дуг состоит только из двух указателей. Первый указатель используется для того, чтобы показать в какую вершину дуга входит, а второй — для связи элементов в списке дуг вершины.

На рисунке 3 изображено представление графа в виде списочной структуры.

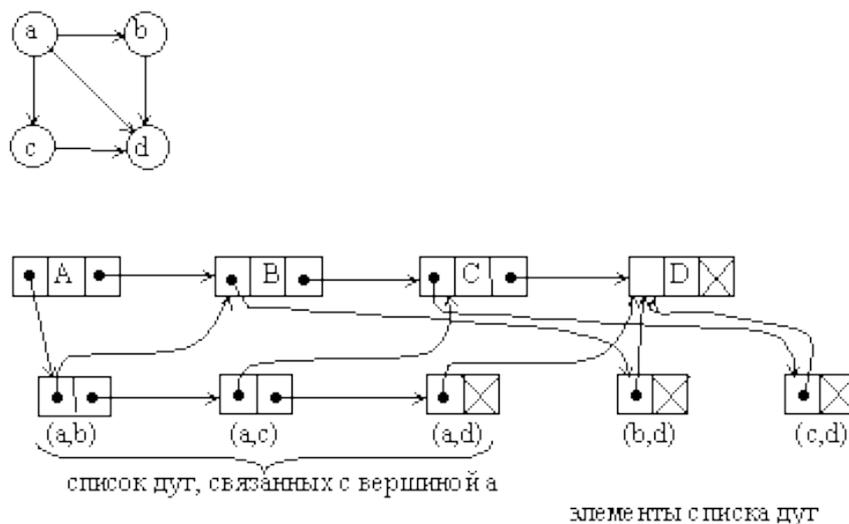


Рисунок 3 – Представление графа в виде списочной структуры

3.1.3 Графический метод

Представление графа в виде перечисления вершин и ребер или в табличной форме удобно для машинной обработки. Человеку же удобно работать с графом-рисунком, так как для пользователя такое представление является интуитивно

понятным, а также упрощает поиск связей между вершинами в наглядном виде с помощью ребер, изображаемые непрерывными линиями.

На рисунке 4 представлено графическое изображение графа.

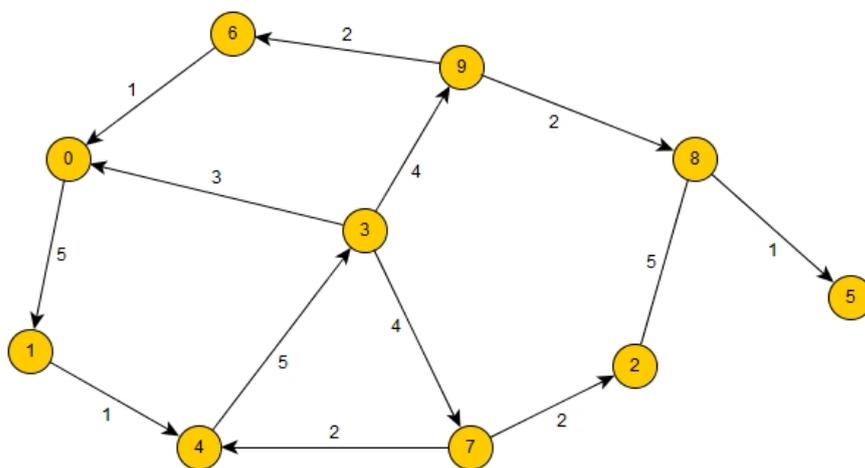


Рисунок 4 – Графическое представление графа

4 Реализация информационной системы

4.1 Выбор инструментария

Наиболее широко распространёнными языками программирования (при разработке десктопных приложений) являются: c++, c#, Object Pascal / Delphi, Компонентный Паскаль, Java.

При разработке данного приложения необходима была надёжность и устойчивость разработки, а значит нужен безопасный объектно-ориентированный язык с функцией автоматической сборки мусора для автоматического освобождения памяти, с функцией обработки исключительных ситуаций, а также недопущением считывания неинициализированных переменных и выхода индекса массива за пределы допустимого диапазона или выполнения непроверенных приведений типов. Такому требованию удовлетворяют два языка из вышеприведенного списка: c# и Java.

Каждый из этих языков программирования имеет свои значительные преимущества. Так как программный продукт создавался для работы под управлением операционной системы семейства Windows, кросс-платформенность, предоставляемая языком Java, не представляет интереса для разработчика. Язык c# был выбран, так как выбор необходимо было выполнить.

В качестве среды разработки была выбрана среда разработки Microsoft Visual Studio 2010, которая является основным продуктом, внедряющим язык c#.

4.2 Интерфейс пользователя

При открытии программы на экране появляется форма, соответствующий класс которой называется MainForm. Её вид во время исполнения программы приведен на рисунке 5.



Рисунок 5 – Вид основной формы программы

Слева сверху изображена инструментальная панель программы, позволяющая задать любой неориентированный граф, не имеющий более чем одну ветвь между двумя выбранными вершинами. Данная инструментальная панель программы, содержит три кнопки при её открытии: О, V и E. С помощью инструмента “О” пользователь может перемещать (управлять) уже созданные вершины графа. С помощью инструмента “V” пользователь имеет возможность создавать новые вершины графа. Инструмент “E” предназначен для задания рёбер графа. Пример задания с помощью данной программы графа приведен на рисунке 6.

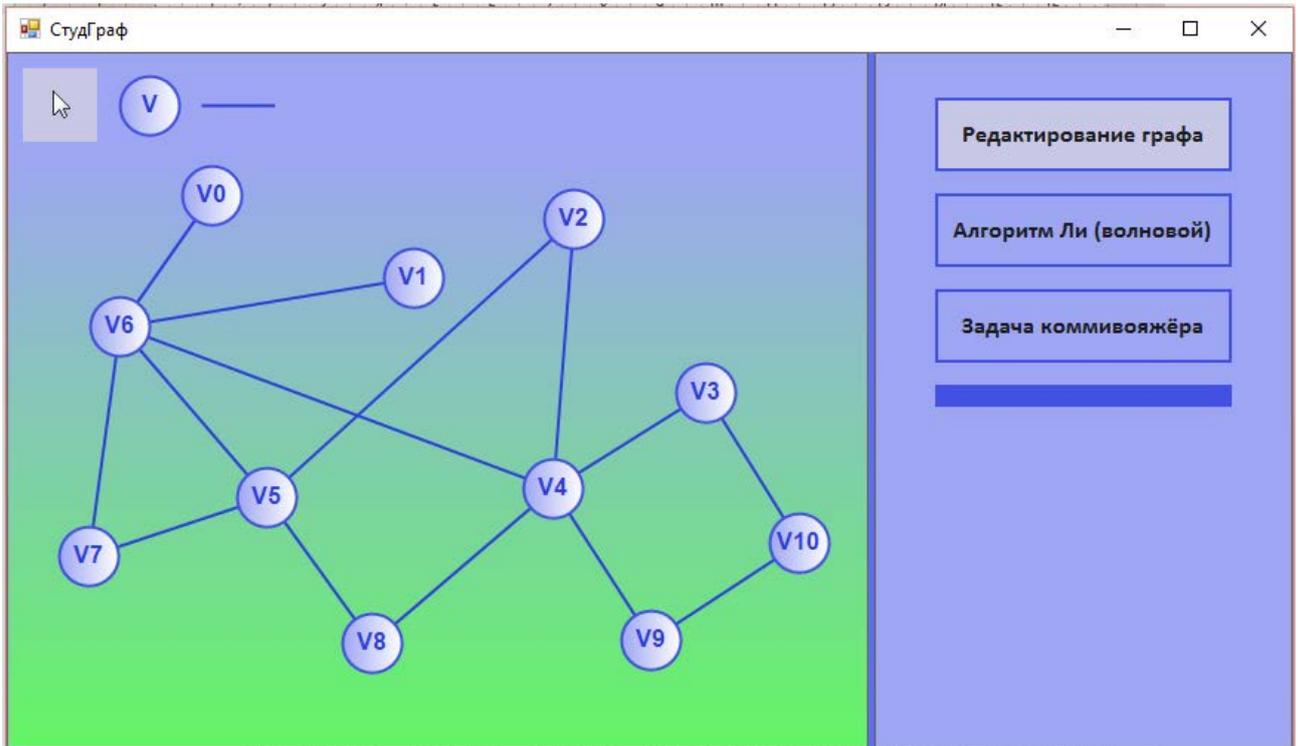


Рисунок 6 – Пример задания графа

Как видно, рабочая область, на которой изображается заданный пользователем граф, расположена слева снизу. Справа расположена дополнительная инструментальная панель, с помощью которой пользователь имеет доступ к реализации специфических функций, связанных с воплощением в жизнь уже реализованных алгоритмов и их визуализации. На рисунке 6 видно, что на данный момент выбран пункт “Редактирование графа” в дополнительной инструментальной панели. С помощью данного пункта (который открывается по умолчанию) пользователь имеет возможность задавать различные неориентированные графы в рабочей области.

Если необходимо посмотреть на работу алгоритма Ли, пользователю необходимо выбрать данный пункт на дополнительной инструментальной панели. После чего на данной панели появляются специфические для выбранного алгоритма дополнительные элементы интерфейса, которые можно увидеть на рисунке 7.

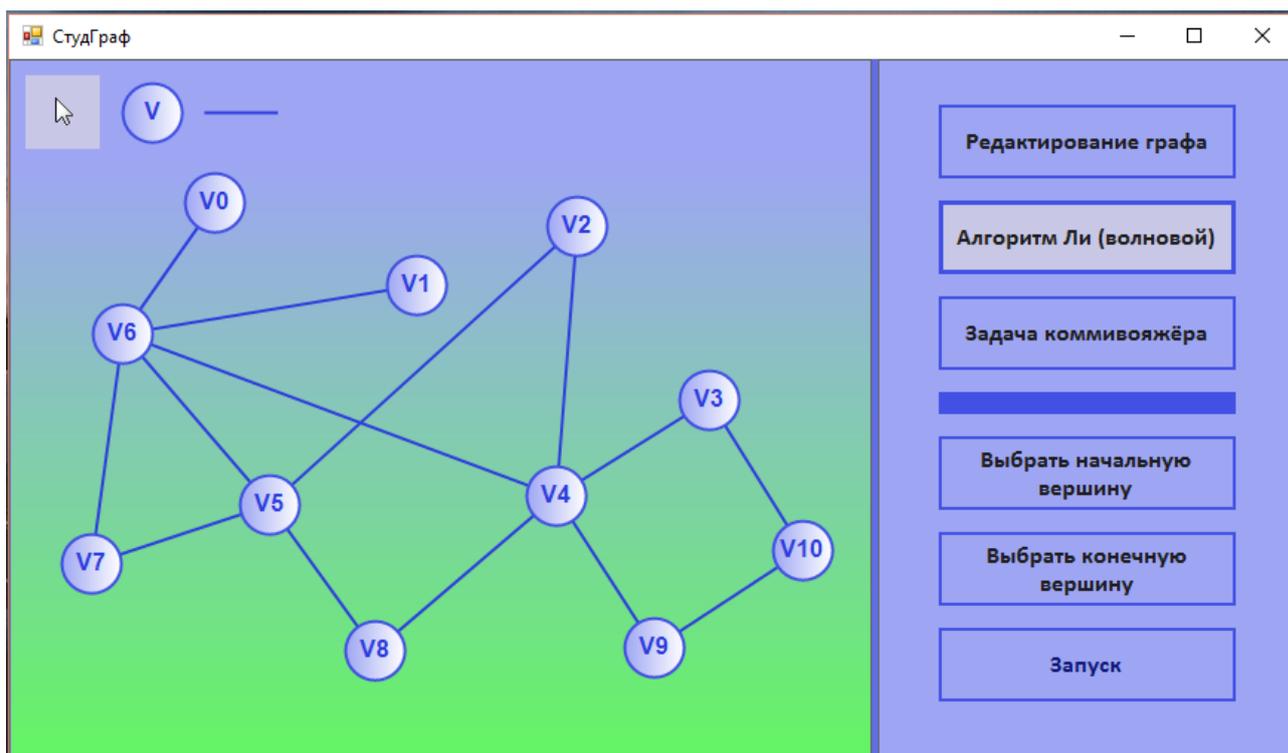


Рисунок 7 – Выбран «Алгоритм Ли» на дополнительной инструментальной панели

С помощью появившихся кнопок пользователь может указать, какая из вершин графа будет начальной для алгоритма, а какая конечной. При этом у пользователя имеется также возможность запустить алгоритм на исполнение спомощью нажатия по кнопке «Запуск». Визуализация алгоритма Ли представлена на рисунке 8.

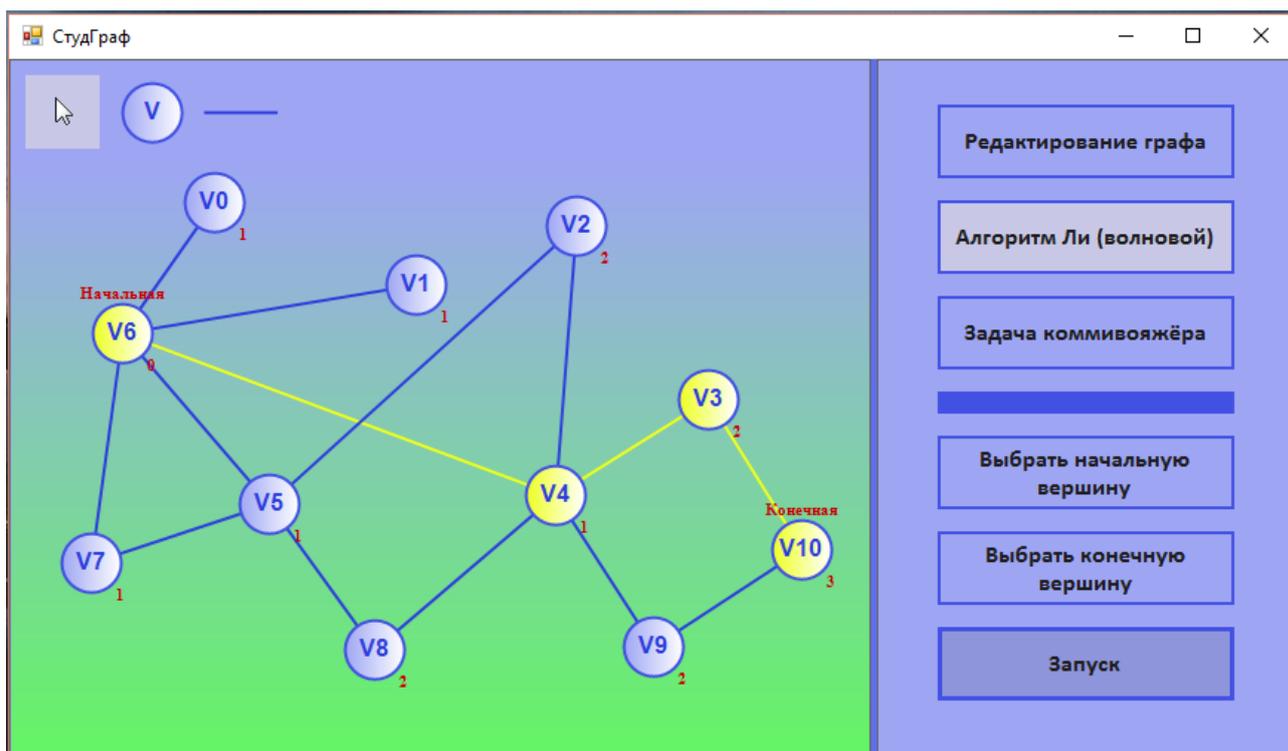


Рисунок 8 – Визуализация алгоритма Ли

Для того, чтобы решить с помощью разработанной программы задачу поиска оптимального маршрута в графе (задачу коммивояжёра), пользователю необходимо выбрать пункт «Задача коммивояжёра» на дополнительной инструментальной панели. При этом в рабочей области окна замещается панелью, заключающей в себе элементы управления, с помощью которых имеется возможность задания матрицы расстояний (как исходных данных к алгоритму решения поставленной задачи) и запуска решения с мгновенным отображением промежуточных результатов в виде видоизменяемой таблицы, что продемонстрировано на рисунке 9.



Рисунок 9 – Выбрана «Задача коммивояжёра» на дополнительной инструментальной панели

Пользователю предоставляется возможность ввести число городов и расстояние между ними. После применения введенного числа городов, создается шаблон таблицы, учитывающий введенное пользователем число городов, элементы которой могут быть заполнены пользователем расстояниями между соответствующими городами (при этом расстояния от города «А» в город «Б» и от города «Б» в город «А» могут быть различными, что может быть связано, например, с наличием односторонних дорог). После заполнения всех расстояний между городами для получения кратчайшего маршрута из города «1», проходящего через все остальные города, в город «1» (что является результатом работы алгоритма, реализующего метод ветвей и границ для решения поставленной задачи), достаточно кликнуть левой кнопкой мыши на кнопку «Получить кратчайший маршрут». На рисунке 10 изображена программа в момент представления результатов её работы, связанной с решением вышеупомянутой задачи.

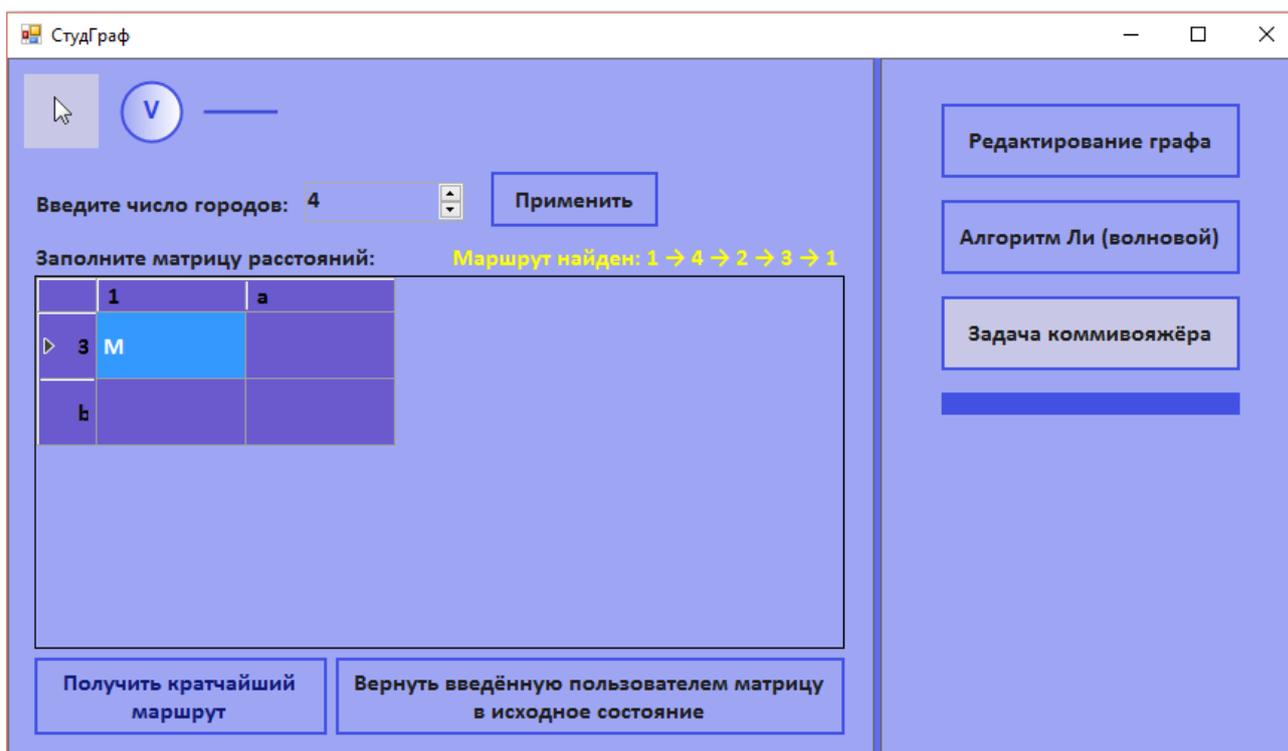


Рисунок 10 – Результат работы программы в режиме «Задача коммивояжёра»

4.3 Описание программной реализации

Исходники полученной системы - файлы, имена и содержимое которых указано в таблице 1.

Таблица 1 – Файлы с исходным кодом и их содержимое

Имя файла	Содержит
ActiveToolBoxButton.cs	Перечисление ActiveToolBoxButton
AlgLiEdge.cs	Класс AlgLiEdge
AlgLiGraph.cs	Класс AlgLiGraph
AlgLiVertex.cs	Класс AlgLiVertex
AlgorithmLi.cs	Класс AlgorithmLi
AlgorithmLiControl.cs	Класс AlgorithmLiControl, представляющий пользовательский элемент управления
Edge.cs	Класс Edge

Продолжение таблицы 1

EdgeVisualizer.cs	Класс EdgeVisualizer
Graph.cs	Класс Graph
GraphVisualizer.cs	Класс GraphiVisualizer
MainForm.cs	Класс MainForm, представляющий основное и единственное окно программы
Program.cs	Статический класс Program
SalesmanAlgorithmControl	Класс SalesmanAlgorithmControl, представляющий пользовательский элемент управления
SpecialFunction.cs	Перечисление SpecialFunction
Vertex.cs	Класс Vertex
VertexVisualizer.cs	Класс VertexVisualizer
SalesmanPanelControl.cs	Класс SalesmanPanelControl

Назначение классов и перечислений, используемых в разработанной программе представлено в таблице 2.

Таблица 2 – Назначение классов и перечислений

Элемент	Назначение
Перечисление ActiveToolBoxButton	Определяет одну из инструментальных кнопок
Класс AlgLiEdge	Наследуется от класса Edge. Ребро с возможностью его выделения, для отображения найденного пути алгоритмом Ли

Продолжение таблицы 2

Класса AlgLiGraph	Наследуется от класса Graph. Граф, который предназначен для работы внутри алгоритма Ли
Класс AlgLiVertex	Наследуется от класса Vertex. Вершина, которая предназначена для работы внутри алгоритма Ли
Класс AlgorithmLi	Реализация алгоритма Ли
Класс AlgorithmLiControl	Пользовательский элемент управления, который расположен справа снизу в главном окне программы и содержит элементы управления, используемые для реализации работы алгоритма Ли
Класс Edge	Ребро графа
Класс EdgeVisualizer	Визуализация (визуализатор) ребра графа
Класс Graph	Граф
Класс GraphVisualizer	Визуализация графа
Класс MainForm	Главное окно программы
Класс Program	Содержит точку входа в программу
Класс SalesmanAlgorithmControl	Пользовательский элемент управления, который расположен справа снизу в главном окне программы и содержит элементы управления, используемые для реализации задачи коммивояжёра (их нет)

Продолжение таблицы 2

Перечисление SpecialFunction	Определяет тип текущей специальной функции
Класс Vertex	Вершина графа с её позицией на 2хмерной плоскости
Класс VertexVisualizer	Визуализация вершины графа
Класс SalesmanPanelControl	Панель, визуализирующая решение задачи коммивояжёра методом ветвей и границ. Включает в себя код, реализующий алгоритм решения задачи коммивояжёра и его анимацию в данной панели

Класс Program создаёт основное окно программы и отображает его, запуская на исполнение очередь сообщений Windows с помощью метода Run класса Application. Реализация интерфейса пользователя для основного окна программы была выполнена в конструкторе среды разработки Visual Studio и содержит в себе элементы управления, необходимые для того, чтобы пользователь имел возможность давать команды программе и получать от неё выходную информацию в виде изображенного графа, визуализации хода решения задачи поиска кратчайшего пути алгоритмом Ли и хода решения задачи коммивояжёра методом ветвей и границ.

Каждый из элементов управления на главном окне имеет необходимые обработчики каких-либо событий. Например, при нажатии на кнопку “О” операционной системой вызывается метод (обработчик события по нажатию на данную кнопку) `onToolBoxButtonClick()`, который является общим обработчиком события по нажатию на одну из кнопок основной инструментальной панели, смысл которой был разъяснен ранее.

Данный метод выполняет поочередно следующие действия:

1. Вначале назначает всем инструментам из основной панели инструментов цвета неактивных кнопок.
2. Затем проверяет, если элемент, по которому кликнули, является экземпляром класса `Button`, то:
 - 2.1. Данный элемент становится подсвеченным (меняет свой цвет на цвет активной кнопки);
 - 2.2. Приватному полю `_activeToolBoxButton` класса `MainForm`, являющемуся объектом типа перечисления `ActiveToolBoxButton`, представляющего собой информацию о том, какая из кнопок инструментальной панели активна в данный момент, назначается значение в соответствии с `Tag`’ом элемента, по которому пользователь выполнил нажатие кнопкой мыши, который (`Tag`) из числового представления преобразуется в объект типа вышеуказанного перечисления;
 - 2.3. Далее задаётся видимость дополнительной кнопки, которая видима только в том случае, когда инструмент “Алгоритм Ли” выбран на дополнительной панели инструментов.

Основным содержанием тела данного метода является задание нового значения полю `_activeToolBoxButton`, которое используется при нажатии пользователем кнопкой мыши по рабочей области, в которой пользователь задаёт граф, что выполняется в методе `onGraphPanelMouseDown()`.

Вся отрисовка графа происходит в методе (обработчике события `onPaint` элемента управления `_graphPanel`) `onGraphPanelPaint()`. При этом свою работу он делегирует классу `GraphVisualizer`, конструктору которого необходимо передать объект класса `Graph` (граф, который необходимо нарисовать) и после чего необходимо вызвать метод `addGraph()`, передав ему в качестве параметра объект класса `Graphics` (полотно, на которое будет добавлен граф). Сам же визуализатор графа `GraphVisualizer` использует в своём “арсенале” визуализатор вершин и визуализатор рёбер графа – `VertexVisualizer` и `EdgeVisualizer` соответственно.

Реализация алгоритма Ли инкапсулирована в класс `AlgorithmLi`.

4.3.1 Алгоритм Ли

Алгоритм Ли позволяет найти кратчайший путь от одной вершины к другой, например, в неориентированном графе. Для нахождения кратчайшего пути согласно алгоритму Ли необходимо выполнить следующие действия:

1. Выделить начальную и конечную вершины графа;
2. Около начальной вершины, выставить пометку «0»;
3. Указать на то, что текущей вершиной является начальная.
4. Для всех смежных с текущей вершиной вершин, которые ещё не имеют пометок, необходимо поставить пометку, на единицу больше той, что стоит около текущей вершины;
5. Каждую вершину, около которой поставили пометку, необходимо поочередно сделать текущей и перейти на шаг 4; помечать вершины можно до тех пор, пока на шаге 4 не будет помечена ни одна вершина, или до тех пор, пока не будет помечена конечная вершина;
6. После того, как конечная вершина помечена, необходимо её назначить текущей;
7. Найти вершину, смежную текущей и имеющую пометку, на единицу меньше, чем у текущей; выделить её (добавив в будущем найденный кратчайший путь); повторять данный шаг до тех пор, пока весь путь не будет найден.

4.3.2 Задача Коммивояжёра

Для поиска пути обхода вершин в графе из вершины v_0 в вершину v_0 методом ветвей и границ необходимо выполнить следующие шаги:

1. Составить матрицу расстояний, строкам и столбцам которой назначить множество вершин графа; ячейкам которой назначить расстояние от вершины, соответствующей строке матрицы, до вершины, соответствующей столбцу матрицы;

2. Добавить в матрицу расстояний дополнительный столбец, заполнив его ячейки минимальными значениями из остальных ячеек каждой из строк;
3. Вычесть из каждой ячейки матрицы, кроме ячеек дополнительного столбца значение, стоящее в той же строке дополнительного столбца;
4. Добавить в матрицу расстояний дополнительную строку, заполнив её ячейки минимальными значениями из остальных ячеек каждого из столбцов;
5. Вычесть из каждой ячейки матрицы, кроме ячеек дополнительных столбца и строки значение, стоящее в том же столбце дополнительной строки;
6. Выполнить оценку ячеек, имеющих в качестве значения «ноль»; для этого необходимо получить сумму минимального значения по строке (не считая текущую ячейку) и минимального значения по столбцу (не считая текущую ячейку);
7. Выбрать одну из нулевых ячеек такую, для которой оценка оказалась максимальной;
8. Пометить её значение бесконечным; пометить значение ячейки, столбец которой соответствует вершине строки исходной ячейки, а строка которой соответствует вершине столбца исходной ячейки;
9. Затем необходимо удалить столбец и строку исходной ячейки;
10. Затем необходимо удалить строку или столбец, в котором оказалось две ячейки, содержащих бесконечные значения, предварительно добавив соответствующие строке и столбцу пару вершин, которые будут составляющей найденного пути;
11. Повторять шаги 2-10 до тех пор, пока не будут задействованы все вершины;
12. Подкорректировать найденные участки пути таким образом, чтобы они образовали путь.

4.3.3 Цветовая схема

В программу была добавлена поддержка выбора цветовой схемы. Цветовая схема задаётся в конфигурационном файле в формате xml с названием colors.xml. Пример содержимого данного файла изображен на рисунке 11.

```
<?xml version="1.0" encoding="utf-8" ?>
<colors>
  <upPanel desc="Фоновый цвет верхней панели">158, 166, 243</upPanel>
  <main desc="Основной цвет">158, 166, 243</main>
  <selBtnUpPanel desc="Фоновый цвет выделенной кнопки верхней панели">255, 208, 83</selBtnUpPanel>
  <btnUpPanelFore desc="Цвет текста кнопок верхней панели">45, 62, 221</btnUpPanelFore>
  <selBtnRightPanel desc="Фоновый цвет выделенной кнопки правой панели">255, 208, 83</selBtnRightPanel>
  <btnRightPanel desc="Фоновый цвет кнопок правой панели">158, 166, 243</btnRightPanel>
  <borderBtnRightPanel desc="Цвет рамки кнопок правой панели">66, 81, 227</borderBtnRightPanel>
  <btnRightPanelFore desc="Цвет текста кнопок правой панели">45, 62, 221</btnRightPanelFore>
  <mainSplit desc="Цвет основного разделителя">97, 110, 233</mainSplit>
  <rightSplit desc="Цвет правого разделителя">66, 81, 227</rightSplit>
  <edge desc="Цвет ребра">45, 62, 221</edge>
  <selEdge desc="Цвет ребра найденного кратчайшего пути">255, 208, 83</selEdge>
  <vertex desc="Цвет вершины">158, 166, 243</vertex>
  <borderVertex desc="Цвет границы вершины">66, 81, 227</borderVertex>
  <selVertex desc="Цвет вершины в найденном пути">255, 208, 83</selVertex>
  <selBorderVertex desc="Цвет границы вершины в найденном пути">66, 81, 227</selBorderVertex>
  <foreVertex desc="Цвет текста внутри вершины">45, 62, 221</foreVertex>
  <textBesideVertex desc="Цвет текста около вершины">255, 208, 83</textBesideVertex>
  <textInTable desc="Цвет текста в таблице">255, 208, 83</textInTable>
</colors>
```

Рисунок 11 – Пример содержимого файла colors.xml

В данном файле задаются различные цвета, которые должны иметь какие-либо объекты программы.

Конфигурация загружаются при создании основной формы окна и считывается с помощью класса XDocument, встроенного в .NET Framework.

Для преобразования текста вида «<число>, <число>, <число>» в экземпляр класса Color, используется следующий код:

```
/// <summary>
/// Класс, инкапсулирующий в себе методы расширения для работы с цветом
/// </summary>
public static class MyColorExtension {
  /// <summary>
  /// Получить цвет из текста
  /// </summary>
  /// <param name="colorInString">Цвет, заданный текстом</param>
  /// <returns>Цвет</returns>
  public static Color toColor(this string colorInString) {
    var rgb = colorInString.Split(',').Select(b => int.Parse(b.Trim())).ToList();
    return Color.FromArgb(rgb[0], rgb[1], rgb[2]);
  }
}
```

То есть вначале мы разделяем исходную строку на три части, соответствующие числам в строковом формате, введенным через запятую. Затем

числа из строкового представления преобразуется в числовой вид (предварительно из строк удаляются обрамляющие пробельные символы) с помощью статического метода `Parse` класса `int`. Затем вызывается статический метод `FromArgb()` класса `Color` с тремя параметрами, соответствующими интенсивности каждого из трёх цветов: красный, зеленый и синий, с целью создания экземпляра класса `Color`.

Все полученные цвета затем задаются свойствам различных элементов управления и полей классов, созданных с целью визуализации графов.

4.3.4 Реализация анимации

Анимация реализована с помощью класса `System.Windows.Forms.Timer`. Метод обработки события `Tick()` (срабатывания таймера) выбирает тот или иной шаг алгоритма в соответствии с алгоритмом. При этом при каждом срабатывании таймера меняются лишь свойства некоторых объектов, отрисовка происходит в соответствии с новыми значениями свойств объектов.

Отрисовка происходит в обработчике события `Paint` соответствующего элемента управления. Для указания на тот факт, что требуется переотрисовка элемента управления, используется метод `Invalidate()` соответствующего элемента управления.

4.3.5 Визуализация графа

Для визуализации графа был создан класс `GraphVisualizer`, который в конструкторе в качестве параметра принимает экземпляр класса `Graph`, описывающий модель графа. Для добавления на конву (экземпляр класса `Graphics`) графа, необходимо вызвать метод `addGraph()` класса `GraphVisualizer`, который вначале рисует все рёбра графа, вызывая для каждого ребра метод `addEdge()` соответствующего визуализатора рёбер `EdgeVisualizer`, а затем все

вершины графа, вызывая для каждой вершины метод `addVertex()` соответствующего визуализатора вершин `VertexVisualizer`.

```
/// <summary>
/// Добавляет изображение, соответствующее графу в настоящий момент времени.
/// </summary>
/// <param name="graphics">Канва, на которой необходимо нарисовать граф</param>
public void addGraph(Graphics graphics) {
    //Рисуем рёбра графа
    _graph.Edges.ToList().ForEach(e => new EdgeVisualizer(e).addEdge(graphics));
    //Рисуем вершины графа
    _graph.Vertices.ToList().ForEach(v => new VertexVisualizer(v).addVertex(graphics));
}
```

Для рисования ребра графа используется метод `DrawLine()` класса `Graphics`. При этом, чтобы учитывался эффект сглаживания `AntiAlias`, свойство `SmoothingMode` экземпляра класса `Graphics` задаётся в `SmoothingMode.AntiAlias`. Если отрисовывается ребро согласно алгоритму Ли, то учитывается, что данное ребро может быть помеченным как часть пути, что требует применения условного оператора.

```
/// <summary>
/// Добавляет ребро к графу
/// </summary>
/// <param name="graphics">Канва, на которой необходимо нарисовать ребро</param>
public void addEdge(Graphics graphics) {
    graphics.SmoothingMode = SmoothingMode.AntiAlias;

    var p = new Pen(edgeColor, 2);
    if (_edge is AlgLiEdge && (_edge as AlgLiEdge).PathMarked) {
        p = new Pen(selEdgeColor, 2);
    }
    graphics.DrawLine(p, _edge.Vertex1.Position, _edge.Vertex2.Position);
}
```

4.3.6 Взаимосвязь классов

Представим на рисунке 12 диаграмму классов программного обеспечения «Студграф».

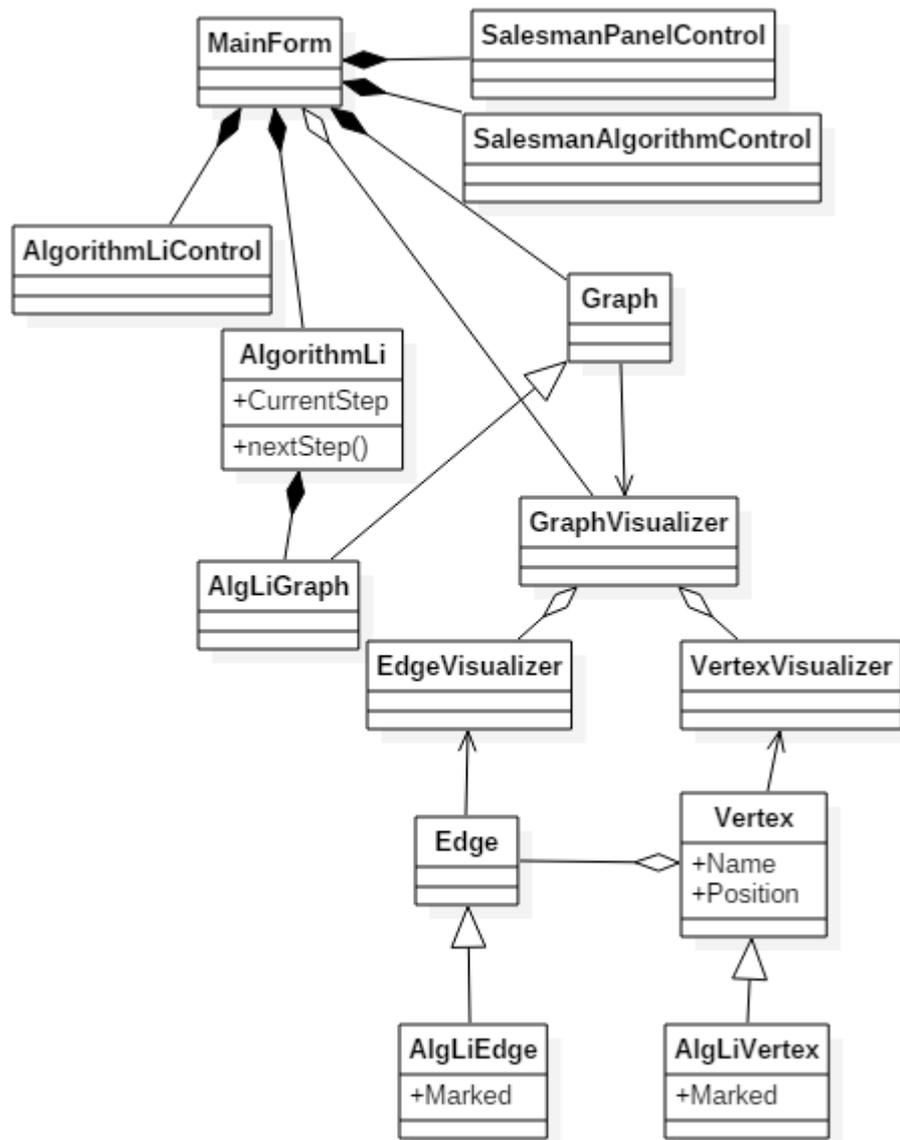


Рисунок 12 – Диаграмма классов

Из диаграммы классов видно, что основная форма программы создаёт её дочерние элементы управления: `AlgorithmLiControl`, `SalesmanPanelControl`, `SalesmanAlgorithmControl`. Помимо прочего она создаёт граф и при отрисовке панели, в которой визуализируется граф, создаёт визуализатор графа, которому передаётся созданный основной формой (`Graph`) или экземпляром класса `AlgorithmLi` (`AlgLiGraph`) граф. Визуализатор графа использует визуализатор ребер и визуализатор вершин для отрисовки объектов, из которых он состоит. При этом выделяются ребра обычного графа и рёбра (наследуются от обычных) графа для отрисовки этапов волнового алгоритма (`AlgLiEdge`), в котором помимо

прочих данных необходимо учитывать, например, такую составляющую, как помеченность ребра. То же касается вершин графа. Таким образом, при необходимости добавления в программу других алгоритмов над графами, требующих отрисовку дополнительных элементов при вершинах или при рёбрах, достаточно будет наследоваться от классов `Vertex`, `Edge` и дописать реализацию, соответствующую специфике отрисовки вершин и рёбер для этих алгоритмов.

5 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

Целью данного раздела является комплексное описание и анализ финансово-экономических аспектов выполненной работы.

При этом решаются следующие задачи:

- 1) Оценка коммерческого потенциала и перспективности проведения научных исследований;
- 2) Определение возможных альтернатив проведения научных исследований, отвечающих современным требованиям в области ресурсоэффективности и ресурсосбережения;
- 3) Планирование научно-исследовательских работ;
- 4) Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования.

5.1 Организация и планирование работ

Перечень работ и продолжительность их выполнения в рамках проведения научного исследования приведен в таблице 3.1.

Таблица 3.1 – Перечень работ и продолжительность их выполнения

№	Этапы работы	Исполнители	Загрузка исполнителей
1	Составление и утверждение технического задания	НР	НР – 100%
2	Подбор и изучение материалов по теме	НР, И	НР – 60% И – 100%
3	Изучение уже существующих решений в данной области	И	И – 100%
4	Выбор направления исследований	НР, И	НР – 50% И – 100%
5	Календарное планирование работ по теме	НР, И	НР – 100% И – 100%

Продолжение таблицы 3.1

Теоретическая разработка структуры алгоритма, решающего поставленную задачу	И	И – 100%
Построение пробного варианта алгоритма	И	И – 100%
Оценка результатов и, при необходимости, итеративное внесение изменений в структуру алгоритма	И, НР	И – 100% НР – 100%
Сравнение результатов работы конечного варианта алгоритма с другими решениями в данной области	И	И – 100%
Оценка эффективности полученных результатов	И, НР	И – 100% НР – 100%
Оценка целесообразности проведения дальнейших исследований по данной теме	И, НР	И – 100% НР – 100%

Для оценки трудоемкости выполнения работ необходимо оценить минимальное и максимальное время выполнения каждой работы. Произведем расчет ожидаемого значения трудоемкости по следующей формуле:

$$t_{ож,i} = \frac{(3t_{min} + 2t_{max})}{5} \quad (1)$$

Временные показатели научного исследования приведены в таблице 3.2. Календарный план-график приведен в таблице 3.3. Коэффициент $K_{вн}$, учитывающий влияние внешних факторов на соблюдение предварительно определенных длительностей выбран равным единице, то же справедливо для коэффициента $K_{д}$, учитывающего дополнительное время на компенсацию непредвиденных задержек и согласование работ. Таким образом, $t_{ож}$ и $T_{рд}$ совпадают.

Таблица 3.2 – Временные показатели научного исследования.

Этап	Исполнители	Продолжительность работ, дни			Трудоемкость работ по исполнителям чел.-дн.			
					$T_{РД}$		$T_{КД}$	
		t_{min}	t_{max}	$t_{ож}$	НР	И	НР	И
1	2	3	4	5	6	7	8	9
Составление и утверждение технического задания	НР	7	14	9,8	9,8	–	15	–
Подбор и изучение материалов по теме	НР, И	20	40	28	16,8	28	25	42
Изучение уже существующих решений в данной области	И	15	30	21	–	21	–	31
Выбор направления исследований	НР, И	5	10	7	3,5	7	5	10
Календарное планирование работ по теме	НР, И	5	10	7	7	7	10	10
Теоретическая разработка структуры алгоритма, решающего поставленную задачу	И	20	50	32	–	32	–	48
Построение пробного варианта алгоритма	И	15	25	19	–	19	–	28
Оценка результатов и, при необходимости, итеративное внесение изменений в структуру алгоритма	НР, И	30	50	38	38	38	57	57
Сравнение результатов работы конечного варианта алгоритма с другими решениями в данной области	И	7	14	9,8	–	9,8	–	15
Оценка эффективности полученных результатов	НР, И	5	10	7	7	7	10	10
Оценка целесообразности проведения дальнейших исследований по данной теме	НР, И	5	10	7	7	7	10	10
Итого:				185,6	89,1	175,8	132	261

Таблица 3.3 – Линейный график работ.

Этап	Н	И	Продолжительность выполнения работ																										
			Сент			Окт			Ноябрь			Дек			Февр			Март			Апр			Май			Июнь		
			10	20	30	10	20	30	10	20	30	10	20	30	10	20	30	10	20	30	1	2	3	1	2	3	1	2	3
1	15	–	■																										
2	25	42				■																							
3	–	31							■																				
4	5	10										■																	
5	10	10										■																	
6	–	48										■																	
7	–	28																■											
8	57	57																			■								
9	–	15																									■		
10	10	10																									■		
11	10	10																									■		

НР – ■; И – ■

Оценим текущие состояния результатов работы над проектом.

Таблица 3.4 – Нарастание технической готовности работы и удельный вес каждого этапа

Этап	ТР _i , %	СГ _i , %
Составление и утверждение технического задания	3,70	3,70
Подбор и изучение материалов по теме	16,91	20,61
Изучение уже существующих решений в данной области	7,93	28,54
Выбор направления исследований	3,96	32,50
Календарное планирование работ по теме	5,29	37,79
Теоретическая разработка структуры алгоритма, решающего поставленную задачу	12,08	49,87
Построение пробного варианта алгоритма	7,17	57,04
Оценка результатов и, при необходимости, итеративное внесение изменений в структуру алгоритма	28,69	85,73
Сравнение результатов работы конечного варианта алгоритма с другими решениями в данной области	3,70	89,43
Оценка эффективности полученных результатов	5,29	94,71
Оценка целесообразности проведения дальнейших исследований по данной теме	5,29	100,00

5.2 Расчет сметы затрат на выполнение проекта

Цель данного раздела – оценка текущих состояний (результатов) работы над проектом. Величина накопления готовности работы показывает, на сколько процентов по окончании текущего (i-го) этапа выполнен общий объем работ по проекту в целом.

Степень готовности проекта определяется следующей формулой:

$$СГ_i = \frac{ТР_i^H}{ТР_{общ}} = \frac{\sum_{k=1}^i ТР_k}{ТР_{общ}} = \frac{\sum_{k=1}^i \sum_{j=1}^m ТР_{km}}{\sum_{k=1}^I \sum_{j=1}^m ТР_{km}}, \quad (2)$$

где ТР_{общ} – общая трудоемкость проекта;

TP_i (TP_k) – трудоемкость i -го (k -го) этапа проекта;

$i = \overline{1, I}$ – индекс этапа;

TP_i^H – накопленная трудоемкость i -го этапа проекта по его завершении;

TP_{ij} (TP_{ik}) – трудоемкость работ, выполняемых j -м участником на i -м этапе;

$j = \overline{1, m}$ – индекс исполнителя.

Исходя из таблицы 4.4 величины TP_{ij} (TP_{kj}) находятся в девятом и десятом столбцах таблицы. $TP_{\text{общ}}$ равна сумме чисел из итоговых ячеек этих столбцов.

В статье затрат на материалы относим стоимость материалов и других материальных ценностей, расходуемых непосредственно в процессе выполнения работ над проектом.

Приведём расчёт затрат на материалы в таблице 3.5.

Таблица 3.5 – Расчёт затрат на материалы

Наименование материалов	Цена за ед., руб.	Кол-во	Сумма, руб.
Фотобумага для принтера 170 гр 127*8*50,8	270	3 ролика	810
Бумага для принтера формата А4	190	1 уп.	150
Картридж для принтера	1550	1 шт.	1550
Итого:			2510

Допустим, что ТЗР составляют 5 % от отпускной цены материалов, тогда расходы на материалы с учетом ТЗР равны $C_{\text{мат}} = 2510 * 1,05 = 2635,5$ руб.

Раздел расчета заработной платы включает заработную плату научного руководителя и исполнителя проекта, а также премии, входящие в фонд заработной платы.

Среднедневная тарифная заработная плата рассчитывается по следующей формуле:

$$ЗП_{\text{дн-т}} = \frac{МО}{24,83}, \quad (3)$$

где МО – величина месячного оклада.

Данная формула учитывает, что в году 298 рабочих дней и, следовательно, в месяце в среднем 24,83 рабочих дня (при шестидневной рабочей неделе).

Для учета в заработной плате премий, дополнительной зарплаты и районной надбавки используется следующий ряд коэффициентов:

$$K_{\text{ПР}} = 1,1;$$

$$K_{\text{доп ЗП}} = 1,188;$$

$$K_{\text{р}} = 1,3.$$

Для того, чтобы перейти от тарифной суммы заработка исполнителя проекта к соответствующему полному нужно воспользоваться следующей формулой:

$$ЗП_{\text{полн}} = ЗП_{\text{дн-т}} \cdot K_{\text{ПР}} \cdot K_{\text{доп ЗП}} \cdot K_{\text{р}} = ЗП_{\text{дн-т}} \cdot 1,699 \quad (4)$$

Рассчитаем заработную плату научного руководителя и инженера. Результаты расчётов приведены в таблице 3.6. Месячный оклад научного руководителя в соответствии с таблицей окладов ППС и НС, должностью и степенью составляет 23 264,86, что учтено в таблице.

Таблица 3.6 – Затраты на заработную плату.

Исполнитель	Оклад, руб./мес.	Среднедневная ставка, руб./раб.день	Затраты времени, раб.дни	Коэффициент	Фонд з/платы, руб.
НР	23 264,86	936,97	89	1,699	140 783,76
И	14 584	587,35	176	1,62	167 465,23
Итого					308 249

Затраты на единый социальный налог (ЕСН), включающий в себя отчисления в пенсионный фонд, на социальное и медицинское страхование, составляют 30 % от полной заработной платы по проекту, т.е.

$$C_{\text{соц.}} = C_{\text{зп}} * 0,3 \quad (5)$$

Итак, в нашем случае $C_{\text{соц.}} = 308249 * 0,3 = 92,474$ руб.

Расходы на электроэнергию включают в себя затраты на электроэнергию, потраченную в ходе выполнения проекта на работу используемого оборудования, рассчитываемые по формуле:

$$C_{\text{эл.об.}} = P_{\text{об}} \cdot t_{\text{об}} \cdot Ц_{\text{э}} \quad (6)$$

где $P_{\text{об}}$ – мощность, потребляемая оборудованием, кВт;

$Ц_{\text{э}}$ – тариф на 1 кВт·час;

$t_{\text{об}}$ – время работы оборудования, час.

Время работы оборудования вычисляется на основе итоговых данных таблицы 1.2 для инженера ($T_{\text{рд}}$) из расчета, что продолжительность рабочего дня равна 8 часов.

$$t_{\text{об}} = T_{\text{рд}} \cdot K_t \quad (7)$$

где $K_t \leq 1$ – коэффициент использования оборудования по времени, равный отношению времени его работы в процессе выполнения проекта к $T_{\text{рд}}$, определяется исполнителем самостоятельно. В рамках выполнения данного проекта коэффициент K_t для ноутбука составил 1

Затраты на электроэнергию технологическую приведены в таблице 3.7.

Таблица 3.7 – Затраты на электроэнергию технологическую.

Наименование оборудования	Время работы оборудования $t_{\text{об}}$, час	Потребляемая мощность $P_{\text{об}}$, кВт	Затраты $Э_{\text{об}}$, руб.
Персональный компьютер	504*0,6	0,3	476,92
Струйный принтер	30	0,1	15,77
Итого:			492,69

В статье «Амортизационные отчисления» рассчитывается амортизация используемого оборудования за время выполнения проекта.

Расчет амортизационных расходов осуществляется по формуле 1.13.

$$C_{\text{ам}} = \frac{N_A \cdot Ц_{\text{об}} \cdot t_{\text{рф}} \cdot n}{F_d} \quad (8)$$

где N_A – годовая норма амортизации единицы оборудования;
 $Ц_{ОБ}$ – балансовая стоимость единицы оборудования с учетом ТЗР;
 F_D – действительный годовой фонд времени работы соответствующего оборудования;

$t_{рф}$ – фактическое время работы оборудования в ходе выполнения проекта;
 n – число задействованных однотипных единиц оборудования.

Годовая норма амортизации рассчитывается по следующей формуле:

$$N_A = \frac{1}{CA}, \quad (9)$$

где CA – срок амортизации оборудования.

Из постановления правительства «О классификации основных средств, включенных в амортизационные группы» срок амортизации для электронно-вычислительной техники составляет свыше двух лет до трех лет включительно.

Стоимость ПК 45000 руб., время использования 504 часа, тогда для него $C_{AM}(ПК) = (0,4*45000*504*1)/2384 = 3805,37$ руб. Стоимость принтера 12000 руб., его $F_D = 500$ час.; $N_A = 0,5$; тогда его $C_{AM}(Пр) = (0,5*12000*30*1)/500 = 360$ руб. Итого начислено амортизации 4 165,37 руб.

В статье «Прочие расходы» отражены расходы на выполнение проекта, которые не учтены в предыдущих статьях. Данный тип расходов рассчитывается по следующей формуле:

$$C_{проч} = (C_{мат} + C_{ЗП} + C_{соц} + C_{эл.об.} + C_{AM}) \cdot 0,1, \quad (10)$$

Произведём расчет расходов, учитываемых непосредственно на основе платежных документов. Время пребывания в командировке составило 50 календарных дней (с учетом дней приезда и отъезда); оплата проживания в общежитии 50 руб./день*45 дней = 2250 руб. (основные расходы за счет принимающей стороны); оплата проезда по ж.д. в обе стороны – 4720 руб.; аренда специальных приборов – 0 руб.; почтовые расходы – 0 руб.; консалтинговые услуги – 0 руб. Итого по данному пункту $C_{пр} = (50 - 1)*100 + 2250 + 4720 + 0 + 0 + 0 = 11\ 870$ руб.

Рассчитаем сумму прочих расходов как 10% от суммы всех предыдущих расходов.

$$C_{\text{проч.}} = (C_{\text{мат}} + C_{\text{зп}} + C_{\text{соц}} + C_{\text{эл.об.}} + C_{\text{ам}} + C_{\text{нп}}) \cdot 0,1 = (2635,5 + 308\,249 + 92,474 + 492,69 + 4165,37 + 11\,870) \cdot 0,1 = 327505 \cdot 0,1 = 32750,5 \text{ руб.}$$

Проведя расчет по всем статьям сметы затрат на разработку, определим общую себестоимость проекта «Разработка информационной системы «Студграф». Смета затрат на разработку проекта представлена в таблице 3.8.

Таблица 3.8 – Смета затрат на разработку проекта

Статья затрат	Условное обозначение	Сумма, руб.
Материалы и покупные изделия	$C_{\text{мат}}$	2635,5
Основная заработная плата	$C_{\text{зп}}$	147475,68
Отчисления в социальные фонды	$C_{\text{соц}}$	44242,7
Расходы на электроэнергию	$C_{\text{эл.}}$	492,69
Амортизационные отчисления	$C_{\text{ам}}$	4165,37
Непосредственно учитываемые расходы	$C_{\text{нр}}$	11 870
Прочие расходы	$C_{\text{проч}}$	21088,19
Итого:		231 970,13

Таким образом, затраты на разработку составили $C = 231\,970,13$ руб.

Рассчитаем прибыль от реализации проекта, это 20% от расходов на разработку проекта, что составит 46 394,03 руб. НДС составляет 18% от суммы затрат на разработку и прибыли.

$$(231\,970,13 + 46\,394,03) \cdot 0,18 = 278\,364,16 \cdot 0,18 = 50\,105,55 \text{ руб.}$$

Цена разработки НИР равна сумме полной себестоимости, прибыли и НДС, т.е. $C_{\text{НИР(КР)}} = 231\,970,13 + 46\,394,03 + 50\,105,55 = 328\,469,71$ руб.

5.3 Оценка экономической эффективности проекта

Так как работа имеет чисто дидактическую направленность на совершенствование преподавания курса "Теория графов" и "Структуры и алгоритмы обработки данных", то оценка экономического эффекта не корректна.

5.4 Оценка научно-технического уровня НИР

Научно-технический уровень характеризует влияние проекта на уровень и динамику обеспечения научно-технического прогресса в данной области. Для оценки научной ценности, технической значимости и эффективности использовался метод балльных оценок.

Сущность метода заключается в том, что на основе оценок признаков работы определяется интегральный показатель ее научно-технического уровня, рассчитываемый по следующей формуле:

$$I_{НТУ} = \sum_{i=1}^3 R_i \cdot n_i \quad (12)$$

где $I_{НТУ}$ – интегральный индекс научно-технического уровня;

R_i – весовой коэффициент i -го признака научно-технического эффекта;

n_i – количественная оценка i -го признака научно-технического эффекта в баллах.

Оценки научно-технического уровня данной работы представлены в виде таблицы 3.9.

Таблица 3.9 – Оценка научно-технического уровня НИР

Значимость	Фактор НТУ	Уровень фактора	Выбранный балл	Обоснование выбранного балла
0,4	Уровень новизны	Относительно новая	3	Облегчает выбор алгоритма сегментации

Продолжение таблицы 3.9

0,1	Теоретический уровень	Разработка программы	7	Разработка программы со множеством алгоритмов для различных задач
0,5	Возможность реализации	В течение первых лет	10	Приложение сразу готово к использованию

Используя данные таблицы 3.9 рассчитаем показатель научно-технического уровня для данного проекта:

$$I_{НТУ} = 0,4 \cdot 3 + 0,1 \cdot 7 + 0,5 \cdot 10 = 6,9.$$

В таблице 3.10 указано соответствие качественных уровней НИР значениям показателя.

Таблица 3.10 – Соответствие качественных уровней НИР значению интегрального индекса научно-технического уровня

Уровень НТЭ	Показатель НТЭ
Низкий	1-4
Средний	4-7
Высокий	8-10

Исходя из данных, указанных в таблице 2.10, данный проект имеет средний уровень научно-технического эффекта.

6 Социальная ответственность

Согласно ГОСТ Р ИСО 26000-2012 [1] социальная ответственность – ответственность организации за воздействие её решений и деятельности на общество и окружающую среду через прозрачное и этическое поведение, которое содействует устойчивому развитию, включая здоровье и благосостояние общества; учитывает ожидания заинтересованных сторон; соответствует применяемому законодательству и согласуется с международными нормами поведения; интегрировано в деятельность всей организации и применяется в её взаимоотношениях.

В данном разделе будут описаны различные мероприятия, которые были приняты для устранения возможных причин сбоев в работе разрабатываемой системе, и рекомендации по предотвращению тех причин, возникновение которых не может быть предусмотрено в работе самой системы.

6.1 Анализ возможных сбоев и их последствий

Данная информационная система предназначена для использования в учебных целях, использование в военной инженерии и на промышленных производствах не предполагается. Некорректная работа системы не может привести к чрезвычайной ситуации и не связана с повышенным риском для здоровья пользователя. Но при этом не стоит упускать возможность возникновения определенных нештатных ситуаций и важно продумать пути их устранения.

Неправильная реализация графического интерфейса системы может спровоцировать ошибки пользователя программы или же замедлять время его работы до неприемлемой величины. Например, ошибка в наименовании кнопок для работы с визуализацией алгоритмов приведет к ошибочным действиям пользователя и предоставлению программой некорректной информации.

Сбой возможен в связи с возможной утечкой памяти, так как в программе имеются неуправляемые ресурсы, обёрнутые в управляемую оболочку. Если во время работы программы будет происходить постепенная утечка памяти, то размер оперативной памяти на компьютере, на котором запущена данная программа, будет сокращаться, что в конечном итоге приведёт к тому, что исполняемая программа израсходует всю память, ограниченную системой для одного приложения в 4 Гбайта.

Сбой в программе возможен из-за подмены системных библиотек сторонним программным обеспечением, в том числе вирусами. В этом случае может исполниться любой код, например, во время запуска программного обеспечения, компьютер запустит системные библиотеки для того, чтобы визуализировать алгоритм, но так как системные библиотеки были подменены, то в результате, исполнится код злого вирусописателя, который может привести к таким последствиям, как выгрузка всех личных данных, включая хранящиеся, в том числе и в зашифрованном виде, на компьютере паролей, в сеть Интернет. Далее, возможно причинить вред аппаратному обеспечению ПК, если код вирусописателя использует так называемые "дыры" в операционной системе, под управлением которой находится разработанный продукт; "дыры" могут пропустить код злого вирусописателя к нулевому кольцу защиты и дать ему прямой доступ к аппаратуре персонального компьютера (ПК), что может повлечь за собой механические повреждения составляющих ПК.

Возможно возникновение ситуации, когда пользователь случайно, либо целенаправленно произведет действия, которые могут привести к сбою в системе. В процессе выполнения такого действия возникнет ошибка, которая будет выведена пользователю, а само действие будет прекращено. Работа же системы прекращена не будет.

Каждый из перечисленных источников ошибки может породить большое число субъективных или физических неисправностей, которые необходимо

локализовать и устранить. Субъективные неисправности отличаются от физических тем, что после обнаружения, локализации и коррекции больше не возникают. Однако, как следует из перечня источников ошибок, субъективные неисправности могут быть внесены на этапе разработки спецификации системы, а это означает, что даже после самых тщательных испытаний системы на соответствие ее внешним спецификациям в системе могут находиться субъективные неисправности.

Надо обратить внимание на то, что неправильное предоставление информации студенту или школьнику ведет за собой множество последующих ошибок, связанных с его дальнейшей учебной и профессиональной деятельностью. Например, студент получивший не правильную информацию в период своего обучения, в дальнейшем попытается использовать свои знания на практике, при работе с синтаксическим анализатором, при составлении путей сложных химических реакций, изготовлении микросхем или в других сферах связанных с теорией графов, может совершить ошибки, что приведет к серьезным последствиям для предприятия и общества в целом.

Ошибка в информационной системе и в дальнейшем ее использовании на практике может привести к составлению неправильной транспортной системы, например, не эффективном и неправильном расположении пожарных частей города или составлению не эффективных маршрутов авиационных перелетов, составлению не правильной маршрутизации данных в интернете, когда данные отправленные от одного компьютера могут проходить слишком много промежуточных серверов, чтобы приблизиться к компьютеру-получателю.

Многие постановки экономического содержания сводятся к задаче коммивояжера. Например, составить наиболее выгодный маршрут обхода наладчика в цехе (контролера, охранника, милиционера), отвечающего за должное функционирование заданного множества объектов (каждый из этих объектов моделируется вершиной графа), составить наиболее выгодный

маршрут доставки деталей рабочим или хлеба с хлебозавода по заданному числу булочных и других торговых точек, не правильное решения задачи при определении выгодного пути приведет к экономическим потерям для организации.

6.2 Методы и технологии встроенной защиты от ошибок

Известно, что практически во всех более или менее сложных программах (а также в операционной системе, представляющих собой комплекс многих программ) имеются ошибки. Причина этого очевидна - программы составляют люди, а людям свойственно ошибаться. Уже на этапе проектирования нужно предусмотреть максимальное количество факторов, влияющих на возникновение сбоев и постараться защитить систему от их влияния.

Учитывая рассмотренные особенности действий человека при переводе можно указать следующие пути борьбы с ошибками:

- сужение пространства перебора (упрощение создаваемых систем),
- обеспечение требуемого уровня подготовки разработчика (это функции менеджеров коллектива разработчиков),
- обеспечение однозначности интерпретации представления информации,
- контроль правильности перевода (включая и контроль однозначности интерпретации).

Встроенная система защиты от сбоев реализована с помощью:

– Отключения доступности элементов управления, когда работа программы при их доступности имеет неопределенный характер. Пример: когда запускаем алгоритм решения задачи Коммивояжёра, кнопки запуска алгоритма и применения другого количества городов временно становятся

недоступными, что достигается с помощью свойства `Enabled` соответствующих элементов управления;

– Включение условных операторов в программный код с целью проверки неожиданных случаев.

– Выбор элементов управления, ограничивающих множество значений, которые пользователь может ввести.

Пример: для задания количества городов в задаче Коммивояжера используется элемент управления `NumericUpDownValue`, предназначенное для ввода только чисел, а не обычное текстовое поле `TextBox`, в которое пользователь может ввести произвольные строки.

– Ограничение множества значений, которые пользователь может ввести, с помощью соответствующих свойств элементов управления.

Пример: для того, чтобы пользователь не мог ввести отрицательное число городов, для которого работа алгоритма не определена, использовано свойство `MinValue` класса (элемента управления) `NumericUpDown`, которое задаёт минимальное значение, которое может ввести пользователь в данный элемент управления.

– Отключение возможности редактирования графов при включенной визуализации. Например, если пользователь попытается добавить вершину графу при включенной визуализации, программа не позволит выполнить данную команду. Так же отключение возможности добавления города и установки расстояния между городами в момент визуализации работы задачи коммивояжера.

6.3 Мероприятия по защите здоровья и безопасности пользователя

Согласно ГОСТ Р ИСО 26000-2012 [1] защита здоровья и безопасности пользователя включает предоставление продуктов и услуг, являющихся безопасными и не представляющих неприемлемого риска причинения ущерба при использовании или потреблении.

Разработанное программное обеспечение представляет из себя информационную систему, предназначенную для использования в учебных целях, а также для решения прикладных задач. Применение данного программного обеспечения предполагает использование персональной электронной вычислительной машины (ПЭВМ), что влечет за собой определенные риски для здоровья и безопасности пользователя. Так, например, электромагнитное излучение от (ПЭВМ) распространяется во всех направлениях и может оказать вредное воздействие на пользователя. На работающего за ПЭВМ действуют электромагнитные поля (ЭМП) от различных источников электромагнитного излучения, таких как системный блок (магнитопроводы, электромагниты и т.д.), монитор (электронно-лучевая трубка), внутренняя часть монитора: катушки строчной и кадровой развертки и т.д.), клавиатура и мышь (в процессе трения поверхности электризуются). Основным источником ЭМП при работе за ПЭВМ является монитор с ЭЛТ. Необходимо использовать жидкокристаллические мониторы, так как они обладают пониженным уровнем излучения.

ЭМП имеет следующее воздействие на человека: может вызвать головную боль, утомляемость, различные заболевания нервной, иммунной, эндокринной и половой систем.

С целью обеспечения электромагнитной безопасности необходимо принять следующие меры согласно СанПиН 2.2.2/2.4.1340-03 [10]:

- 1) Использовать защитные экранные фильтры;
- 2) Размещать дисплей на расстоянии 50 см от глаз;
- 3) Организовать время для отдыха;

4) Использовать нейтрализаторы ЭМП.

Для предупреждения преждевременной утомляемости пользователей ПЭВМ рекомендуется организовывать рабочую смену путем чередования работ с использованием ПЭВМ и без него.

Если характер работы требует постоянного взаимодействия с ПЭВМ без переключения на другие виды деятельности, не связанные с ПЭВМ, рекомендуется организация перерывов на 10.15 минут через каждые 45.60 минут работы.

Продолжительность непрерывной работы с видеодисплейным терминалом (ВДТ) без регламентированного перерыва не должна превышать 1 час.

Использование защитных экранных фильтров так же помогает в защите зрения от избыточного светового потока, уменьшает ультрафиолетовое и рентгеновское излучение, устраняет блики, обеспечивает защиту от электростатического заряда, образующего на трубке монитора.

При использовании ПЭВМ присутствуют риски поражения электрическим током, воздействия электрической дуги и статического электричества.

Основными причинами поражения человека электрическим током являются короткое замыкание, искровые разряды.

С целью обеспечения электробезопасности электроустановок следует принимать следующие меры согласно СанПиН 2.2.2/2.4.1340-03 [10]:

- 1) Помещения с ПЭВМ должны быть оборудованы защитным заземлением, не превышающим 4 Ом;
- 2) Сопротивление изоляции не менее 5 МОм;
- 3) Размещение рабочих мест с электрооборудованием вдали от силовых кабелей;

Также, должны быть введены следующие запреты:

- 1) Запрещено прикасаться к задней панели монитора, системного блока;
- 2) Запрещено вскрывать оборудование, производить ремонт;
- 3) Запрещено перемещать силовые кабели на системы отопления;

Положительное психофизиологическое состояние человека, здоровое зрение во многом зависят от правильной освещенности рабочего места. При плохом освещении наблюдается напряжение зрения, ослабление внимания, преждевременная утомляемость, раздражение и резь в глазах, дезориентация.

Выбор системы освещения зависит, прежде всего, от такого важного фактора, как точность выполняемых зрительных работ (наименьший размер объекта различения).

6.4 Мероприятия по защите оборудования

Поскольку разрабатываемый проект представляет собой программный комплекс, то логично, что воздействие факторов, влияющих на физическую платформу, на базе которой будет разворачиваться система, невозможно предотвратить средствами самой системы. Для защиты необходимо обеспечить платформу физическими средствами, которые позволят игнорировать данные факторы.

Для блокирования (парирования) случайных угроз безопасности информации в компьютерных системах должен быть решен комплекс задач. Дублирование информации является одним из самых эффективных способов обеспечения целостности информации. Оно обеспечивает защиту информации как от случайных угроз, так и от преднамеренных воздействий. В качестве внешних запоминающих устройств для хранения дублирующей информации используются накопители на жестких магнитных дисках и магнитных лентах. Накопители на жестких магнитных дисках применяются обычно для оперативного дублирования информации.

Пользователи довольно часто разделяют служебную и личную информацию, по различным причинам, среди них ежедневные вирусные атаки, повышенная конфиденциальность данных, служебная информация, которую не стоит оставлять на рабочем ПК по ряду причин. Именно это и предполагает использование внешних накопителей, жестких дисков.

Для надежной защиты компьютера от сбоев с электропитанием рекомендуется использовать специальный прибор – источник бесперебойного питания. Его характерной особенностью является то, что компьютер питается именно от него, а не непосредственно из сети. Иначе говоря, источник бесперебойного питания – это своеобразный буфер между электрической сетью и компьютером. В его состав, помимо прочего, входит аккумуляторная батарея, средний срок службы которой – от трех до пяти лет. Эта батарея позволяет корректно завершить работу компьютера и спокойно выключить его даже после внезапного отключения электроэнергии.

Кроме этого, источник бесперебойного питания «сглаживает» любые перепады напряжения в сети, защищая тем самым персональный компьютер от связанных с этим поломок. Такие меры значительно снизят риски неожиданной потери информации и торможению работы.

Сетевое оборудование, обеспечивающее соединение между компьютерами, на которых используется система, и сервером, на котором система непосредственно расположена, так же должно быть подключено к источнику бесперебойного питания. для того, чтобы предоставить возможность закончить передачу данных на сервер и не потерять их.

Обезопасить целостность кабельной линии компьютера можно путем установки специализированного кабель канала и дополнительной прокладки

кабеля в гофре. На рисунке 13 представлен способ защита кабельной линии компьютера.



Рисунок 13 – Способ защиты кабельной линии (кабель канал)

Заключение

В ходе проделанной работы была спроектирована система программных классов, был выполнен ввод графа визуальным способом, реализован алгоритм Ли и решена задача коммивояжёра методом ветвей и границ, в результате чего был разработан программный продукт «Студграф».

Conclusion

The following tasks have been solved:

1. Analyzed ways of applying for graphs;
2. Software class system was designed;
3. Entering graph was implemented with visual method;
4. The algorithm Li was implemented;
5. The solution to the traveling salesman problem was implemented by branch and bound method;
6. The most simple and user-friendly graphical user interface was developed.

After all of implemented stuffs was received program product named by «Студграф».

Список использованных источников

1. ГОСТ Р ИСО 26000-2012 Руководство по социальной ответственности.
2. Домнин Л.М. Элементы теории графов: учебное пособие — М.: Изд-во Пензенского государственного университета, 2007—144с.
3. Граф (математика) // Википедия. [Электронный ресурс] / URL: <http://ru.wikipedia.org/?oldid=75568978> (дата обращения: 15.04.2016).
4. Касьянов В.В. Графы в программировании. Обработка, визуализация и применение: научное издание — М.: Изд-во ВHV-СПБ издательство, 2003—1150с.
5. Шилдт, Г.. С# 4.0: полное руководство. Пер. с англ. / Герберт Шилдт. - М.: ООО "И.Д. Вильяме", 2011. - 1056 с.
6. Уотсон, К. Visual C# 2010: полный курс. Пер. с англ./ Карли Уотсон, Кристиан Нейгел, Якоб Хаммер Педерсен, Джон Д. Рид, Морган Скиннер. - М.: Диалектика, 2010. - 960 с.
7. Сайт вопросов и ответов для профессиональных программистов и программистов-энтузиастов. [Электронный ресурс] / URL: <http://stackoverflow.com/>. – Дата обращения: 15.04.2016 г.
8. Интеграция LINQ с C# // Профессорвеб. [Электронный ресурс] / URL: (дата обращения: 15.04.2016).
9. Графы и их применение // Интуит. [Электронный ресурс] / URL: <http://www.intuit.ru/studies/courses/58/58/info> (дата обращения: 15.04.2016).
10. СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы. Средства защиты от излучений оптического диапазона и электромагнитных полей ПЭВМ.

11. Петцольд Ч. Программирование с использованием Microsoft Windows Forms. Мастер-класс. Пер. с англ. - СПб: Питер, 2006. - 432 с.
12. Жарков В.А. Компьютерная графика, мультимедиа и игры на Visual C# 2005.- М: Жарков-Пресс, 2005. - 812 с.
13. ГОСТ Р 12.1.009-2009 Система стандартов безопасности труда. Электробезопасность. Термины и определения.
14. СНиП III-42-80 Охрана окружающей среды.
15. C#. Спецификация языка. Версия 4.0. / Microsoft, 1999-2010. – 567 с.
16. Подразделение компании Майкрософт, ответственное за взаимодействие фирмы с разработчиками (Microsoft Developer Network) [Электронный ресурс] / URL: <https://msdn.microsoft.com/ru-ru/default.aspx>. – Дата обращения: 30.05.2016 г.

Приложение А

Техническое задание

Введение

В данном разделе документа представлено техническое задание. Данная разработка представляет собой программное обеспечение «СтудГраф», которое применяется для визуализации методов решения следующей алгоритмической задачи над графами: поиск кратчайшего пути между двумя выбранными вершинами.

1. Основания для разработки

1.1. Документ, на основании которого ведется разработка

Задание на выпускную квалификационную работу.

1.2. Организация, утвердившая документ

ООО «Энергопромавтоматика».

1.3. Наименование и (или) условное обозначение темы разработки

Программа «Студграф».

2. Назначение разработки

Данная разработка представляет собой программу, работающую под управлением операционной системы Windows. Программа предназначена для задания неориентированных графов визуальным способом и поиска кратчайшего пути между двумя выбранными вершинами с визуализацией результата с применением алгоритма Ли.

3. Требования к программе или программному изделию

3.1. Требования к функциональным характеристикам

Для данной разработки применяются следующие требования к функциональным характеристикам:

- Программа должна предоставлять пользователю возможность задания неориентированных графов визуальным способом;

- Программа должна предоставлять результаты реализации алгоритма Ли посредством генерации последовательности кадров на экран монитора;
- Программа должна предоставлять результаты решения задачи коммивояжера.

3.2. Требования к надежности

Отсутствуют.

3.3. Требования к составу и параметрам технических средств

Для функционирования программы необходим компьютер IBM/PC совместимый и следующие технические средства:

- процессор Intel или совместимый;
- объем свободной оперативной памяти 164670 Кб;
- объем необходимой памяти на жестком диске 2048 Мб;
- стандартный VGA-монитор или совместимый, с поддерживаемым разрешением от 1024x768;
- мышь;
- клавиатура.

3.5. Требования к информационной и программной совместимости

Отсутствуют.

3.6. Специальные требования

Отсутствуют.

3.7. Стадии и этапы разработки

1. Согласование и получение технического задания;
2. Создание эскизного проекта;
3. Разработка рабочего проекта;
4. Отладка и устранение ошибок;
5. Написание отчета и защита работы.

Приложение Б

Методическое указание к выполнению лабораторной работы по теории алгоритмов на тему: «Решение задачи коммивояжера методом ветвей и границ»

Цель работы: получить теоретические знания и практические навыки для решения задач коммивояжера методом ветвей и границ.

Постановка задачи

Требуется решить задачу коммивояжера для четырех городов. Матрица расстояний представлена в таблице 1.

Таблица 1 Исходная матрица расстояний

i \ j	1	2	3	4
1	∞	5	11	9
2	10	∞	8	7
3	7	14	∞	8
4	12	6	15	∞

Порядок выполнения работы

1. Познакомиться с методическим указанием.
2. Найти минимумы по строкам в исходной матрице.
3. Выполнить редукцию строк.
4. Найти минимумы по столбцам.
5. Выполнить редукцию столбцов.
6. Выполнить оценку нулевых строк.
7. Выполнить редукцию матрицы.
8. Если полный путь еще не найден, переходим к пункту 2, если найден к пункту 9.
9. Построить итоговый маршрут.

Методические указания

Имеются n городов, расстояния между городами известны. Коммивояжер должен пройти все n городов по одному разу, вернувшись в тот город, с которого начал. Требуется найти такой маршрут движения, при котором суммарное пройденное расстояние (стоимость проезда и т.д.) будет минимальным.

Очевидно, что задача коммивояжера – это задача отыскания кратчайшего гамильтонова цикла в полном графе.

Для нахождения оптимального решения задачи коммивояжера применяется метод ветвей и границ.

Посчитаем города за вершины графа, а коммуникации (i, j) – дугами, то задача поиска минимального пути, проходящего через каждый город только по одному разу и возвращение обратно, можно рассмотреть, как нахождения на графе гамильтоново контура минимальной длины.

Для практической реализации идеи метода ветвей и границ необходим метод поиска и определения нижних границ подмножества и разбиение множества гамильтоновых контуров на подмножества.

Поиск нижних границ предполагает, что если ко всем элементам i -й строки или j -го столбца матрицы C прибавить или отнять число α , то задача останется эквивалентной прежней, то есть оптимальность маршрута коммивояжера не изменится, а длина любого гамильтонова контура изменится на данную величину α .

Опишем алгоритм метода ветвей и границ для нахождения минимального гамильтонова пути для графа с n вершинами. Граф представляют в виде матрицы его дуг. Если между вершинами X_i и X_j нет дуги, то ставится символ « ∞ ».

Сформулируем основные правила для решения задачи коммивояжера:

- 1) Находим в каждой строке матрицы минимальный элемент и вычитаем его из всех элементов соответствующей строки.

- 2) Если в приведенной матрице окажутся столбцы, не содержащие нуля, то приведем ее по столбцам. Для этого находим минимальное значение в каждом столбце и вычитаем его из всех элементов соответствующего столбца. Такая матрица называется приведенной по строкам и столбцам.
- 3) После приведения матрицы по строкам и столбцам появятся нулевые клетки. Найдем степени нулей, для этого нули в матрице заменяем на знак бесконечности « ∞ » и найдем сумму минимальных элементов строки и столбца, соответствующих этому нулю.
- 4) Выбираем дугу, для которой степень нулевого элемента достигает максимального значения. Заменяем ее значением бесконечности. Данная дуга будет первым отрезком пути коммивояжера.
- 5) Ту строку и тот столбец где образовались две бесконечности полностью вычеркиваем.
- 6) Если не нашли все отрезки пути, то снова произведем указанные пункты выше, найдем минимальные значения по строкам и столбам, проведем их редукцию и т.д.
- 7) Если все отрезки пути найдены, остается только соединить их вместе с собой.

Алгоритм решение задачи методом ветвей и границ.

1) Справа к исходной таблице присоединяем столбец d_i , в котором записываем минимальные элементы соответствующих строк. Вычитаем элементы d_i из соответствующих элементов строки матрицы. Поиск минимума в каждой строке представлен в таблице 2.

Таблица 2 – Поиск минимума в каждой строке

i \ j	1	2	3	4	d_i
1	∞	5	11	9	5
2	10	∞	8	7	7

Продолжение таблицы 2

3	7	14	∞	8	7
4	12	6	15	∞	6

2) Производим редукцию строк – из каждого элемента в строке вычитаем соответствующее значение найденного минимума. В таблице 3 представлена редукция строк исходной матрицы.

Таблица 3 – Редукция строк

i \ j	1	2	3	4
1	∞	0	6	4
2	3	∞	1	0
3	0	7	∞	1
4	6	0	9	∞

3) В нижней части полученной матрицы добавляем строку d_j , в которой записываем минимальные элементы столбцов. Далее вычитаем d_j из соответствующих столбцов матрицы. Присоединение строки с минимальными значениями к исходной матрице представлено в таблице 4.

Таблица 4 – Поиск минимального значения в каждом столбце

i \ j	1	2	3	4
1	∞	0	6	4
2	3	∞	1	0
3	0	7	∞	1
4	6	0	9	∞
d_j	0	0	1	0

4) Вычитаем из каждого элемента матрицы соответствующее ему d_j . Результат вычисления представлен в таблице 5.

Таблица 5 – Редукция столбцов

i \ j	1	2	3	4
1	∞	0	5	4
2	3	∞	0	0
3	0	7	∞	1
4	6	0	8	∞
d_j	0	0	1	0

В результате в каждом столбце появилась хотя бы одна нулевая клетка.

5) Для каждой нулевой клетки получившейся преобразованной матрицы находим «оценку». Ею будет сумма минимального элемента по строке и минимального элемента по столбцу, в которых размещена данная нулевая клетка. Найденные ранее d_i и d_j не учитываются. Полученную оценку записываем рядом с нулем, в скобках. Вычисленные оценки нулевых клеток представлены в таблице 6.

Таблица 6 – Вычисление оценок нулевых клеток

i \ j	1	2	3	4
1	∞	0^4	5	4
2	3	∞	0^5	0^1
3	0^4	7	∞	1
4	6	0^6	8	∞

б) Выбираем нулевую клетку с наибольшей оценкой. Она равна 6 и соответствует клетке (4;2). Таким образом, одним из вариантов на включение в гамильтонов контур является дуга (4;2). Ту строку и тот столбец, где образовалось две «М» полностью вычеркиваем. В клетку соответствующую обратному пути ставим еще одну букву «М» (т.к. не будем возвращаться обратно). В таблице 7 продемонстрировано удаление строки и столбца из

таблицы.

Таблица 7 – Удаление из таблицы строки и столбца

i \ j	1	2	3	4
1	∞	0^4	5	4
2	3	∞	0^5	∞
3	0^4	7	∞	1
4	6	0^6	8	∞

7) После сокращения таблицы необходимо вновь повторить редукцию строк и столбцов. Результат вычисления представлен в таблице 8.

Таблица 8 – Результат редукции строк и столбцов

i \ j	1	3	4
1	∞	1	0^2
2	3	0^4	∞
3	0^4	∞	1

Далее вновь выбираем нулевую клетку с наибольшей оценкой. И продолжаем редукцию строк и столбцов пока не найдется все отрезки пути.

8) После того как все отрезки пути найдены, остается только соединить их между собой. Длины дорог, соединяющих города, берем из самой первой таблицы с исходными данными. В нашем примере маршрут получился следующий: 1-4-2-3-1

Заключение

Задача коммивояжера является типичной задачей оптимизации, которая широко применяется при разработке программного обеспечения. Задача о коммивояжере является упрощенной моделью для многих других задач дискретной оптимизации, а также часто является подзадачей. В своей области

(оптимизации дискретных задач) она служит своеобразным катализатором, стимулирующим разработку наиболее эффективных методов, алгоритмов и способов их машинной реализации.

Применение задачи коммивояжера на практике довольно обширно. В частности ее можно использовать для поиска кратчайшего маршрута при гастролях эстрадной группы по городам, нахождения последовательности технологических операций обеспечивающей наименьшее время выполнения всего производственного цикла.

Задание для самостоятельной работы:

1. Применив алгоритм ветвей и границ, решите задачу коммивояжера для 4 городов, используя следующую матрицу стоимостей проезда, представленную в таблице 9:

Таблица 9 – Исходная матрица

i \ j	1	2	3	4
1	∞	15	11	20
2	17	∞	10	6
3	7	12	∞	13
4	18	5	15	∞

2. Решите задачу коммивояжера методом ветвей и границ, исходная матрица расстояний задана в таблице 10:

Таблица 10 – Исходная матрица

i \ j	Омск	Томск	Астана	Абакан	Красноярск
Омск	∞	880	670	1450	1450
Томск	900	∞	1250	800	590
Астана	690	1300	∞	1900	2050

Продолжение таблицы 10

Абакан	1500	750	1850	∞	445
Красноярск	1400	580	2000	440	∞

Проверьте результаты вычислений с помощью программы «Студграф».