**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский

технический университет им. А.Н. Туполева-КАИ»

(КНИТУ-КАИ)

Институт компьютерных технологий и защиты информации

Кафедра систем автоматизированного проектирования

Направление подготовки:09.03.01 - «Информатика и вычислительная техника»

Образовательная программа: «Искусственный интеллект и системы автоматизированного проектирования»

|  |
| --- |
| **К защите допустить** |
| **Зав. каф. \_\_\_\_\_\_\_\_\_** |
| **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(ФИО)** |
| **«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.** |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему: «Оптимизация перемещения беспилотного такси в условиях города»

ОБУЧАЮЩИЙСЯ Бикмуллин А.Р.

*(фамилия, имя, отчество) (подпись)*

РУКОВОДИТЕЛЬ к.т.н., доцент Невзоров В.Н.

*(ученая степень, звание, фамилия, имя, отчество) (подпись)*

Казань 2024 г.

Optimization of the movement of an unmanned taxi in the city

*By*

*Bikmullin Amir Rostislavovich*

Submitted to the Department of Computer-Aided Design

in partial fulfillment of the Requirements for the degree of

BACHELOR OF SCIENCE

at the

Federal State Budgetary Educational Institution of Higher Education

«Kazan National Research Technical University named after A.N.Tupolev-KAI»

(KNRTU-KAI)

|  |  |  |  |
| --- | --- | --- | --- |
| Author |  |  | Bikmullin Amir Rostislavovich |
| *(signature)* |  |  |
| Supervisor |  |  | Nevzorov Vladimir Nikolaevich |
| *(signature)* |  | Candidate of Science, Associate Professor Department of Computer-Aided Design |
| Certified by |  |  | Chermoshentsev Sergey Fedorovich |
| *(signature)* |  | Doctor of Technical Science, Head of the Department of Computer-Aided Design |
| date |  |  |  |
|  |

Kazan 2024

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования «Казанский национальный исследовательский технический**

**университет им. А.Н. Туполева-КАИ»**

**(КНИТУ-КАИ)**

Институт (факультет), филиал : институт компьютерных технолоий и защиты информации

Кафедра: систем автоматизированного проектирования

Направление подготовки/специальность: 09.03.01(Информатика и вычислительная техника)

**ЗАДАНИЕ**

на выпускную квалификационную работу

обучающегося: Бикмуллина Амира Ростиславовича

(фамилия, имя, отчество)

1 Тема выпускной квалификационной работы:

Оптимизация перемещения беспилотного такси в условиях города

утверждена приказом №\_\_\_\_\_\_\_\_\_\_\_ от «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

2 Срок сдачи обучающимся законченной ВКР «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

3 Исходные данные к выпускной квалификационной работе:

План размещения пассажиров, карта города

4 Перечень подлежащих разработке вопросов и исходные данные к ним:

**Глава 1. Анализ проектной процедуры оптимизации перемещения беспилотного такси в условиях города**

1.1 Современные тенденции и проблемы процедуры оптимизации перемещения беспилотного такси в условиях города

1.2 Разработка функциональных и поведенческих моделей (IDEF0 и IDEF3) процедуры оптимизации перемещения беспилотного такси в условиях города.

1.3 Обзор систем автоматизации перемещения беспилотного такси в условиях города.

1.4 Цель и задачи процедуры оптимизации перемещения беспилотного такси в условиях города.

Вывод по главе 1.

**Глава 2. Разработка метематического обеспечения для автоматизации процедуры оптимизации перемещения беспилотного такси в условиях города**

2.1 Содержательная и математическая постановка задачи оптимизации перемещения беспилотного такси в условиях города

2.2 Методы и алгоритмы решения задачи оптимизации перемещения беспилотного такси в условиях города.

2.3 Разработка алгоритма задачи оптимизации перемещения беспилотного такси в условиях города.

Вывод по главе 2.

**Глава 3. Разработка информационного обеспечения для процедуры оптимизации перемещения беспилотного такси в условиях города**

3.1 Разработка модели потока данных (DFD) и концептуальной модели базы данных.

3.2 Разработка логической модели базы данных

3.3 Физическое проектирование базы данных.

Вывод по главе 3

**Глава 4. Разработка програмного обеспечения для процедуры оптимизации перемещения беспилотного такси в условиях города**

4.1 Архитектура программы для автоматизации процедуры оптимизации перемещения беспилотного такси в условиях города.

4.2 Разработка ползовательского интерфейса программы

4.3 Примеры решения задачи оптимизации перемещения беспилотного такси в условиях города.

Выводы по главе 4

Заключение

Список литературы

Приложение 1. Глоссарий (словарь терминов предметной области)

Приложение 2. Образцы проектной документации.

Приложение 3. Листнинг программы.

5 Перечень графического материала (при наличии):

1. Цель, задачи выпускной квалификационной работы (2 слайдa).
2. Функциональная модель IDEF0 процедуры оптимизации перемещения беспилотного такси в условиях города.(3 слайда)
3. Формальная (математическая) постановка задачи оптимизации перемещения беспилотного такси в условиях города. (1 слайд)
4. Aлгоритмы решения задачи оптимизации перемещения беспилотного такси в условиях города. (1 слайд)
5. Логическая модель базы данных (1 слайд)
6. Концептуальная модель базы данных (1 слайд)
7. Архитектура программы для процедуры оптимизации перемещения беспилотного такси в условиях города. (1 слайд)
8. Экранные формы программы (1 слайда)
9. Примеры решения задачи оптимизации перемещения беспилотного такси в условиях города. (3 слайда)
10. Основные выводы по выпускной квалификационной работе (1 слайд)

6 Консультанты по ВКР (при их наличии, с указанием относящихся к ним разделов):

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

*(наименование раздела, ФИО консультанта, подпись)*

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

*(наименование раздела, ФИО консультанта, подпись)*

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

*(наименование раздела, ФИО консультанта, подпись)*

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

*(наименование раздела, ФИО консультанта, подпись)*

Дата выдачи задания «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_г.

Руководитель ВКР \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (ФИО)

Задание к исполнению принял \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (ФИО)

**Календарный план выполнения ВКР**

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов (разделов)  выпускной квалификационной работы | Срок выполнения  этапов (разделов) ВКР | Примечание |
| *1.* | *Современные тенденции и проблемы процедуры оптимизации перемещения беспилотного такси в условиях города.* | *31.10.2023-12.03.2024г.* |  |
| *2.* | *Разработка функциональных и поведенческих моделей (IDEF0 и IDEF3) процедуры оптимизации перемещения беспилотного такси в условиях города.* | *13.03.2024-29.03.2024г.* |  |
| *3.* | *Обзор алгоритмов оптимизации перемещения беспилотного такси в условиях города.* | *20.03.2024-26.03.2024г.* |  |
| *4.* | *Цель и задачи процедуры оптимизации перемещения беспилотного такси в условиях города.* | *27.03.2024-2.04.2024г.* |  |
| *5.* | *Содержательная и математическая постановка задачи оптимизации перемещения беспилотного такси в условиях города.* | *3.04.2024-9.04.2024г.* |  |
| *6.* | *Методы и алгоритмы решения задачи оптимизации перемещения беспилотного такси в условиях города.* | *10.04.2024-16.04.2024г.* |  |
| *7.* | *Разработка алгоритма задачи оптимизации перемещения беспилотного такси в условиях города.* | *17.04.2024-23.04.2024г.* |  |
| *8.* | *Разработка модели потока данных (DFD) и концептуальной модели базы данных.* | *24.04.2024-30.04.2024г.* |  |
| *9.* | *Разработка логической модели базы данных* | *1.05.2024-7.05.2024г.* |  |
| *10.* | *Физическое проектирование базы данных* | *8.05.2024-14.05.2024г.* |  |
| *11.* | *Архитектура программы для автоматизации процедуры оптимизации перемещения беспилотного такси в условиях города.* | *15.05.2024-21.05.2024г.* |  |
| *12.* | *Разработка пользовательского интерфейса программы* | *22.05.2024-28.05.2024г.* |  |
| *13.* | *Примеры решения задачи оптимизации перемещения беспилотного такси в условиях города.* | *29.05.2024-2.06.2024г.* |  |
| *14.* | *Подготовка к защите выпускной квалификационной работы* | *3.06.2024г.* |  |

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бикмуллин А.Р \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись) (Фамилия, инициалы) ( дата)*

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись) (Фамилия, инициалы) (дата)*

**Аннотация**

Данная выпускная квалификационная работа посвящена разработке математического, информационного и программного обеспечения для процедуры оптимизации перемещения беспилотного такси в условиях города.

В работе можно найти: исследование тенденций и проблем задачи оптимизации перемещения беспилотного такси в условиях города, алгоритмы нахождения минимального пути, разработка собственной базы данных для ведения статистики и отчетности, пример решения оптимизации перемещения беспилотного такси в условиях города.

Ключевые слова: беспилотное такси, такси, перемещение, оптимизация перемещения, программное обеспечение, беспилотный транспорт.

Количество страниц: 126

Количество иллюстраций:41

Количество таблиц: 15

Количество приложений:2

Количество использованных источников:50

**Annotation**

This graduate qualification work is devoted to the development of mathematical, informational and software for the procedure of optimizing the movement of an unmanned taxi in the city.

In the work you can find: a study of trends and problems of optimizing the movement of an unmanned taxi in the city, algorithms for finding the minimum path, the development of our own database for statistics and reporting, an example of a solution for optimizing the movement of an unmanned taxi

Keywords: unmanned taxi, taxi, moving, optimization of movement, software, unmanned transport.

Number of pages: 126

Number of illustrations:41

Number of tables: 15

Number of applications:2

Number of sources used:50

СОДЕРЖАНИЕ

[Введение 5](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620244)

[Глава 1. АНАЛИЗ ПРОЕКТНОЙ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА 6](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620245)

[1.1. Современные тенденции и проблемы процедуры оптимизации перемещения беспилотного такси в условиях города 6](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620247)

[1.2. Разработка функциональных и поведенческих моделей (IDEF0 и IDEF3) процедуры оптимизации перемещения беспилотного такси в условиях города.. 8](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620248)

[1.3. Обзор систем автоматизации перемещения беспилотного такси в условиях города. 19](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620251)

1.4. Цель и задачи процедуры оптимизации перемещения беспилотного такси в условиях города.. 21

[Вывод по главе 1. 22](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620256)

[Глава 2. РАЗРАБОТКА МАТЕМАТИЧЕСКОГО ОБЕСПЕЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА 23](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620266)

[2.1. Содержательная и математическая постановка задачи оптимизации перемещения беспилотного такси в условиях города 23](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620267)

[2.2. Методы и алгоритмы решения задачи оптимизации перемещения беспилотного такси в условиях города. 27](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620268)

[2.3. Разработка алгоритма задачи оптимизации перемещения беспилотного такси в условиях города. 30](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620273)

[Вывод по главе 2. 40](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620256)

[Глава 3. РАЗРАБОТКА ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ ДЛЯ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА 41](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620245)

[3.1. Разработка модели потока данных (DFD) и концептуальной модели базы данных.. 41](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620267)

[3.2. Разработка логической модели базы данных 50](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620268)

[3.3. Физическое проектирование базы данных. 53](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620273)

[Вывод по главе 3. 58](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620256)

Глава 4. РАЗРАБОТКА ПРОГРАМНОГО ОБЕСПЕЧЕНИЯ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА 59

[4.1. Архитектура программы для автоматизации процедуры оптимизации перемещения беспилотного такси в условиях города. 59](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620267)

[4.2. Разработка пользовательского интерфейса программы 78](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620268)

[4.3. Примеры решения задачи оптимизации перемещения беспилотного такси в условиях города. 85](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620273)

[Вывод по главе 4. 90](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620256)

[Заключение 91](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620276)

[Список литературы 93](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620277)

[Приложение 1. Глоссарий (словарь терминов предметной области) 100](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620278)

[Приложение 3. Листнинг программы. 10](file:///C:/Users/Пользователь/Desktop/Диплом%20бикмулл/Содержание%20Бикмул.docx#_Toc103620280)3

**Введение**

В современном мире технологии развиваются с невероятной скоростью, и автоматизация всех жизненных процессов человека выходит на новый уровень. Технологии позволяют человеку задействовать минимальное количество усилий для получения должного результата, а бизнес полностью автоматизирует свое предприятие для получения прибыли. С целью уменьшения затрат и использования человеческого труда, рассмотрим технологию беспилотного такси. Беспилотное такси - это автомобиль, который способен передвигаться без участия водителя и может быть использован для перевозки пассажиров. Развитие технологии беспилотного такси имеет огромный потенциал для улучшения безопасности дорожного движения, снижения загрязнения окружающей среды и повышения эффективности транспортной системы.

Целью данной курсовой работы является изучение основных аспектов разработки и функционирования беспилотного такси, а также анализ проблем и перспектив развития данной технологии. В ходе выполнения работы будут рассмотрены основные моменты, которые относятся к математическому обеспечению оптимизации перемещения беспилотного такси в условиях города.

**Глава 1. АНАЛИЗ ПРОЕКТНОЙ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА**

**1.1** **Современные тенденции и проблемы процедуры оптимизации перемещения беспилотного такси в условиях города**

С развитием технологий беспилотные автомобили стали одним из самых перспективных направлений в сфере транспорта. Беспилотные такси являются одним из видов беспилотных автомобилей, которые используются для перевозки пассажиров без участия водителя.

Одной из главных тенденций в области беспилотного транспорта является разработка и внедрение новых технологий, которые позволяют улучшить безопасность и эффективность работы беспилотных такси. К таким технологиям относятся системы автономного вождения, системы навигации и коммуникации, а также системы контроля и управления.

Однако, несмотря на все преимущества, существуют и проблемы, которые необходимо решить для успешного внедрения беспилотного такси в городскую среду. Одной из таких проблем является отсутствие четкой законодательной базы, которая бы регулировала работу беспилотного транспорта. Кроме того, существуют опасения по поводу безопасности использования беспилотных такси, особенно в условиях городской среды, где возможны различные непредвиденные ситуации.

Еще одной проблемой является необходимость адаптации инфраструктуры города для работы беспилотного транспорта. Это включает в себя создание специальных зон для парковки и зарядки беспилотных такси, а также разработку системы управления трафиком, которая бы учитывала особенности работы беспилотного транспорта.

С точки зрения бизнеса, существуют некоторые проблемы, которые необходимо решить для успешного развития рынка беспилотного такси. Одной из главных проблем является высокая стоимость разработки и внедрения беспилотных технологий. Также важной проблемой является конкуренция на рынке беспилотного транспорта. Многие крупные компании, такие как Uber и Lyft, уже начали разработку своих собственных беспилотных такси, что может привести к большой конкуренции на рынке и потери потенциальной прибыли.

Беспилотное такси имеет ряд преимуществ:

1. Снижение затрат на содержание парка автомобилей: Беспилотные такси не требуют водителей, что снижает затраты на заработную плату и социальные выплаты.
2. Увеличение эффективности использования транспорта: Беспилотные автомобили могут передвигаться по городу с большей эффективностью, чем автомобили с водителем, благодаря отсутствию пробок и возможности точного планирования маршрутов.
3. Улучшение качества услуг: Беспилотное такси может обеспечить более высокий уровень комфорта для пассажиров, так как оно не подвержено влиянию человеческого фактора и может работать круглосуточно.
4. Снижение уровня аварийности: Беспилотные автомобили не подвержены влиянию усталости или человеческого фактора, что снижает вероятность аварии.
5. Повышение уровня экологической безопасности: Беспилотное такси работает на электричестве, что снижает выбросы вредных веществ в атмосферу.

В целом, беспилотное такси представляет собой перспективное направление развития транспортной инфраструктуры, которое может принести значительные экономические и экологические выгоды.

## **1.2. Разработка функциональных и поведенческих моделей (IDEF0 и IDEF3) процедуры оптимизации перемещения беспилотного такси в условиях города**

IDEF0 - это методология для описания бизнес-процессов, которая позволяет моделировать логические отношения между различными видами деятельности. Она отличается от других методов тем, что основное внимание уделяется взаимосвязям между элементами процесса, а не временной последовательности.

IDEF0 представляет собой “черный ящик” с входами, выходами и управляющими факторами, которые детализируются до нужного уровня. Для каждого элемента модели есть словарь описания, который помогает понять, что означает каждый элемент.

Эта модель также включает в себя все управляющие сигналы, которые не отображались на диаграммах потоков данных (DFD). Она используется для организации бизнес-процессов и моделирования всех административных и организационных процессов.

На рисунке 1 представлена функциональная модель проектной процедуры IDEF0.

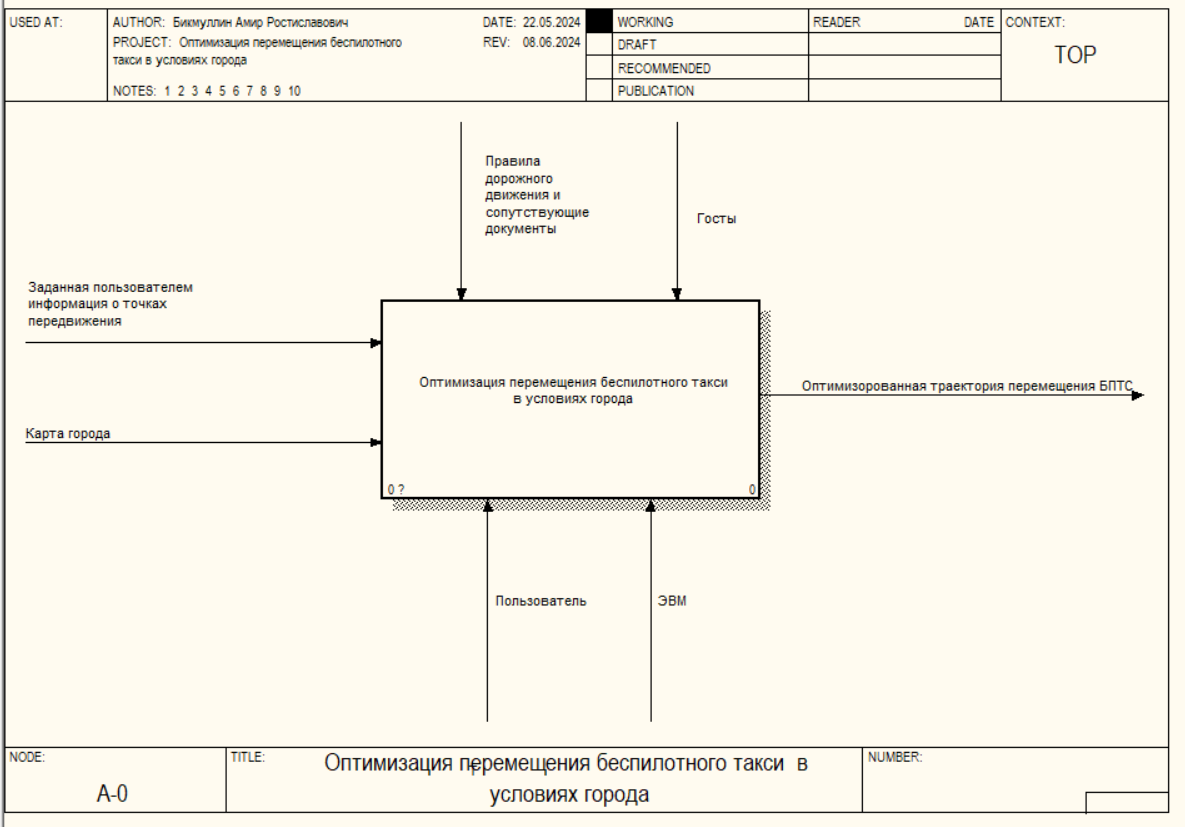


Рис. 1 Диаграмма IDEF A-0

Диаграмма IDEF A-0:

Блок «Оптимизация перемещения беспилотного такси в условиях города»:

Входные данные:   
 Заданная пользователем информация о точках передвижения – данные о координатах передвижения, введенные пользователем в программу. Обычно данные содержат информацию о: стране, городе, улице, доме посадки и доставки пассажира

Карта города - размеченная карата города, на которой присутствует информация о заторах, местах ремонта дорог, дворовых дорогах и другая необходимая информация.

Выходные данные: Оптимизированная траектория перемещения БПТС - наименьший путь, рассчитанный в компьютере, по которому будет перемещаться транспортное средство.

Управление процессом осуществляется в соответствии с нормативными документами и действиями персонала:

* ГОСТ 3.1105-84 - "Единая система технической документации (ЕСТД)". Формы и правила оформления документов общего назначения (с Изменением N 1). Данный стандарт устанавливает формы и правила оформления технических документов общего назначения, разработанных с применением различных методов проектирования.
* ГОСТ Р 56122-2014 – «Беспилотные транспортные средства. Общие технические требования», который устанавливает общие технические требования к беспилотным транспортным средствам. К сожалению, в РФ еще не существует госта для регламентирования движения наземных транспортных средств, поэтому будем руководствоваться уже существующим гостам в сфере беспилотного транспорта.
* ПДД и сопутствующие документы - совокупность документов, в которых прописаны правила поведения БПТС на дорогах общего пользования, а также правила взаимодействия с пользователем, координаты заправочных пунктов, сервисных центров и другой необходимой информации.
* ЭВМ - это устройство, которое используется для ведения учета, создания логистических маршрутов, расчета дистанции на дороге.
* Пользователь - непосредственный пользователь программы, который задает начальную и конечную точку перемещения.

На рисунке 2 представлена Диаграмма IDEF A0.

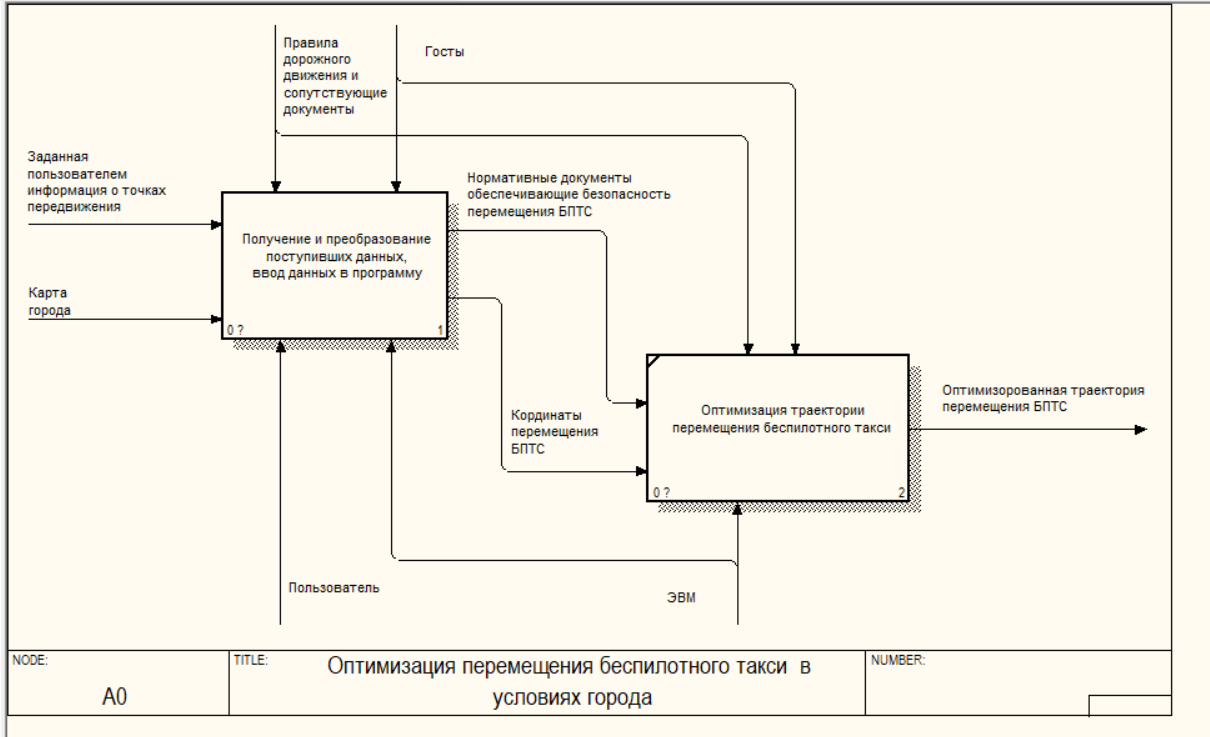
****

Рис. 2 Диаграмма IDEF A0

Диаграмма IDEF A0.

Блок «Получение и преобразование поступивших данных, ввод данных в программу»:

Входные данные в блоке «Подготовка исходных данных и введение проверенных данных в программу»: Заданная пользователем информация о точках передвижения, карта города, ПДД и сопутствующие документы.

Выходные данные в блоке «Подготовка исходных данных, введение проверенных данных в программу»:

Нормативные документы обеспечивающие безопасность перемещения БПТС - данные документы включают в себя правила дорожного движения, методические рекомендации и инструкции по перемещению в городе и взаимодействия с человеком, технические регламенты безопасности, федеральные законы.

Координаты перемещения БПТС - Координаты точек перемещения размеченные в формате [X,Y]

Управление процессом осуществляется посредством нормативных документов:

1. ГОСТ 3.1105-84 – «Единая система технологической документации (ЕСТД)»;
2. ПДД и другие документы обеспечивающие безопасное перемещение БПТС по городу

Блок «Оптимизация траектории перемещения БПТС»:

Входные данные в блоке «Оптимизация траектории перемещения БПТС»:

1. Нормативные документы обеспечивающие безопасность перемещения БПТС
2. Координаты перемещения БПТС

Выходные данные в блоке «Оптимизация траектории перемещения БПТ»: Оптимизированная траектория перемещения БПТС.

Управление процессом осуществляется посредством нормативных документов и вычислительной машины:

1. ГОСТ 3.1105-84 – «Единая система технологической документации (ЕСТД)»;
2. ГОСТ Р 56122-2014 – «Беспилотные транспортные средства. Общие технические требования»;
3. ПДД и другие документы обеспечивающие безопасное перемещение БПТС по городу
4. ЭВМ.

На рисунке 3 представлена диаграмма IDEF A1.

Изображение выглядит как текст, диаграмма, Параллельный, План

Автоматически созданное описание

Рис. 3 Диаграмма IDEF A1

Диаграмма IDEF A1.

Блок «Получение и преобразование поступивших данных, ввод данных в программу»:

Входные данные блок «Получение параметров перемещения БПТС»: Заданная пользователем информация о точках передвижения, карта города, ПДД и сопутствующие документы.

Выходные данные блока «Получение параметров перемещения БПТС»: проверенные данные – Нормативные документы обеспечивающие безопасность перемещения БПТС, местоположения объектов перемещения.

Управление процессом осуществляется посредством нормативных документов и персонала:

1. ГОСТ 3.1105-84 – «Единая система технологической документации (ЕСТД)»;
2. ПДД и другие документы обеспечивающие безопасное перемещение БПТС по городу

Блок «Преобразование данных в координатную плоскость и ввод данных в алгоритм программы»:

Входные данные блока «Преобразование данных в координатную плоскость и ввод данных в алгоритм программы»: Местоположения объектов перемещения, цифровизированная карта города.

Выходные данные блока «Преобразование данных в координатную плоскость и ввод данных в алгоритм программы»: Координаты перемещения БПТС.

**Модель IDEF3**

IDEF3 (Integrated Definition for Process Description Capture Method) - это методология, разработанная в США для описания бизнес-процессов. Она используется для моделирования и документирования процессов, связанных с информационными системами.

IDEF3 предназначена для описания процессов на уровне функциональной деятельности организации. Это значит, что методология позволяет описать, как выполняются определенные функции внутри организации, какие данные используются и какие результаты получаются.

IDEF3 использует графическую нотацию для описания процессов. Она состоит из блоков, которые представляют собой функции или действия, выполняемые в рамках процесса. Блоки соединяются линиями, которые показывают последовательность выполнения действий.

Одним из преимуществ IDEF3 является возможность использования этой методологии в сочетании с другими методами моделирования, такими как UML и BPMN. Это позволяет создавать более подробные и точные модели процессов.

На рисунке 4 представлена диаграмма IDEF3.

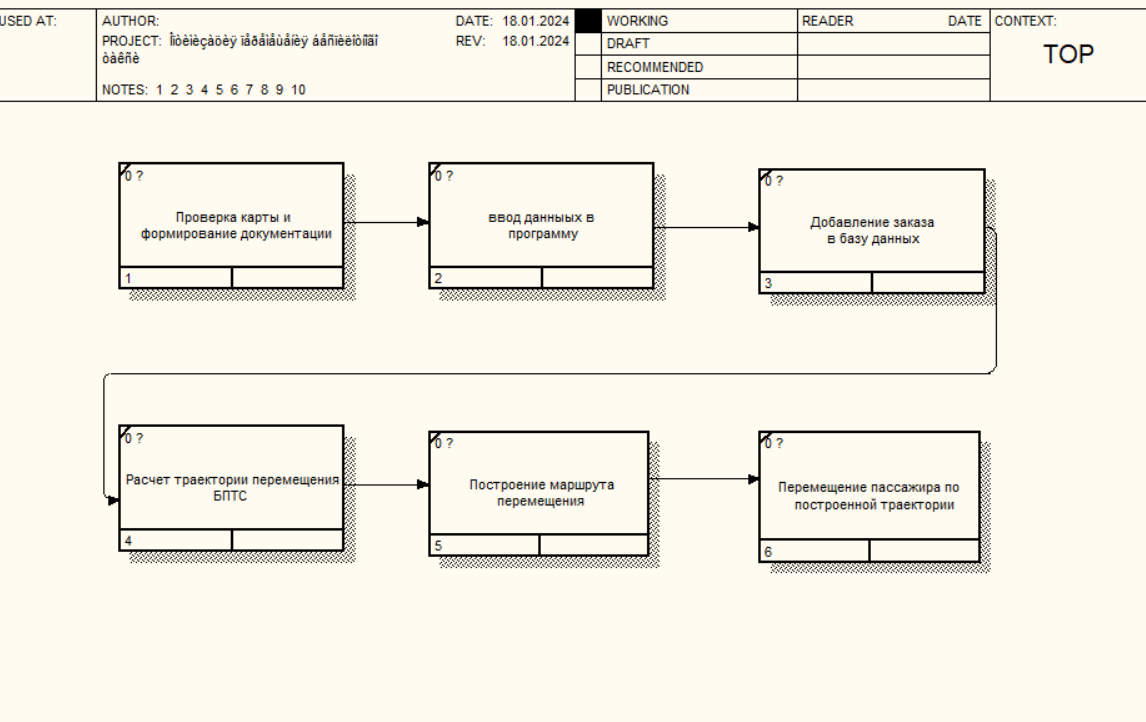


Рис. 4 Диаграмма IDEF3

Диаграмма IDEF3:

Для подготовки исходных данных процесса необходимо получить карту города и точки перемещения, а также необходимо предоставить программе документацию обеспечивающую безопасное перемещение БПТС по городу. Далее программа форматирует цифровизированную карту города и точки перемещения данной карте.

После предварительной подготовки программа начинает расчет оптимальной траектории для перемещения беспилотного транспортного средства и строит наикротчайший маршрут. По окончании, начинается движение БПТС по построенному маршруту с соблюдением прописанных в требованиях мер безопасности.

**1.3 Обзор систем автоматизации перемещения беспилотного такси в условиях города.**

В наше время не так много систем автоматизации перемещения беспилотного такси в условиях города. В силу того, что этим вопросом занялись совсем недавно и компании только начинают осваивать данные технологии, мы можем лицезреть, как такие системы постепенно внедряются в инфраструктуру разных городов мира.

Но уже сейчас есть компании, которые активно занимаются данными системами, например Yandex или Google.

Yandex начала работу над технологиями автономного вождения в 2017 году и создала подразделение Yandex Self-Driving Group. В рамках этого подразделения компания разрабатывает технологии для беспилотных автомобилей, которые включают системы компьютерного зрения, машинного обучения и высокоточные карты.

В России Yandex активно тестирует свои беспилотные автомобили в нескольких городах, включая Москву, Иннополис и другие. В Иннополисе Yandex запустила пилотный проект беспилотного такси, который предоставляет услуги по вызову такси для жителей города. Также Yandex проводила испытания своих беспилотных автомобилей в США и Израиле. В США компания тестировала свои технологии в Мичигане, а в Израиле — в Тель-Авиве.

Yandex использует такие технологии и особенности:

* Компьютерное зрение: Yandex использует передовые технологии компьютерного зрения для распознавания объектов на дороге, таких как автомобили, пешеходы, светофоры и дорожные знаки.
* Машинное обучение: Алгоритмы машинного обучения помогают беспилотным автомобилям Yandex прогнозировать поведение других участников дорожного движения и принимать решения в реальном времени.
* Высокоточные карты: Yandex создаёт детализированные карты, которые позволяют беспилотным автомобилям точно ориентироваться на дорогах и учитывать мельчайшие детали.

Yandex продолжает развивать свои технологии и расширять географию тестирования и внедрения беспилотных такси. Компания также рассматривает возможности коммерциализации своих технологий и выхода на новые рынки.

Waymo, основанная в 2009 году как проект Google Self-Driving Car Project, стала отдельной компанией в 2016 году. Waymo считается одной из наиболее продвинутых компаний в области автономного вождения благодаря длительному опыту и значительным ресурсам.

Waymo активно тестирует и предоставляет услуги беспилотных такси в районе Финикса, включая его пригороды. В 2020 году компания запустила полностью беспилотные такси без водителей в некоторых районах.

Waymo также проводит испытания своих автомобилей в Сан-Франциско, где работает над адаптацией технологий к более сложным городским условиям.

Waymo использует такие технологии и особенности:

* LIDAR, радары и камеры: Waymo использует комбинированную систему датчиков, включающую LIDAR, радары и камеры, для получения подробной информации о окружении автомобиля.
* Искусственный интеллект и машинное обучение: Waymo разрабатывает сложные алгоритмы для обработки данных с датчиков и принятия решений в реальном времени. Система Waymo способна предсказывать поведение участников дорожного движения и выбирать безопасные маршруты.
* Высокоточные карты: Waymo создает подробные карты местности с точностью до нескольких сантиметров, что позволяет автомобилям точно ориентироваться в пространстве.

Waymo продолжает расширять свою программу беспилотных такси и планирует выйти на новые рынки. Компания также исследует возможности применения своих технологий в других областях, таких как грузоперевозки и доставка товаров.

Обе компании играют важную роль в развитии технологий автономного вождения и продолжают двигать вперед будущее беспилотного транспорта.

**1.4. Цель и задачи процедуры оптимизации перемещения беспилотного такси в условиях города.**

Целью разработки автоматизации системы оптимизации перемещения беспилотного такси в условиях города является:

* Улучшение транспортной эффективности
* Обеспечение безопасности
* Интеграция с городской инфраструктурой:

Задачи процедуры оптимизации перемещения БПТС в условиях города:

– Разработка математического обеспечения для автоматизированной системы перемещения БПТС в условиях города;

– Создание информационного обеспечения;

– Разработка программного обеспечения для автоматизированной системы оптимизации перемещения БПТС в условиях города.

## **Выводы по главе 1.**

В рамках анализа процедуры оптимизации перемещения беспилотного такси в условиях города были проведены следующие работы:

1. Проведен анализ тенденции развития беспилотного такси, проблем процедуры оптимизации перемещения беспилотного такси в условиях города. и возможных конкурентов, решающих задачу оптимизации перемещения беспилотного такси.
2. Разработана функциональная модель процедуры оптимизации в виде IDEF, IDF3 диаграмм. На основе которой в дальнейшем будет разработана система оптимизации перемещения.
3. Обозначены цели и задачи оптимизации перемещения беспилотного такси в условиях города.

**Глава 2. РАЗРАБОТКА МАТЕМАТИЧЕСКОГО ОБЕСПЕЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА**

**2.1. Содержательная и математическая постановка задачи оптимизации перемещения беспилотного такси в услоовиях города**

**Содержательная постановка задачи оптимизации перемещения беспилотного такси в услоовиях города**

Основная цель системы оптимизации для беспилотного такси заключается в эффективном управлении передвижением транспортных средств в городских условиях с целью максимизации пропускной способности дорожной инфраструктуры и обеспечения оптимальных маршрутов для доставки пассажиров. Исходные данные включают в себя информацию о текущем положении транспортного средства, месте назначения пассажира, а также характеристики дорог и транспортной инфраструктуры.

Результирующие данные системы оптимизации включают в себя оптимальный маршрут перемещения беспилотного такси от точки отправления до точки назначения. Кроме того, система гарантирует, что выбранный маршрут минимизирует возможные конфликты с другими транспортными средствами, обеспечивая безопасное и эффективное передвижение в городской среде.

**Математическая постановка задачи** **оптимизации перемещения беспилотного такси в условиях города**

Необходимо рассчитать кратчайший путь от места отправления до координат места, на которое пассажир запланировал поездку.

Перечень обозначений математической модели представлен в таблице 1.

Таблица 1. Обозначения математической модели

|  |  |  |
| --- | --- | --- |
| **Обозначение компонента объекта и результата проектирования** | **Наименование компонента объекта и результата проектирования** | **Наименование соответствующего элемента математической модели** |
| Xi | Координата по оси Х | Координата точки по оси Х, которая рассматривается в данный момент для перемещения |
| Yi | Координата по оси Y | Координата точки по оси Y, которая рассматривается в данный момент для перемещения |
| g | Стоимость перехода с одной координаты в соседнюю | Вес между двумя клетками, в которых возможно перемещение |
| h(Xi,Yi) | Функция эвристического приближения | Функция, рассчитывающая расстояние от координаты, в которую совершается движение, до конечной цели |

Таблица 1. (Продолжение). Обозначения математической модели.

|  |  |  |
| --- | --- | --- |
| Fi(Xi,Yi) | Функция общей стоимости перемещения в соседнюю точку | Функция, которая вычисляется для доступных клеток, в которые можно продвигаться дальше. |
| S | Стоимость всего пути | Сумма всех Fi(Xi,Yi), которые были на пути до конечной цели |
| X2 | Координата по оси Х | Координата конечной цели маршрута по оси Х |
| Y2 | Координата по оси Y | Координата конечной цели маршрута по оси Y |
| n | Количество шагов | Показывает количество шагов, сделанных БПТС, чтобы достичь конечной точки. |

Устанавливаются координаты для перемещения БПТС. Затем выбираются координаты места, куда направится БПТС, и начинается расчет маршрута перемещения.

Изначально вокруг БПТС имеются 4 точки перемещения. Для каждой точки вычисляется функция:

fi(Xi Yi)=h(Xi Yi)+g (1), где

*g* - постоянное значение, так как БПТС может перемещаться только в соседнюю координату, и стоимость перехода по вертикали или горизонтали равна десяти .

h(u) рассчитывается по методу Манхэттена и равна:

h (X2, Y2; Xi, Yi) = | X2-Xi| + |Y2-Yi| (2), где i=1..n

(Xi, Yi) – координата точки, которая в данный момент рассматривается для перемещения;

(X2, Y2) – координата конечной точки маршрута.

Выбирается минимальное значение из всех доступных, которое будет точкой, куда БПТС следует переместиться дальше. Точка, в которой БПТС находился на предыдущем шаге, больше не участвует в расчете пути и помечается как пройденная. Далее строится маршрут и вычисляется стоимость всего пути. Для каждой точки, в которую БПТС переместился, снова рассчитывается стоимость перехода для всех доступных путей.

Если возникает ситуация, когда для перехода дальше в какой-то точке получается два одинаковых минимальных значения fi(Xi,Yi), то для каждой такой точки рассчитывается свой путь до конечной координаты. Если один из путей приводит в тупик (например, с обеих сторон есть препятствия, спереди стена, а сзади только точка, в которой находился БПТС), то этот путь отбрасывается. Однако, если оба направления ведут к конечной координате, то считается стоимость всего пути.

(3)

где, функция стремится к минимальному значению;

n - количество точек, которые прошло БПТС до конечной координаты;

fi(XiYi) – стоимость пути на каждом этапе перемещения.

Для определения оптимального маршрута сравниваются две суммы, и выбирается маршрут с наименьшей суммой.

Кроме того, проверяется условие, при котором два БПТС не могут находиться на одной и той же клетке одновременно:

X1i=X2i, Y1i=Y2i (4)

Если следующая координата, куда собирается переместиться первое БПТС, совпадает с координатами второго БПТС, второй БПТС должен подождать, пропустив первый, прежде чем продолжить свое движение. Они могут повторять маршруты друг друга, но не пересекаться в одно и то же время.

**2.2. Методы и алгоритмы решения задачи оптимизации перемещения беспилотного такси в условиях города**

Существует несколько алгоритмов поиска кратчайшего пути. Один из них - волновой алгоритм, который работает довольно просто. Он начинается с начальной точки и распространяет «волну» во все доступные стороны, ещё не задействованные на предыдущих этапах. Каждая ячейка получает значение, обычно равное единице, которое увеличивается на единицу на каждом шаге алгоритма. Начальное значение волны равно нулю. Затем обрабатываются следующие ячейки со значением волны два, при этом всегда проверяется условие, что волна не проходит через ячейки со стенами. Этот процесс повторяется до тех пор, пока не будет достигнута ячейка назначения. Путь выбирается на основе минимальной суммы значений ячеек пути. На рисунке 5 представлена визуализация работы алгоритма.

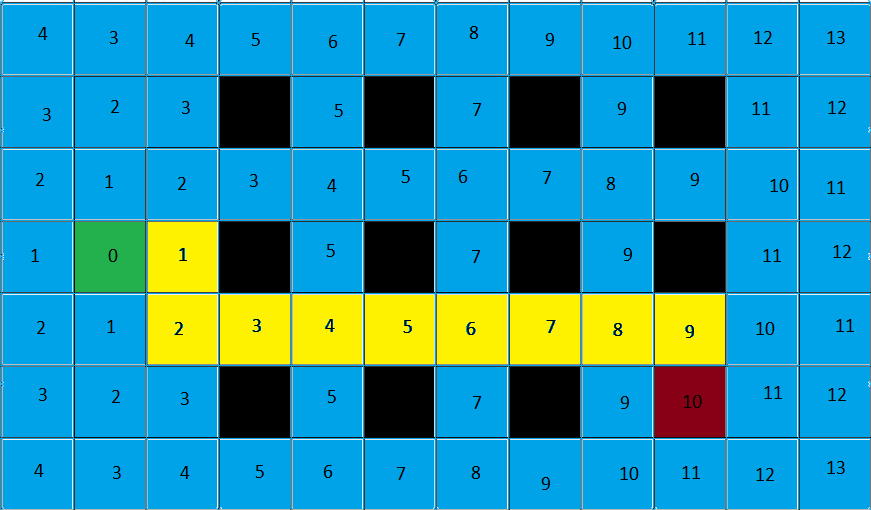


Рис. 5 Реализация волнового алгоритма

Алгоритм Дейкстры представляет собой метод поиска кратчайшего пути от заданной точки до конечной точки маршрута, учитывая стоимость перемещения между клетками. В начале каждому возможному перемещению присваивается вес (цена), который представляет собой стоимость прохождения через данную клетку. Непроходимые "стены" имеют бесконечно большую стоимость перемещения, так как БПТС не может пройти через них. Начальная точка обозначается как ноль, а конечная точка определяется как количество шагов, необходимых БПТС для достижения цели.

Минимальный маршрут определяется тем, что количество шагов, необходимых БПТС для достижения цели, минимально. Учитывая, что стоимость перемещения на всем координатном поле одинакова и равна единице, маршрут будет минимальным, когда количество шагов будет минимальным.

На практике алгоритм Дейкстры эффективно находит кратчайший путь, учитывая указанные условия.

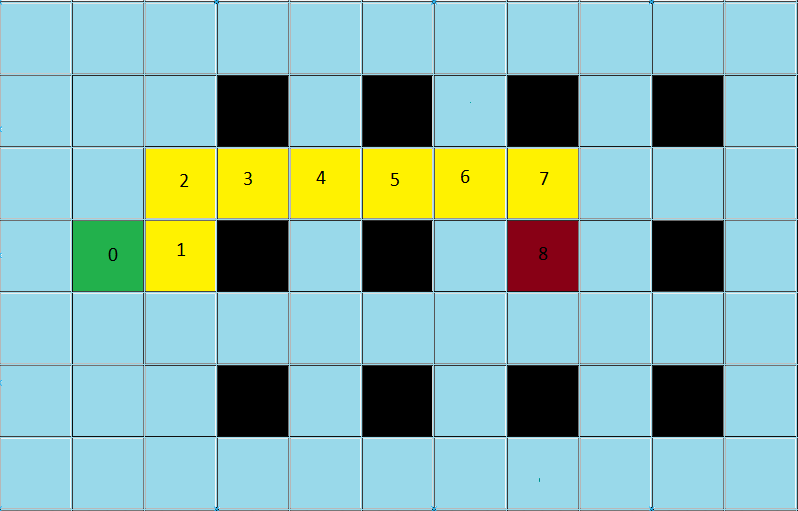


Рис. 6 Реализация алгоритма Дейкстры

Алгоритм A\* является модификацией алгоритма Дейкстры и применяется для поиска кратчайшего пути от начальной точки до конечной. В отличие от алгоритма Дейкстры, который проверяет все возможные пути от начальной точки до конечной, алгоритм A\* фокусируется на проверке пути, который ближе всего к правильному.

Для каждой клетки, в которую может переместиться БПТС, алгоритм A\* вычисляет эвристическое приближение к конечной точке. Это приближение рассчитывается с использованием метода Манхэттенского расстояния, который представляет собой сумму абсолютных разностей между координатами текущей и конечной точек.

Цена перехода и эвристическое приближение складываются, и полученная сумма используется для определения приоритета каждой точки. Точка с наименьшей суммой имеет более высокий приоритет и рассматривается в первую очередь.

Если у двух или более точек получается одинаковое значение суммы, то для каждой из этих точек дополнительно проверяется маршрут дальше для определения наиболее короткого пути.

В обычной реализации алгоритма A\* для поиска пути на местности стоимость перехода обычно устанавливается равной десяти для удобства, что облегчает вычисления.

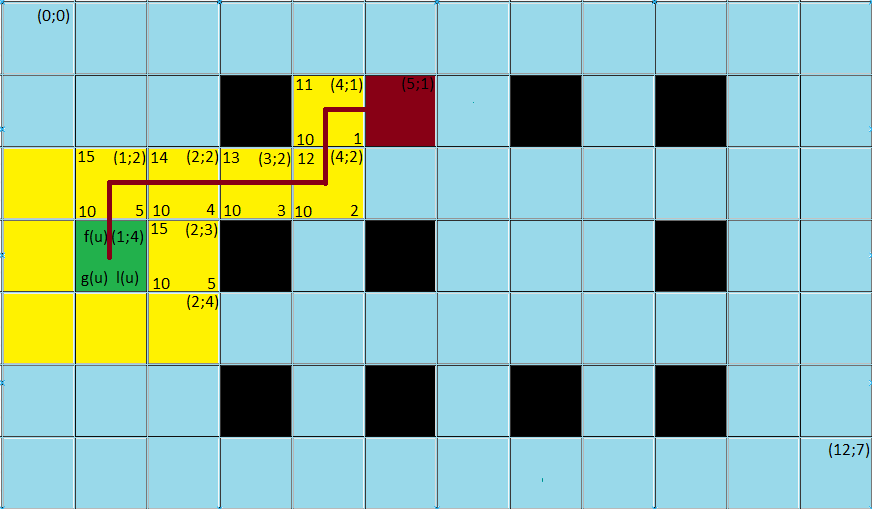


Рис. 7 Графическая реализация алгоритма А\*

**2.3.** **Разработка алгоритма задачи оптимизации перемещения беспилотного такси в условиях города.**

Использование алгоритма A\* в данном случае представляется наиболее целесообразным, поскольку он обладает следующими преимуществами:

* Более быстрое нахождение пути: А\* фокусируется на проверке только тех клеток, которые находятся ближе к конечной точке, что позволяет быстрее найти оптимальный маршрут.
* Экономия памяти: поскольку алгоритм A\* не проверяет все клетки, а только те, которые находятся вблизи конечной точки, он требует меньше памяти для хранения данных.
* Простота в программной реализации: алгоритм A\* относительно прост в реализации и может быть легко встроен в программный код.

Таким образом, выбор алгоритма A\* обоснован его эффективностью, экономией ресурсов и простотой внедрения.

Пример алгоритмической схемы реализации алгоритма A\* представлен на рисунке 8.

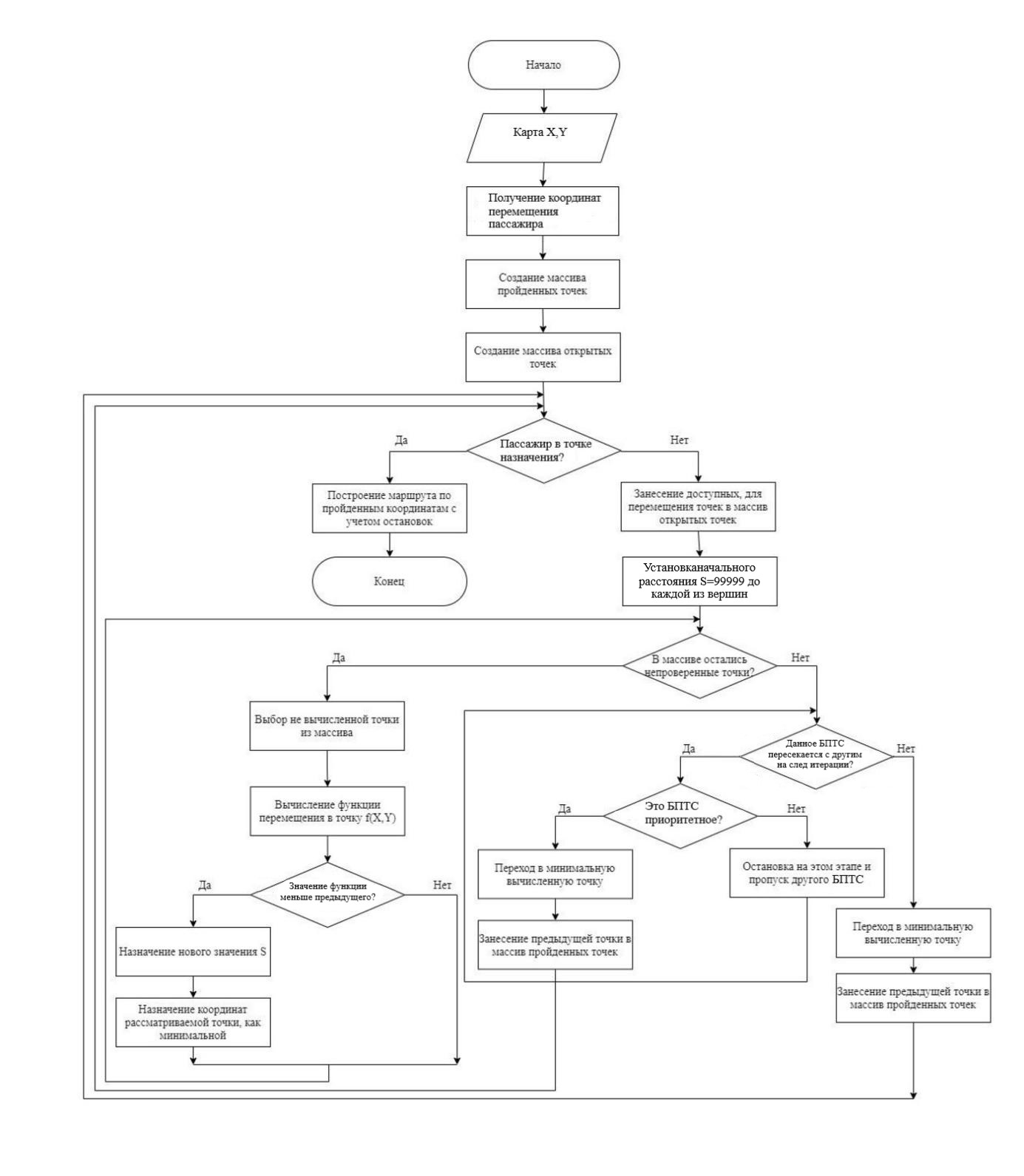


Рис. 8 алгоритмическая схема реализации алгоритма А\*

Входные данные для нашей задачи – карта города.

Первым шагом программа обрабатывает карту города и координаты доставки пассажира.

Этот подход представляет собой эффективный способ использования алгоритма A\* для нахождения оптимального маршрута БПТС. Вот как происходит алгоритмический процесс:

Создание двух массивов координат: доступных точек и уже посещенных точек. В начальной точке БПТС доступно 4 точки перемещения, они заносятся в массив доступных точек.

Устанавливается начальное значение S, которое представляет собой минимальное значение функции fi(Xi, Yi) на этом этапе.

Каждая точка из массива проверяется на перемещение в нее, пока не останется непроверенных координат.

Для каждой точки находится значение f(X, Y), которое вычисляется из постоянной g (равной десяти) и h(u), функции эвристического приближения по методу Манхэтенна.

Значение S сравнивается с минимальным значением f(X, Y), и если оно меньше предыдущего, S изменяется, а координаты данной точки записываются как приоритетные для перемещения дальше.

После проверки всех точек выбирается точка с минимальным значением f(X, Y) для перемещения.

Текущая точка записывается в массив пройденных точек, а в массив доступных заносятся новые точки, доступные для перемещения.

Перед перемещением в следующую точку проверяется условие столкновения двух БПТС в одно время, и приоритетный БПТС проходит первым.

Когда следующая точка около робота равна конечной, строится маршрут по массиву пройденных точек, и алгоритм завершает свою работу.

Этот алгоритм обеспечивает эффективное нахождение оптимального маршрута для БПТС, учитывая его текущее положение и конечную цель.

**Решение задачи оптимизации перемещения беспилотного такси в услоовиях города**

План действий для программы:

* Инициализация поля координат, где каждый квадрат представляет собой определенное положение на поле.
* Определение стартовой точки для каждого БПТС (красные квадраты) и конечных точек для каждого пассажира (зеленые точки).
* Обозначение оранжевых квадратов как мест точек перемещения и синих квадратов как недоступных мест перемещения.
* Реализация алгоритма A\* для каждого БПТС:
  + Начало поиска пути от стартовой точки до ближайшей конечной точки.
  + Определение доступных точек перемещения (серые квадраты) вокруг каждой текущей позиции БПТС.
  + Расчет функции f(X, Y) для каждой доступной точки с использованием эвристического приближения и стоимости перехода.
  + Выбор следующей точки с минимальным значением f(X, Y) для перемещения.
  + Перемещение БПТС в выбранную точку.
  + Проверка столкновений двух БПТС и управление их перемещением, чтобы избежать одновременного нахождения в одной клетке.
  + Повторение этого процесса до достижения конечной точки для каждого пассажира.
* Построение оптимального маршрута для каждого пассажира на основе найденных путей.
* Визуализация результатов на поле координат, показывающая перемещение БПТС и пассажиров.
* Завершение программы после завершения перемещения всех пассажиров к их конечным точкам.

Этот план обеспечит эффективное выполнение задачи оптимизированного перемещения пассажиров с использованием двух БПТС на заданном поле координат.

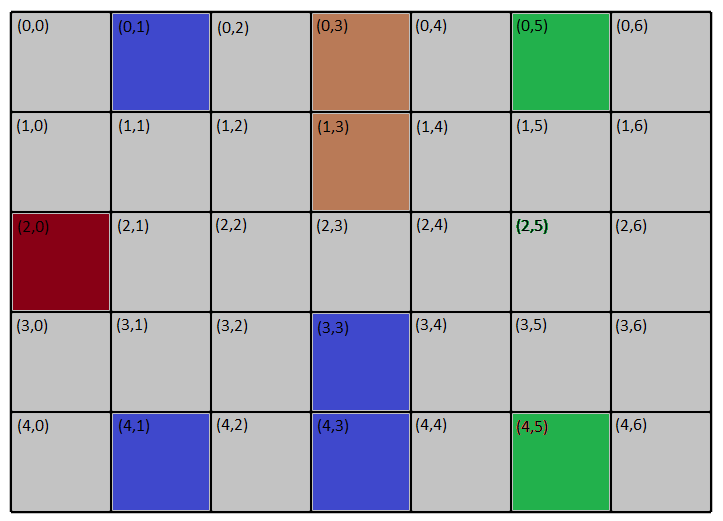


Рис. 9 Координатное поле.

Начинаем с первого шага, где в начальной точке БПТС могут пойти в трех направлениях: вверх, вниз и вправо. Для каждой клетки вычисляем значение f(X,Y) = g + h(X,Y). В нашем случае, так как помещение ровное и нет склонов, g всегда будет равно десяти. Далее для каждой клетки вычисляем h(X,Y). На рисунке 10 будут отображены значения f(X,Y), *g* и h(X,Y) для каждой клетки. Каждое значение будет помечено как "(1)" для первого БПТС и "(2)" для второго.

На втором шаге у нас получается два одинаковых минимальных значения f(X,Y) как для первого, так и для второго БПТС. В данной ситуации целесообразно выбрать перемещение вправо. Однако, оба БПТС не могут начать перемещение одновременно. Так как номер второго БПТС равен двум, что больше номера первого, второй БПТС сначала пропустит первый, а затем начнет перемещение. Клетки, в которых БПТС побывали, будут обозначены желтым цветом.

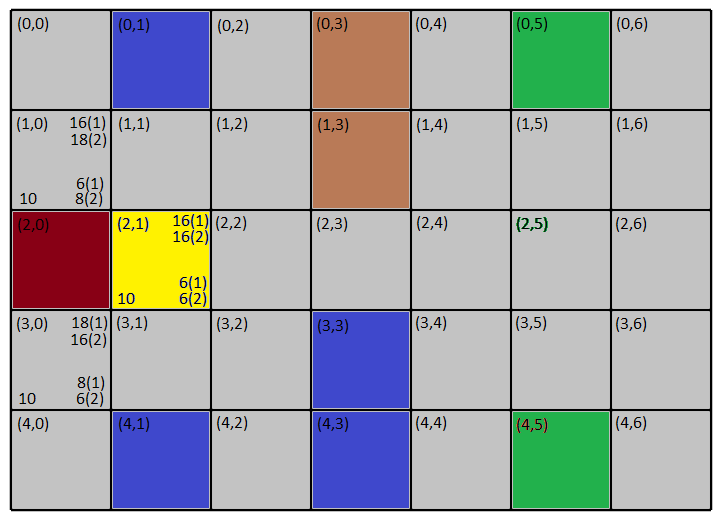


Рис. 10 Нахождение f(X,Y) для каждой доступной клетки

3 шаг: проделываем такую же операцию для новой клетки, как на предыдущем этапе для первого БПТС. Минимальным значением будет 15, в эту клетку БПТС будет перемещаться далее(рисунок 11). Второй БПТС начинает свое движение вправо на первую клетку.

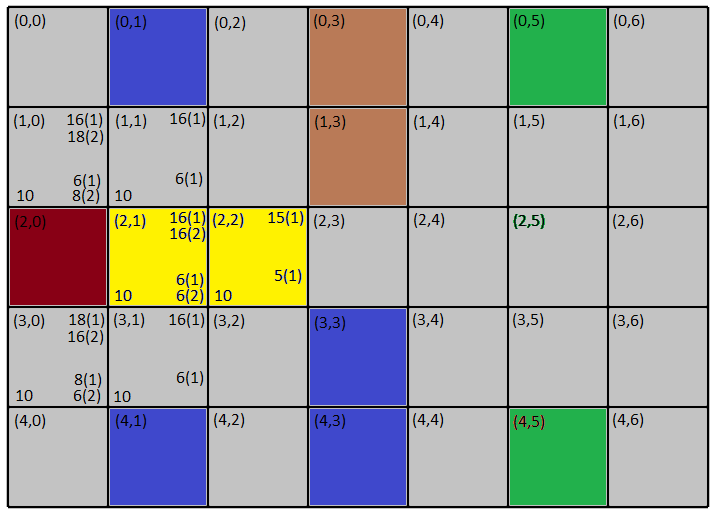


Рис. 11 Вычисления для следующей клетки

4 шаг: у первого БПТС получится два минимальных значения равных 14, выбираем перемещение вправо (рисунок 12). У второго БПТС минимальное значение будет равно 15, перемещение вправо.

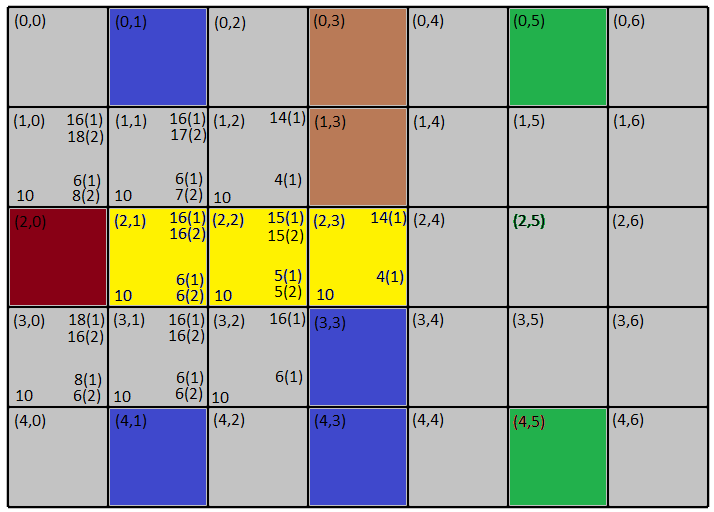


Рис. 12 Маршрут перемещения

5 шаг: первый БПТС будет находится между двумя местами в городе, слева точка, с которой БПТС перешел, поэтому остается путь только вправо (рисунок 13). Для второго БПТС минимальное значение будет два минимальных значения, выбираем перемещение вправо.

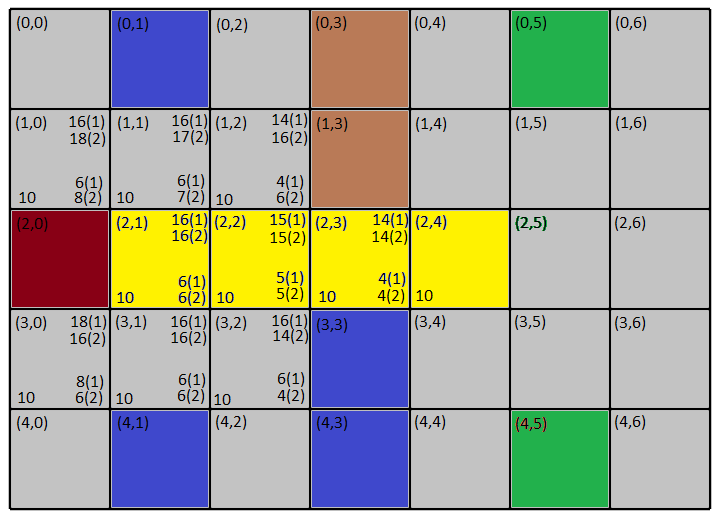


Рис. 13 Следующий шаг маршрута перемещения

Далее повторяем предыдущие шаги и строим полный маршрут до точки назначения первого и второго БПТС (рисунок 14). Красной линией обозначен маршрут первого БПТС, а зеленой – второго.

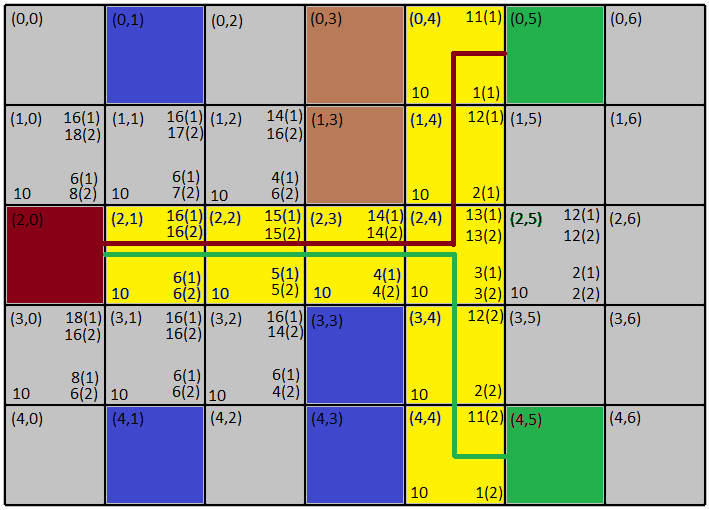


Рис. 14 Итоговые маршруты перемещения

## **Выводы по главе 2.**

В рамках второй главы:

1. Сформулирована содержательная постановка задачи;
2. Сформулирована математическая постановка задачи;
3. Определены условия выполнения решения задачи оптимизации перемещения беспилотного такси в услоовиях города;
4. Выбран алгоритм А\* для решения задачи оптимизации перемещения беспилотного такси в услоовиях города.

**Глава 3. РАЗРАБОТКА ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ ДЛЯ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА**

**3.1.** **Разработка модели потока данных (DFD) и концептуальной модели базы данных.**

Диаграммы потоков данных (Data Flow Diagrams — DFD) представляют собой иерархию функциональных процессов, связанных потоками данных. Цель такого представления — продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

**Модель окружения**

Модель окружения описывает систему как определенный объект, отвечающий на события, которые порождаются внешними сущностями.

Основным процессом является процесс оптимизации перемещения беспилотного транспортного средства в условиях города.

В качестве входных источников данных:

* Пользователь

В качестве выходных источников данных:

* Пользователь
* Оператор поддержки БПТС - это специалист, имеющий квалификацию в области БПТС и ответственный эксплуатацию транспортного средства и его техническое обслуживание;

Модель окружения представлена на рисунке 15.

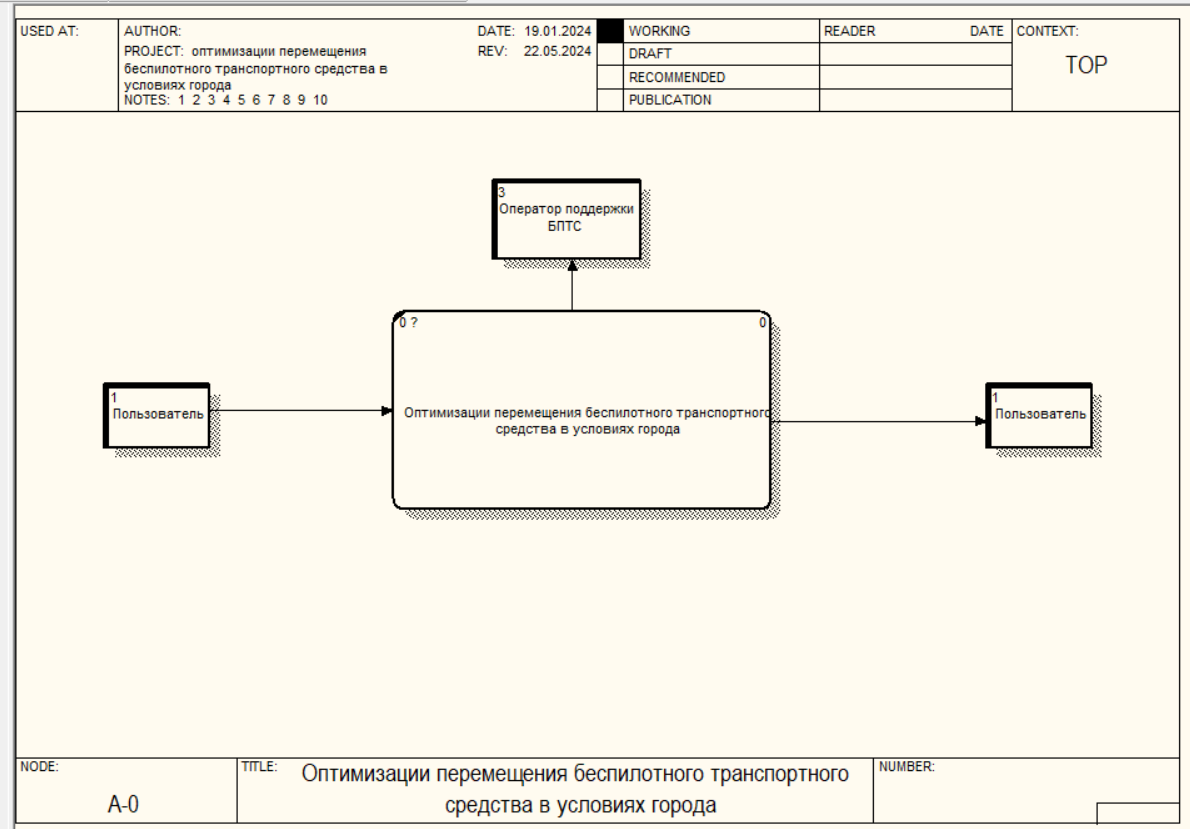


Рис. 15 Модель окружения DFD

**Логическая модель**

Логическая модель представляет систему в виде набора действий и определяет порядок выполнения этих действий. Она описывает систему в терминах функциональных блоков, которые взаимодействуют между собой через потоки данных. Что позволяет представить систему в виде графической диаграммы, где блоки представляют отдельные функции, а потоки данных показывают передачу информации и результатов между блоками.

Основные функции:

«Оптимизации перемещения беспилотного транспортного средства» - на данном этапе происходит вычисление оптимального маршрута перемещения с помощью ЭВМ

Входными потоками данных являются:

* Координаты перемещения БПТС;
* Документы обеспечивающие безопасное перемещение БПТС;

Выходным потоком является:

* Оптимизированная траектория перемещения

«Формирование истории о поездках в базе данных» - на данном этапе происходит заполнение базы данных информацией по пассажире и точках перемещения.

Входными потоками данных являются:

* Информация о поездке;

На рисунке 16 представлена логическая модель.

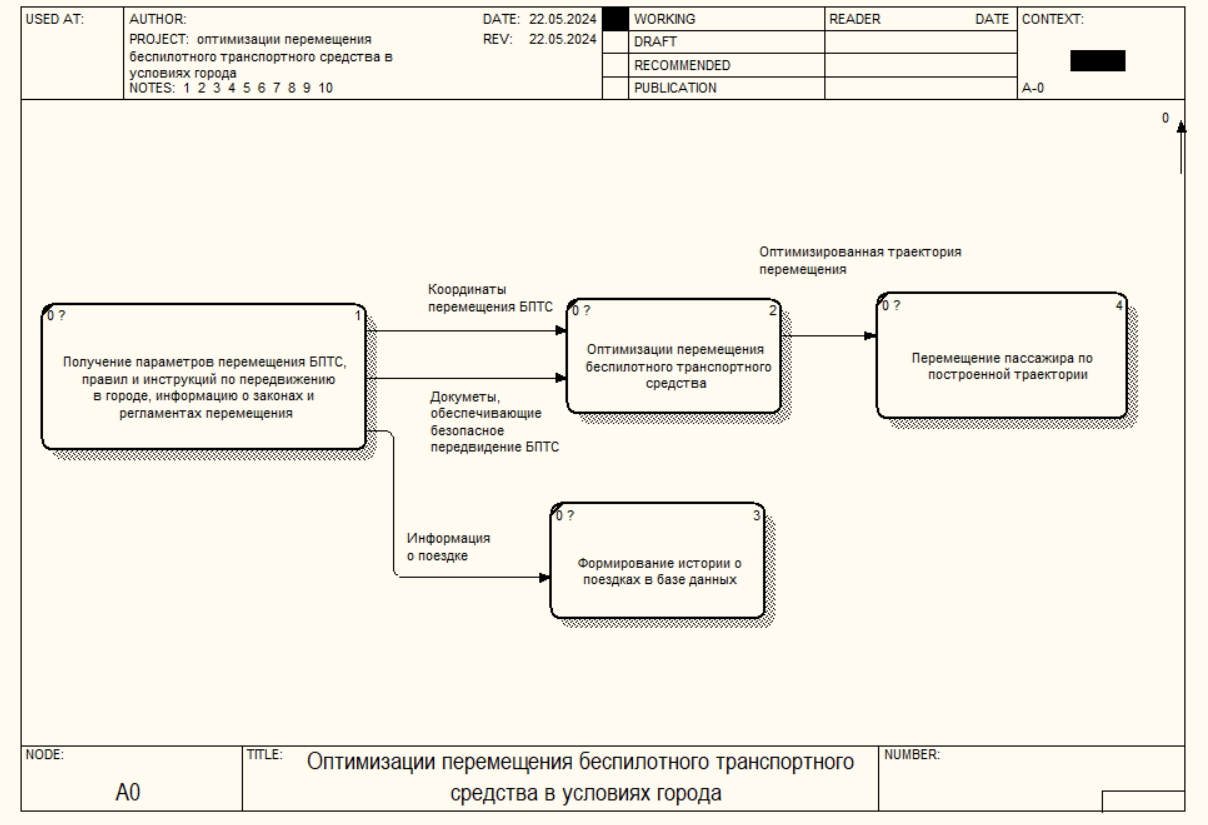


Рис. 16 Диаграмма DFD. Логическая модель

**Модель поведения**

Модель поведения описывает последовательность обработки событий в системе. Она включает в себя внешние сущности, которые определены в модели окружения и выступают в роли источников и/или приемников информации, а также функциональные блоки, которые определены в логической модели и отвечают за обработку этой информации.

Модель поведения представляет собой описание того, как события и данные передаются между внешними сущностями и функциональными блоками, позволяя показать порядок их взаимодействия и обработки.

Взаимосвязь между блоками отражает передачу и прием информации. Чтобы учесть процессы сохранения данных, применяются хранилища данных. В результате, модель поведения представляет собой комплексную схему, которая дополняет и расширяет логическую модель и модель окружения.

Хранилище данных — это абстрактное устройство или способ хранения информации, которая перемещается между различными процессами.

Внешние сущности: команда разработки, пользователь.

Хранилища данных, которые служат входными данными для блока процесса “Преобразование данных в координатную плоскость”:

* Начальная и конечная точка – содержит всю информацию о месте отправления и месте прибытия;

Хранилища данных, которые служат выходными данными для блока процесса “Преобразование данных в координатную плоскость”

* Координаты перемещения БПТС – содержит всю информацию о месте отправления и месте прибытия в понятной для программы форме.

Хранилища данных, которые служат входными данными для блока процесса “ расчет оптимизированной траектории движения”:

* Карта города – содержит всю информацию о дорогах города и точках обеспечения БПТС;
* Нормативные документы обеспечивающие безопасность перемещения БПТС – данные документы включают в себя правила дорожного движения, методические рекомендации и инструкции по перемещению в городе и взаимодействия с человеком, технические регламенты безопасности, федеральные законы;
* Координаты перемещения БПТС – содержит всю информацию о месте отправления и месте прибытия в понятной для программы форме.

Хранилища данных, которые служат выходными данными для блока процесса “ расчет оптимизированной траектории движения”:

* Оптимизированная траектория перемещения БПТС – оптимизированный маршрут перемещения беспилотного транспортного средства.

На рисунке 17 представлена модель поведения.

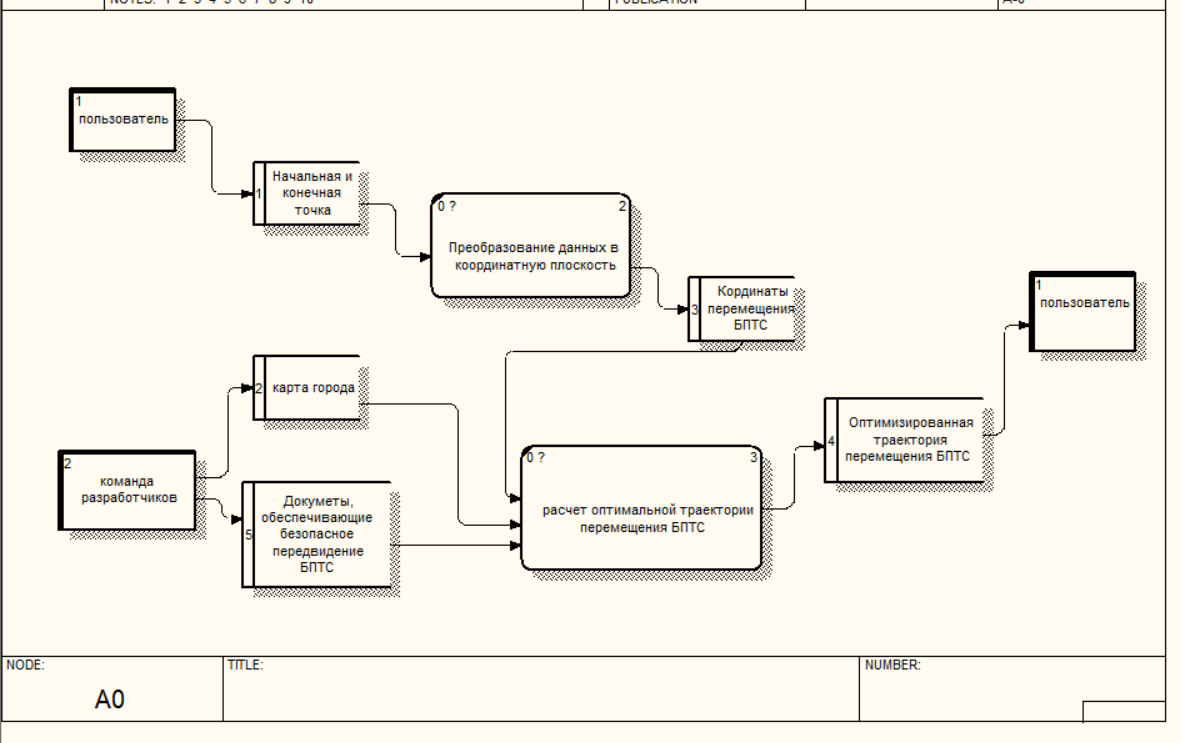


Рис. 17 Диаграмма DFD. Модель поведения.

**Концептуальное проектирование базы данных**

Рассмотрим по пунктам этапы концептуального проектирования базы данных:

1. В таблице 2 продемонстрировано установление сущностей.

Таблица 2. Таблица сущностей.

|  |  |  |
| --- | --- | --- |
| Обозначение сущности (связи) | Наименование сущности (связи) | Количество экземпляров |
| S1 | Пассажир | 1 |
| S2 | Оператор | 1 |
| S3 | Поездка | 1 |

Сущность «Пассажир» содержит сведения о пассажирах, которые пользуются приложением.

1. Присваивание сущностям названия, уникальный идентификатор приведено в таблице 3.

Таблица 3. Таблица определения атрибутов и доменов.

|  |  |  |
| --- | --- | --- |
| Наименование сущности (связи) | Наименование атрибутов | Наименование домена |
| Пассажир | UID | Код |
| Пол | Текст |
| Возраст | Число |
| Номер телефона | Номер |
| Рейтинг | Число |
| Почта | Текст |
| ФИО | Текст |

Таблица 3. (Продолжение). Таблица определения атрибутов и доменов.

|  |  |  |
| --- | --- | --- |
| Поездка | ID | Текст |
| Адрес доставки | Текст |
| Адрес начальной точки | Текст |
| Пассажир | Текст |
| Оператор | Текст |
| Место доставки | Индекс места | Текст |
| Адрес | Текст |
| Координаты по X | Число |
| Координаты по Y | Число |
| Местоположение пассажира | Индекс места | Текст |
| Адрес | Текст |
| Координаты по X | Число |
| Координаты по Y | Число |
| Оператор | UID | Код |
| ФИО | Текст |

1. В таблице 4 демонстрируется определение типа данных для каждого атрибута.

Таблица 4. Таблица наименования, названия и примеры доменов.

|  |  |  |
| --- | --- | --- |
| Наименование домена | Название и описание типов данных | Примеры значений |
| Код | Набор символов (1000) | Hvyi34, 43f45f, 5t4465 |
| Число | Целочисленный тип данных (1000) | 3,4,9 |

Таблица 4. (Продолжение). Таблица наименования, названия и примеры доменов.

|  |  |  |
| --- | --- | --- |
| Логическое значение | Логический тип данных (1) | 1, 0 |
| Текст | Текстовый тип данных (9) | 94229003ABC |
| Название | Текстовый тип данных (2) | A1 |
| Номер | Целочисленный тип данных (1000) | 123,23432,243 |

1. Определение для каждой сущности первичного и основного ключей:

Для сущности «Пассажир» ключом является уникальный код UID.

Для сущности «Оператор» ключом является уникальный код UID.

1. Между сущностями имеются связи:

Cсущность «Адрес» связана с сущностью «Пассажир».

1. Построение графической концептуальной модели представлено на рисунке 18.

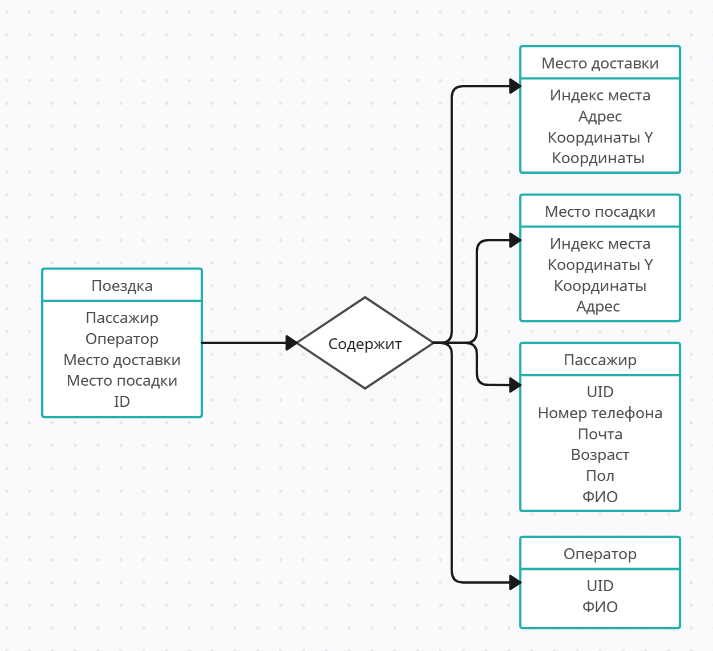


Рис. 18 Графический вид концептуальной модели

## **3.2. Разработка логической модели базы данных**

База данных простроена с использованием паттерна обезличиванием данных

Проектирование базы данных с обезличиванием данных необходимо для соблюдения требований законодательства и обеспечения безопасности персональных данных пользователей. Обезличивание позволяет скрыть информацию о конкретных лицах или организациях, делая её менее уязвимой для кражи или неправомерного использования.

Основные цели проектирования базы данных с обезличиванием данных:

* Соблюдение требований законодательства, таких как ФЗ-152 «О персональных данных» и GDPR (Общий регламент по защите данных).
* Обеспечение безопасности персональных данных пользователей, предотвращение утечек и компрометации информации.
* Возможность использования обезличенных данных для анализа и исследований без нарушения приватности пользователей.

Защита информации обеспечивается методом выдачи прав доступа к таблицам. То есть мы запрещаем читать конфиденциальную таблицу обычным пользователям базы данных

В таблице 5 приведена база данных в третьей нормальной форме.

Таблица 5. Логическое проектирование.

|  |  |  |
| --- | --- | --- |
| **Наименование отношения** | **Наименование атрибута** | **Функциональная зависимость**  F1 |
| Пассажир | UID | \* |
| Пол |
| Возраст |
| Номер телефона |
| Рейтинг |
| Почта |
| ФИО |
| Место доставки | Индекс места | \* |
| Адрес |
| Координаты по X |
| Координаты по Y |
| Местоположение пассажира | Индекс места | \* |
| Адрес |
| Координаты по X |
| Координаты по Y |

Таблица 5. (Продолжение). Логическое проектирование.

|  |  |  |
| --- | --- | --- |
| Оператор | UID | \* |
| ФИО |
| Поездка | Адрес доставки | \*  \*  \*  \* |
| Адрес начальной точки |
| Пассажир |
| Оператор |
| ID |

Диаграмма логического проектирования представлена на рисунке 19.

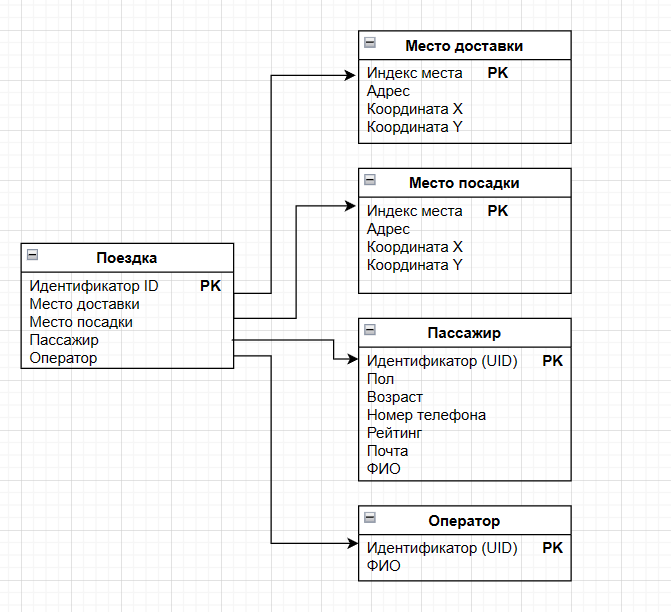


Рис. 19 Диаграмма логического проектирования.

## **3.3 Физическое проектирование базы данных**

При создании базы данных использовалась СУБД ClickHouse, с помощью открытого инструмента DBeaver производилась настройка базы данных.

ClickHouse - это колоночная аналитическая база данных, разработанная для обработки больших объемов данных с высокой производительностью. Основное преимущество ClickHouse заключается в его способности эффективно анализировать и обрабатывать большие массивы данных в режиме реального времени.

Вот несколько ключевых особенностей ClickHouse:

1. Колоночное хранение данных: Данные в ClickHouse хранятся колоночно, что позволяет выполнять агрегации и аналитические запросы быстрее, чем в реляционных базах данных с рядовым хранением данных.
2. Масштабируемость: ClickHouse обладает горизонтальной масштабируемостью, что позволяет легко добавлять новые узлы и распределять данные между ними для обеспечения высокой производительности при работе с большими объемами данных.
3. Высокая производительность: ClickHouse оптимизирован для быстрой обработки аналитических запросов, включая агрегации, фильтрацию и сортировку данных. Это делает его отличным выбором для ситуаций, требующих выполнения сложных аналитических запросов над большими объемами данных.
4. Поддержка SQL: ClickHouse поддерживает SQL-синтаксис, что делает его легким для изучения и использования для разработчиков и аналитиков, уже знакомых с SQL.
5. Работа в реальном времени: ClickHouse способен обрабатывать данные в режиме реального времени, что позволяет быстро анализировать потоковую информацию и принимать оперативные решения на основе полученных данных.
6. Отказоустойчивость: ClickHouse обеспечивает высокую отказоустойчивость благодаря репликации данных и механизмам восстановления после сбоев, что делает его надежным выбором для критически важных приложений.

В целом, ClickHouse является мощным инструментом для работы с большими объемами данных, предоставляя высокую производительность, масштабируемость и отказоустойчивость для аналитических и реально-временных приложений.

Список команд взаимодействия с таблицей показана на рисунке 20.

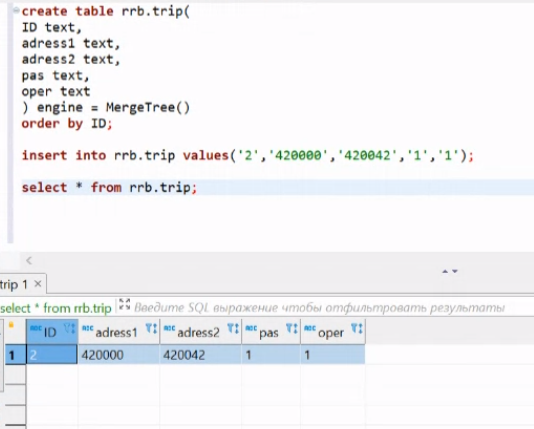


Рис. 20. Список команд для сущности «Поездка»

Таблица trip, которая соответствует сущности «Поездка» показана на рисунке 21.

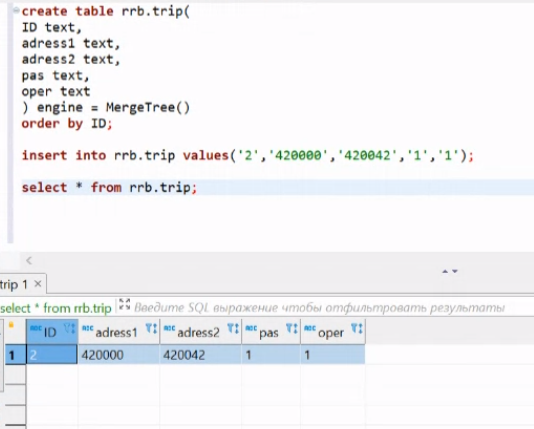


Рис. 21. Таблица сущности «Поездка»

Список команд взаимодействия с таблицей «Место доставки» показана на рисунке 22.

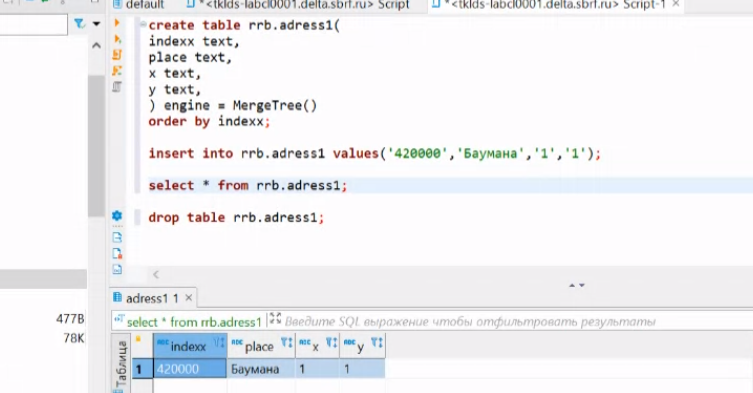


Рис. 22. Список команд для сущности «Место доставки»

Таблица adress1, которая соответствует сущности «Место доставки» показана на рисунке 23.

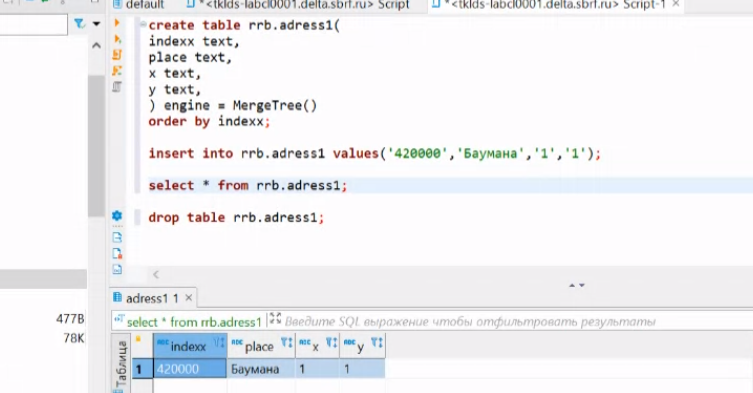


Рис. 23. Таблица сущности «Место доставки»

Список команд взаимодействия с таблицей «Место посадки» показана на рисунке 24.

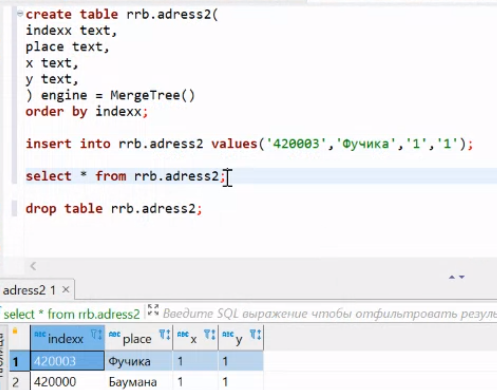


Рис. 24. Список команд для сущности «Место посадки»

Таблица adress2, которая соответствует сущности «Место посадки» показана на рис. 25.

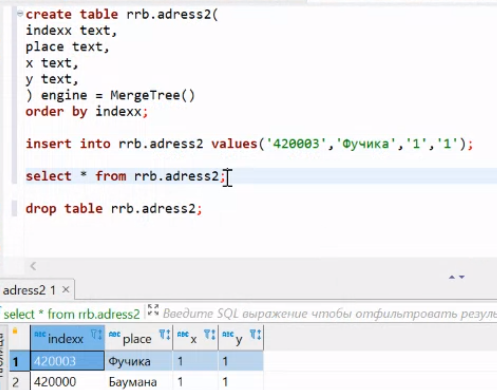


Рис. 25. Таблица сущности «Место посадки»

Список команд взаимодействия с таблицей «Пользователь» показана на рисунке 26.

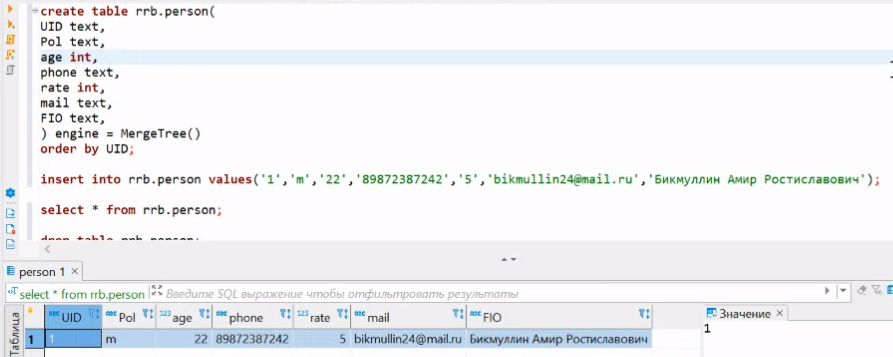


Рис. 26. Список команд для сущности «Пользователь»

Таблица person Пользователь, которая соответствует сущности «Пользователь» показана на рисунке 27.

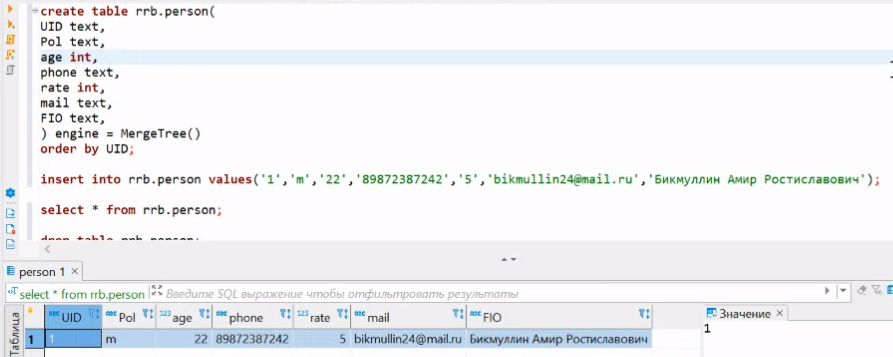


Рис. 27. Таблица сущности «Пользователь»

Список команд взаимодействия с таблицей «Оператор» показана на рисунке 28.

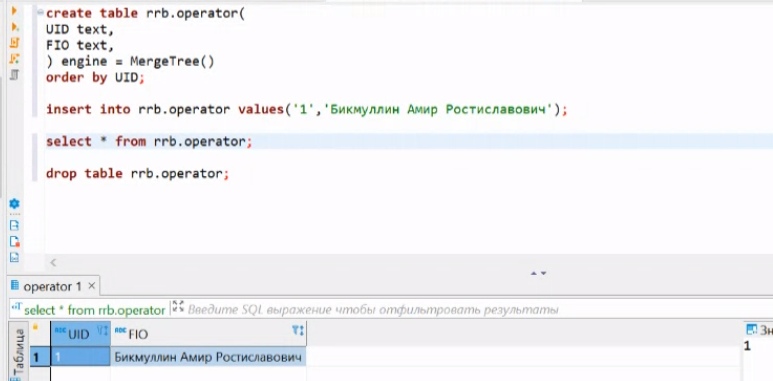


Рис. 28. Список команд для сущности «Оператор»

Таблица operator Пользователь, которая соответствует сущности «Пользователь» показана на рисунке 29.

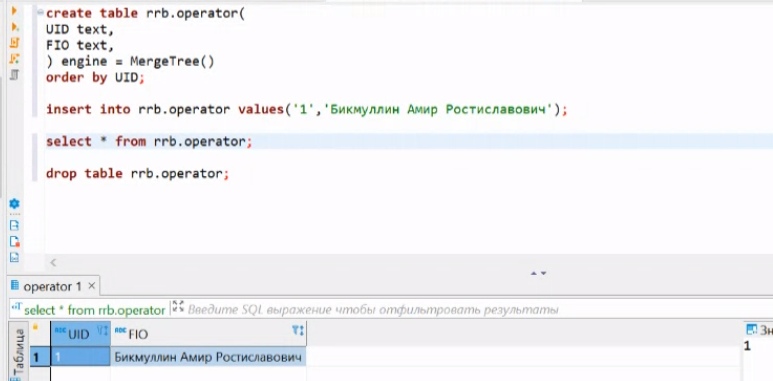


Рис. 29. Таблица сущности «Оператор»

## **Выводы по главе 3.**

На этапе концептуального проектирования были определены сущности и их атрибуты, основные ключи для каждой сущности, а также связи между ними. Затем была создана модель потока данных (DFD), графическая и концептуальная модель базы данных. После этого была построена логическая модель базы данных.

**Глава 4. РАЗРАБОТКА ПРОГРАМНОГО ОБЕСПЕЧЕНИЯ ПРОЦЕДУРЫ ОПТИМИЗАЦИИ ПЕРЕМЕЩЕНИЯ БЕСПИЛОТНОГО ТАКСИ В УСЛОВИЯХ ГОРОДА**

**4.1. Архитектура программы для автоматизации процедуры оптимизации перемещения беспилотного такси в условиях города.**

Приложение разработано в среде разработки Visual Studio Community 2022 на языке C#.

На рисунке 30 представлены диаграммы классов приложения.

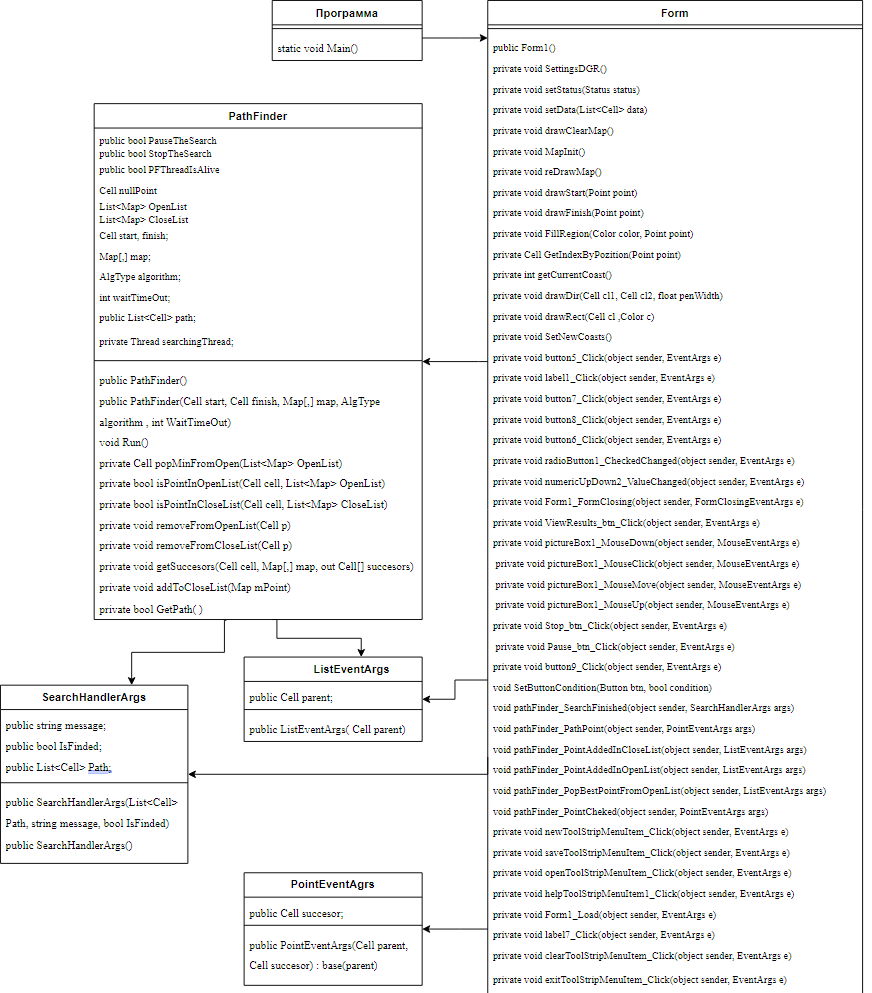


Рис. 30 Диаграмма классов

Описание классов, их свойств и методов приведены в таблице 6.

Таблица 6. Описание классов, их свойств и методов

|  |  |  |
| --- | --- | --- |
| Название классов | Имя атрибута и операции | Описание |
| class ListEventArgs : EventArgs |  | Класс предназначен для одного метода ListEventArgs |
|  | public Cell parent; | Переменная предыдущей клетки |
|  | public ListEventArgs( Cell parent) | Метод получающий предыдущую клетку |
| class PointEventArgs : ListEventArgs |  | Класс предназначен для одного метода PointEventArgs |
|  | public Cell succesor; | Переменная следующей клетки |
|  | public PointEventArgs(Cell parent, Cell succesor) : base(parent) | Метод получающий следующую клетку |
| class SearchHandlerArgs : EventArgs |  | Класс устанавливает информацию в клетки |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | public SearchHandlerArgs(List<Cell> Path, string message, bool IsFinded) | Конструктор устанавливающий в клетку информацию |
|  | public SearchHandlerArgs() | Пустой конструктор |
|  | public string message; | Переменная содержит вспомогательную информацию |
|  | public bool IsFinded; | Переменная содержит информацию,найден ли был путь |
|  | public List<Cell> Path; | Переменная содержит путь |
| class PathFinder |  | Класс PathFinder предназначен для поиска пути (маршрута) между двумя точками на карте, используя различные алгоритмы, такие как алгоритм Дейкстры, волновой алгоритм и алгоритм A\* |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | public bool PauseTheSearch { set; get; } | Булевые свойство, которые управляют выполнением потока поиска пути |
|  | public bool StopTheSearch { set; get; } | Булевое свойство, которые управляют выполнением потока поиска пути |
|  | public bool PFThreadIsAlive | Свойство только для чтения, которое возвращает значение, указывающее, активен ли поток поиска пути |
|  | Cell nullPoint; | Объект типа Cell, представляющий пустую ячейку. |
|  | List<Map> OpenList ; | Список, представляющий открытую точку, используемую в процессе поиска пути |
|  | List<Map> CloseList ; | Список, представляющий закрытую точку, используемую в процессе поиска пути |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | Cell start, finish; | Начальная и конечная точки пути. |
|  | Map[,] map; | Двумерный массив Map, представляющий карту, на которой ищется путь. |
|  | AlgType algorithm; | Тип используемого алгоритма поиска пути. |
|  | int waitTimeOut; | Задержка при расчетах. |
|  | public List<Cell> path; | Список ячеек, представляющих найденный путь |
|  | private Thread searchingThread; | Поток, используемый для выполнения поиска пути. |
|  | public PathFinder() | Пустой конструктор. |
|  | public PathFinder(Cell start, Cell finish, Map[,] map, AlgType algorithm , int WaitTimeOut) | Конструктор, который принимает начальную и конечную точки, карту, тип алгоритма и время ожидания. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | void Run() | Метод получающий путь |
|  | private Cell popMinFromOpen(List<Map> OpenList) | Этот метод извлекает точку с наименьшей общей стоимостью прохода из открытого списка точек. Он принимает список OpenList, содержащий объекты типа Map, которые представляют точки на карте. Метод проходит по списку и возвращает точку с наименьшим значением fValue, которое указывает на оценку стоимости прохода от текущей точки до конечной точки. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private bool isPointInOpenList(Cell cell, List<Map> OpenList) | Этот метод проверяет, содержится ли определенная точка в открытом списке точек. Он принимает точку cell и список OpenList и возвращает true, если точка содержится в списке, и false в противном случае. |
|  | private bool isPointInCloseList(Cell cell, List<Map> CloseList) | Этот метод аналогичен предыдущему, но проверяет наличие точки в закрытом списке точек CloseList. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void removeFromOpenList(Cell p) | Этот метод удаляет точку p из открытого списка точек. Он проходит по списку OpenList и удаляет объекты, которые содержат точку p. Это позволяет поддерживать актуальность списка точек в процессе поиска пути. |
|  | private void removeFromCloseList(Cell p) | Этот метод аналогичен предыдущему, но для закрытого списка точек CloseList. Он удаляет точку p из списка закрытых точек, чтобы избежать повторной обработки той же точки в дальнейшем. |
|  | private void getSuccesors(Cell cell, Map[,] map, out Cell[] succesors) | Метод для получения преемников (соседей) ячейки. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void addToOpenList(Map mPoint) | Метод для добавления точек в открытый список соответственно. |
|  | private void addToCloseList(Map mPoint) | Метод для добавления точек в закрытый список соответственно. |
|  | private bool GetPath( ) | Метод для поиска пути по заданному алгоритму. |
| public partial class Help : Form |  | Класс для заполнения окна Help |
|  | string text; | Текст для вывода в форме |
|  | public Help() | Конструктор класса |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void Help\_FormClosing(object sender, FormClosingEventArgs e) | Этот метод выполняется при закрытии формы. Он проверяет причину закрытия формы и, если она вызвана пользователем (CloseReason.UserClosing), отменяет закрытие и скрывает форму. |
|  | private void Close\_btn\_Click(object sender, EventArgs e) | Этот метод выполняется при нажатии на кнопку Close\_btn. Он скрывает текущую форму. |
|  | private void Help\_Load(object sender, EventArgs e) | Вспомогательный метод для обработки загрузки формы |
|  | private void label2\_Click(object sender, EventArgs e) | Вспомогательный метод для обработки щелчка на label2 |
| static class Program |  | Главный класс запускающий программу |
|  | static void Main() | Метод вызывает формы |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
| public partial class NewMapSettings : Form |  | Вспомогательный класс для сощдания новой карты |
|  | public NewMapSettings() | Объявление класса |
|  | private void NewMapSettings\_Load(object sender, EventArgs e) | Конструктор класса |
| public partial class Form1 : Form |  | Класс представляет весть функционал, который связан с визуальной частью программы |
|  | public Form1() | Конструктор класса |
|  | private void SettingsDGR() | Метод который настраивает элементы карты |
|  | private void setStatus(Status status) | Метод устанавливает начальное состояние статуса поиска |
|  | private void setData(List<Cell> data) | Метод setData предназначен для установки данных в элемент управления Results\_dgv |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void drawClearMap() | Метод для очистки изображения карты, а затем проходит по всем ячейкам карты и определяет их цвета в зависимости от их типа. |
|  | private void MapInit() | Метод создает карту |
|  | private void reDrawMap() | Этот метод перерисовывает карту на основе информации, хранящейся в массиве map |
|  | private void drawStart(Point point) | Этот метод отвечает за отображение начальной точки на карте. Он заполняет ячейку, соответствующую начальной точке, зеленым кругом. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void drawFinish(Point point) | Этот метод отображает конечную точку на карте. Он заполняет ячейку, соответствующую конечной точке, красным кругом |
|  | private void FillRegion(Color color, Point point) | Этот метод заливает область (ячейку) определенным цветом. Он используется для изменения цвета ячейки при клике на нее мышью. |
|  | private Cell GetIndexByPozition(Point point) | Этот метод возвращает индекс ячейки карты по координатам точки, переданной в качестве аргумента. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private int getCurrentCoast() | Этот метод возвращает текущую стоимость прохода по ячейке в зависимости от выбранного цвета в палитре. |
|  | private void drawDir(Cell cl1, Cell cl2, float penWidth) | Этот метод отрисовывает направление на карте от одной ячейки к другой. Он используется для визуализации пути, найденного алгоритмом поиска |
|  | private void drawRect(Cell cl ,Color c) | Этот метод отрисовывает прямоугольник определенного цвета в указанной ячейке на карте. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void SetNewCoasts() | Этот метод применяет изменения стоимости прохода по клеткам, которые были изменены пользователем. Если пользователь изменил стоимость прохода по каким-либо типам территории, этот метод обновляет соответствующие значения в массиве карты. |
|  | private void button5\_Click(object sender, EventArgs e) | Обработчик кнопки5 |
|  | private void label1\_Click(object sender, EventArgs e) | Обработчик кнопки1 |
|  | private void button7\_Click(object sender, EventArgs e) | Обработчик кнопки7 |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void button8\_Click(object sender, EventArgs e) | Обработчик кнопки8 |
|  | private void button6\_Click(object sender, EventArgs e) | Обработчик кнопки6 |
|  | private void radioButton1\_CheckedChanged(object sender, EventArgs e) | Обработчик изменения кнопки1 |
|  | private void numericUpDown2\_ValueChanged(object sender, EventArgs e) | Этот обработчик событий вызывается при изменении значения в NumericUpDown, предназначенном для указания стоимости прохода по определенному типу территории. |
|  | private void Form1\_FormClosing(object sender, FormClosingEventArgs e) | Этот обработчик событий вызывается при попытке закрытия формы. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void ViewResults\_btn\_Click(object sender, EventArgs e) | Этот обработчик событий вызывается при нажатии на кнопку "View Results". |
|  | private void pictureBox1\_MouseDown(object sender, MouseEventArgs e) | Обработчик события "MouseDown" вызывается при нажатии кнопки мыши на изображении. |
|  | private void pictureBox1\_MouseClick(object sender, MouseEventArgs e) | Обработчик события "MouseClick" вызывается при клике мышью на изображении. |
|  | private void pictureBox1\_MouseMove(object sender, MouseEventArgs e) | Обработчик события "MouseMove" вызывается при перемещении мыши по изображению |
|  | private void pictureBox1\_MouseUp(object sender, MouseEventArgs e) | Обработчик события "MouseMove" вызывается при скроле мыши на изображении |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void Stop\_btn\_Click(object sender, EventArgs e) | Обработчик события нажатия на кнопку "Stop". |
|  | private void Pause\_btn\_Click(object sender, EventArgs e) | Обработчик события нажатия на кнопку "Pause" |
|  | private void button9\_Click(object sender, EventArgs e) | Обработчик события нажатия на кнопку "Run". |
|  | void SetButtonCondition(Button btn, bool condition) | Метод используется для установки состояния (включено/отключено) для кнопок |
|  | void pathFinder\_SearchFinished(object sender, SearchHandlerArgs args) | Обработчик события окончания поиска пути. |
|  | void pathFinder\_PathPoint(object sender, PointEventArgs args) | Обработчик события обработки точки пути. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | void pathFinder\_PointAddedInCloseList(object sender, ListEventArgs args) | Обработчик события добавления пути в открытый список |
|  | void pathFinder\_PointAddedInOpenList(object sender, ListEventArgs args) | Обработчик события добавления пути в закрытый список |
|  | void pathFinder\_PopBestPointFromOpenList(object sender, ListEventArgs args) | Обработчик события выбора лучшего пути |
|  | void pathFinder\_PointCheked(object sender, PointEventArgs args) | Этот метод вызывается в ответ на событие, связанное с проверкой определенной точки в процессе поиска пути. |
|  | private void newToolStripMenuItem\_Click(object sender, EventArgs e) | Этот обработчик запускается при выборе опции "New" из меню. |

Таблица 6. (Продолжение). Описание классов, их свойств и методов.

|  |  |  |
| --- | --- | --- |
|  | private void saveToolStripMenuItem\_Click(object sender, EventArgs e) | Этот обработчик запускается при выборе опции "Save" из меню. |
|  | private void openToolStripMenuItem\_Click(object sender, EventArgs e) | Этот обработчик запускается при выборе опции "Open" из меню. |
|  | private void helpToolStripMenuItem1\_Click(object sender, EventArgs e) | Этот метод вызывается при нажатии на пункт меню "Help" |
|  | private void Form1\_Load(object sender, EventArgs e) | Этот метод вызывается при загрузке формы. |
|  | private void label7\_Click(object sender, EventArgs e) | тот метод вызывается при щелчке на метке с именем label7 |
|  | private void clearToolStripMenuItem\_Click(object sender, EventArgs e) | Этот метод вызывается при выборе пункта меню "Clear". |
|  | private void exitToolStripMenuItem\_Click(object sender, EventArgs e) | Этот метод вызывается при выборе пункта меню "Exit". |

## **4.2. Разработка пользовательского интерфейса программы**

## **Анализ функциональных требований к программе**

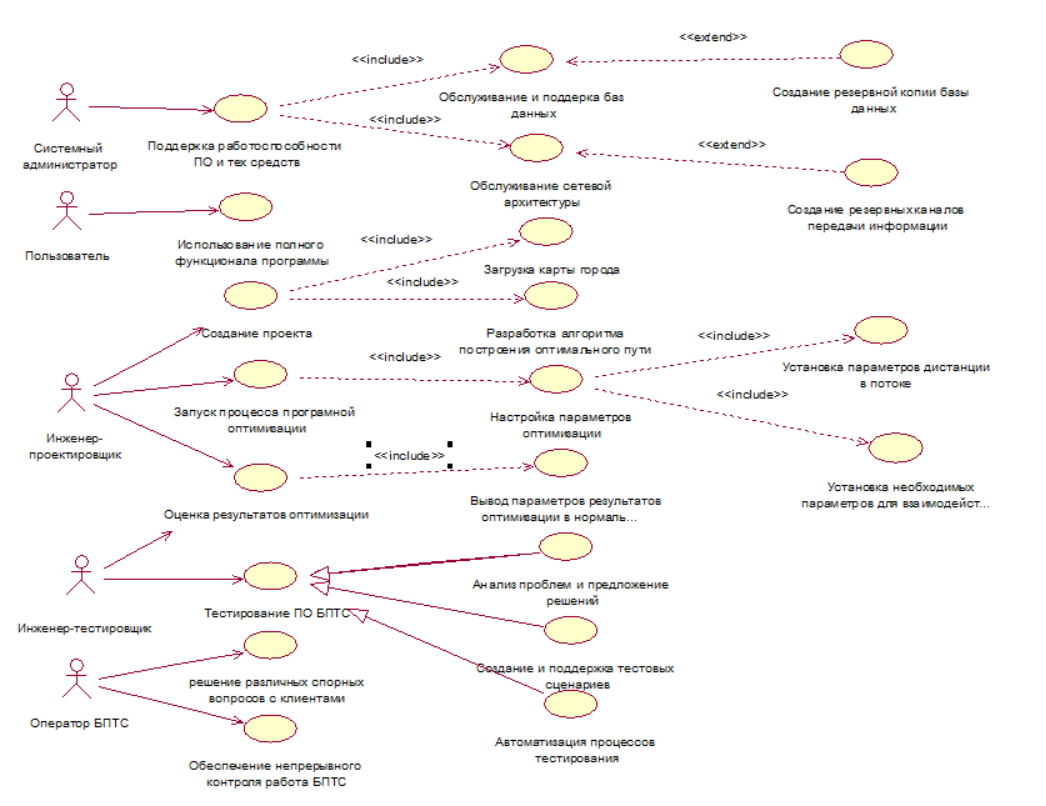
На рисунке 31 представлена диаграмма вариантов использования.

Рис.31 Диаграмма вариантов использования.

Таблица 7. Актеры

|  |  |
| --- | --- |
| Актер | Краткое описание |
| Системный администратор | Занимается поддержанием работоспособности программных и технических средств, баз данных. |
| Инженер-проектировщик | Занимается разработкой и созданием БПТС, производит анализ траектории БПТС и разрабатывает соответствующие меры по оптимизации маршрута движения. |

Таблица 7. (Продолжение). Актеры.

|  |  |
| --- | --- |
| Инженер-тестировщик | Занимается проверкой выполнения требований надежности по методикам структурного и функционального тестирования ПО БПТС. |
| Пользователь | Целевой пользователь программы занимается использованием всего функционала программы |
| Оператор | Обеспечивает поддержку пользователей, решение  различных спорных вопросов с клиентами, обеспечивает  непрерывный контроль за БПТС. |

Сценарий №1 «Создание заказа»

Вариант использования: внесение в базу данных поступивших заказов.

Актер: пользователь, программное изделие.

Цель: создание нового заказа такси для себя в программном изделии.

Краткое описание: пользователь нажимает на пункт «Вызвать такси». Программное изделие открывает диалоговое окно для того, чтобы пользователь указал детали заказа. Далее всплывает окно для установки координат начальной и конечной точки. Программное изделие проверяет корректность введенных данных, создаёт заказ и закрывает диалоговое окно. Тип: Базовый

Таблица 8. Сценарий №1 Создание заказа для себя

|  |  |
| --- | --- |
| Инженер-тестировщик | Программная система |
| 1) пользователь вводит команду для открытия окна и запуска программы. | 2)Появляется главное окно программы. |
| 3) пользователь нажимает на пункт “Вызвать такси” из всплывающего списка на кнопку “вызвать себе”. | 4)Открывается окно для ввода данных. |
| 5) пользователь осуществляет ввод исходных данных маршрута. (Вводит информацию в текстовые поля) и нажимает кнопку “Добавить” | 6)Программа проводит проверку введенной информации. |
| Исключение №1 «Неверно указаны исходные данные» | |
|  | 7)Закрытие диалогового окна. |

Таблица 9. Исключения №1

|  |  |
| --- | --- |
| Исключение №1 «Неверный формат входных данных» | 1)Вывод сообщение об ошибке.  2)Отображение сообщения о готовности к дальнейшей работе. |

Сценарий №2 «Создание заказа»

Вариант использования: внесение в базу данных поступивших заказов.

Актер: пользователь, программное изделие.

Цель: создание нового заказа такси для другого человека в программном изделии.

Краткое описание: пользователь нажимает на пункт «Вызвать такси». Программное изделие открывает диалоговое окно для того, чтобы пользователь указал тип заказа. Далее всплывает окно для установки координат начальной и конечной точки. Программное изделие проверяет корректность введенных данных, создаёт заказ и закрывает диалоговое окно. Тип: Базовый

Таблица 10. Сценарий №2 Создание заказа для другого человека

|  |  |
| --- | --- |
| Инженер-тестировщик | Программная система |
| 1) пользователь вводит команду для открытия окна и запуска программы. | 2)Появляется главное окно программы. |
| 3) пользователь нажимает на пункт “Вызвать такси” из всплывающего списка на кнопку “вызвать для другого человека”. | 4)Открывается окно для ввода данных. |
| 5) пользователь осуществляет ввод исходных данных маршрута. (Вводит информацию в текстовые поля) и нажимает кнопку “Добавить” | 6)Программа проводит проверку введенной информации. |
| Исключение №1 «Неверно указаны исходные данные» | |
|  | 7)Закрытие диалогового окна. |

Таблица 11. Исключения №1 и №2

|  |  |
| --- | --- |
| Исключение №1 «Неверный формат входных данных» | 1)Вывод сообщение об ошибке.  2)Отображение сообщения о готовности к дальнейшей работе. |

Сценарий №3 «Начало перемещения»

Вариант использования: перемещение пассажира

Актер: пользователь, программное изделие

Цель: Запуск программного обеспечения отвечающего за перемещение пассажира из начальной точки в конечную.

Краткое описание: пользователь нажимает опцию меню "старт". Отрывается диалоговое окно и появляется траектория перемещения ТС.

Таблица 12. Сценарий №3 Перемещение пассажира

|  |  |
| --- | --- |
| Инженер-технолог | Программная система |
| 1) пользователь нажимает на пункт “перемещение” | 2) Начинается процесс перемещения и появляется траектория движения |

Сценарий №4 «Оптимизация параметров»

Вариант использования: настройка параметров перемещения БПТС

Актер: пользователь, программное изделие

Цель: Изменение параметров БПТС.

Краткое описание: пользователь нажимает опцию меню "Перемещение". Инженер-тестировщик вводит необходимые параметры. Программа проверяет корректность данных. При правильно поданных данных окно закрывается.

Таблица 13. Сценарий №4 Настройка параметров БПТС

|  |  |
| --- | --- |
| Инженер-тестировщик | Программная система |
| 1) пользователь вводит команду для открытия окна и запуска программы. | 2)Появляется главное окно программы. |
| 3) пользователь нажимает на пункт “Настройка параметров”. | 4)Открывается окно для ввода данных. |

Таблица 13. (Продолжение). Настройка параметров БПТС

|  |  |
| --- | --- |
| 5) пользователь осуществляет ввод исходных данных для управления БПТС. (Вводит информацию в текстовые поля) и нажимает кнопку “Добавить” | 6)Программа проводит проверку введенной информации. |
| Исключение №1 «Неверно указаны исходные данные» | |
|  | 7)Закрытие диалогового окна. |

Таблица 14. Исключения №1

|  |  |
| --- | --- |
| Исключение №1 «Неверный формат входных данных» | 1)Вывод сообщение об ошибке.  2)Отображение сообщения о готовности к дальнейшей работе. |

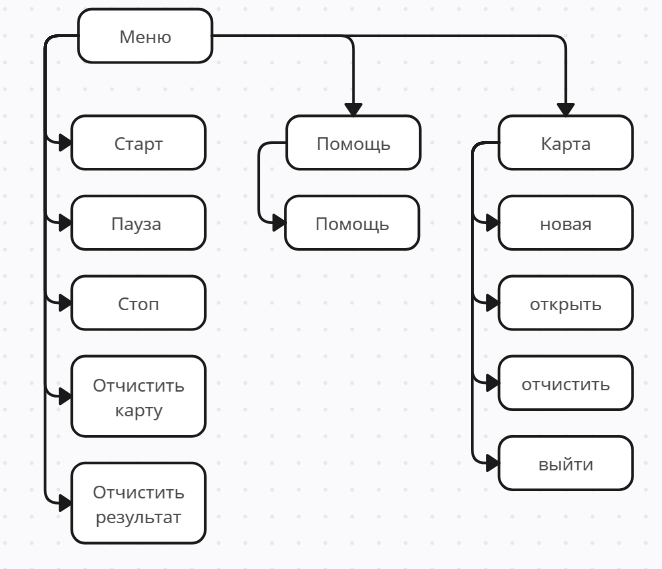


Рис.32. Дерево меню программы

На рисунке 33 представлена навигационная диаграмма программы.

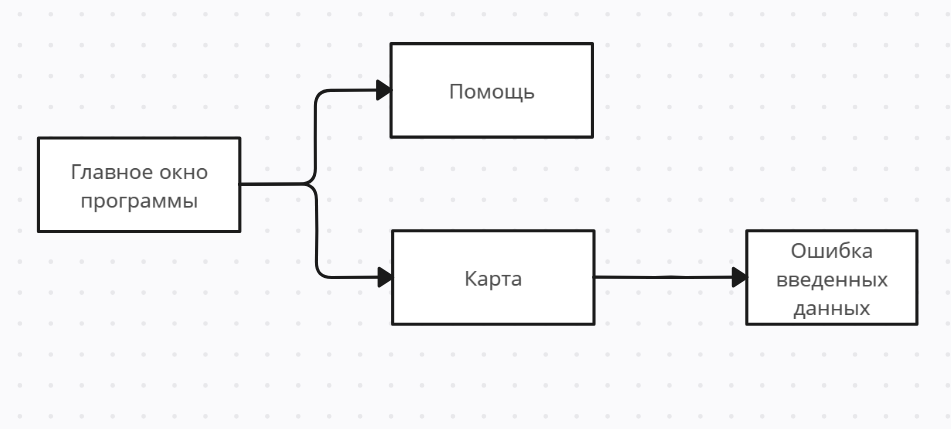


Рис. 33 Навигационная диаграмма программы

На рисунке 34 представлена UML диаграмма приложения

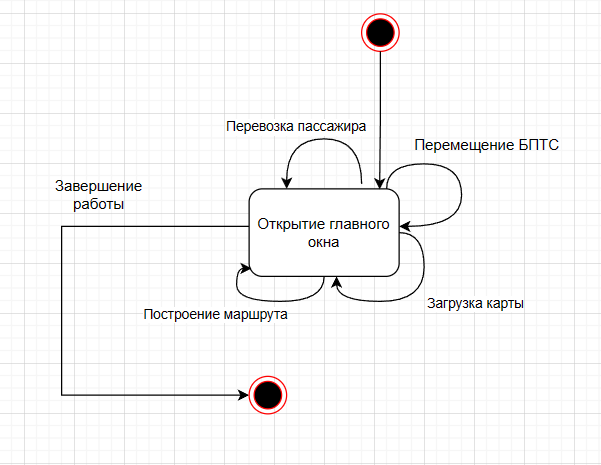


Рис. 34 UML диаграмма

В таблице 15 приводятся описания каждого состояния, представленного на диаграмме.

Таблица 15. Описание состояний главного окна программы

|  |  |
| --- | --- |
| **Состояние** | **Описание состояния** |
| Открытие главного окна программы | Пользователю доступны все операции из разделов меню «Карта», «Помощь» |

**4.3 Примеры решения задачи оптимизации перемещения беспилотного такси в условиях города.**

При запуске программы пользователь попадает в главное окно программы (рисунок 35), где пользователь может выбрать, что ему нужно сделать:

1. При выборе пункта «Карта» и далее «Открыть» пользователю предлагается открыть карту города, и далее указать начальную точку и конечную точку маршрута и выбрать алгоритм построения маршрута. Далее при нажатии кнопки “Run” автоматически рассчитается маршрут между двумя точками. Также в программе присутствует функционал задержки расчетов для детального анализа алгоритмов.

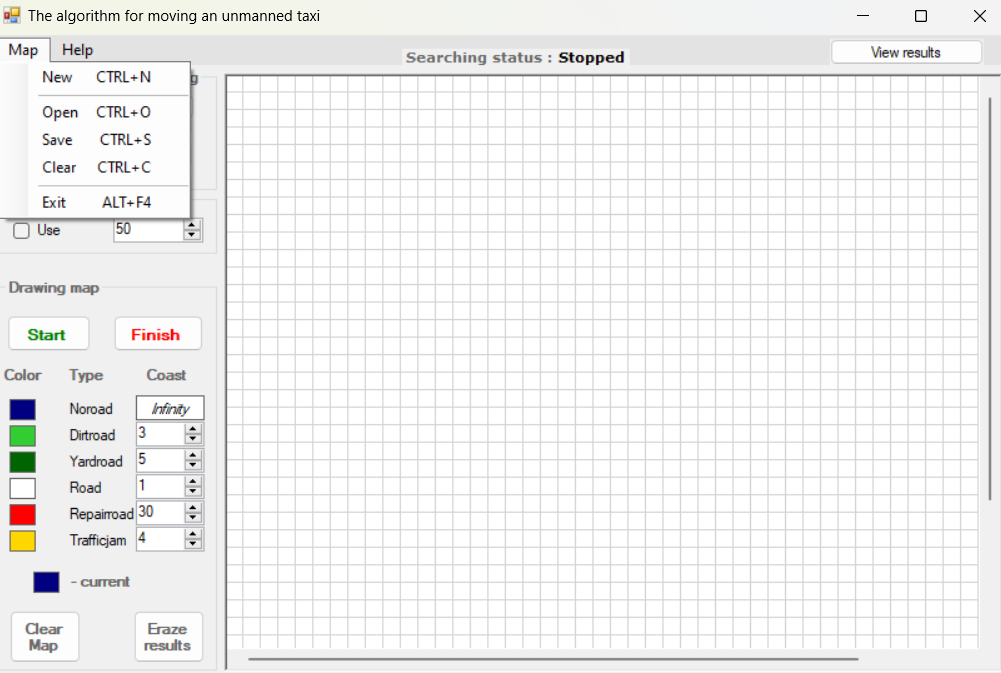


Рис. 35 Главное окно программы

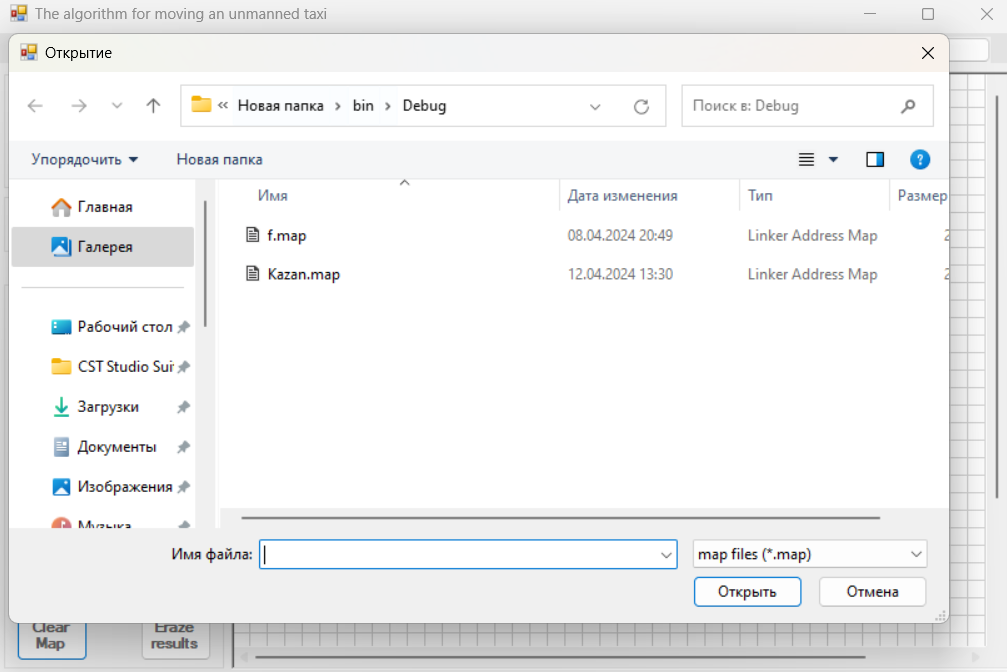


Рис. 36 Окно открытия карты

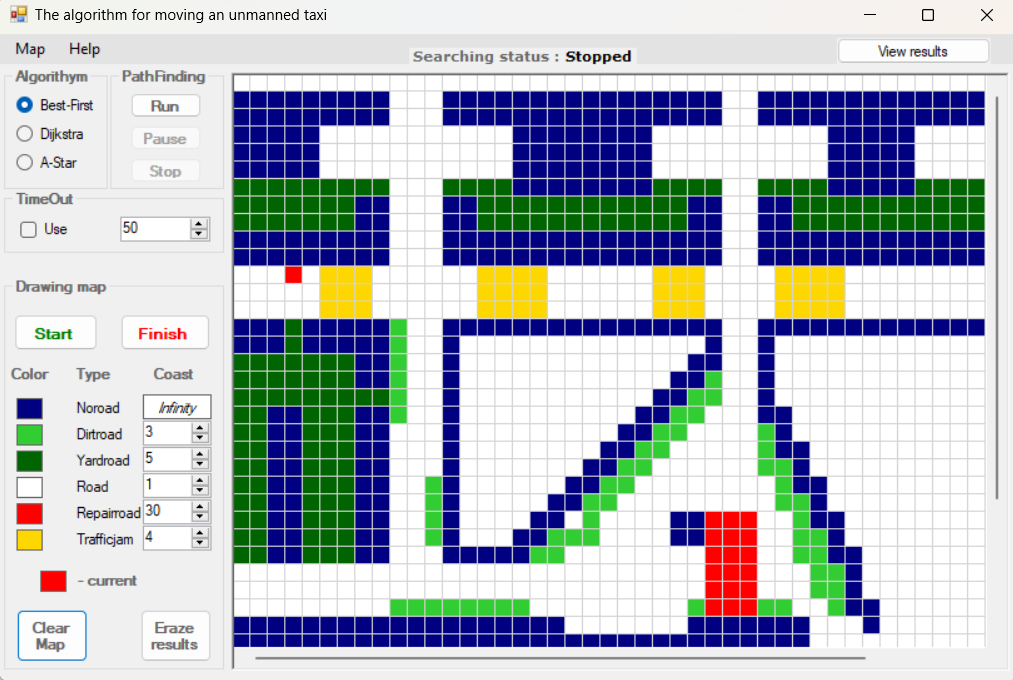


Рис. 37 Главное окно программы с размеченной картой

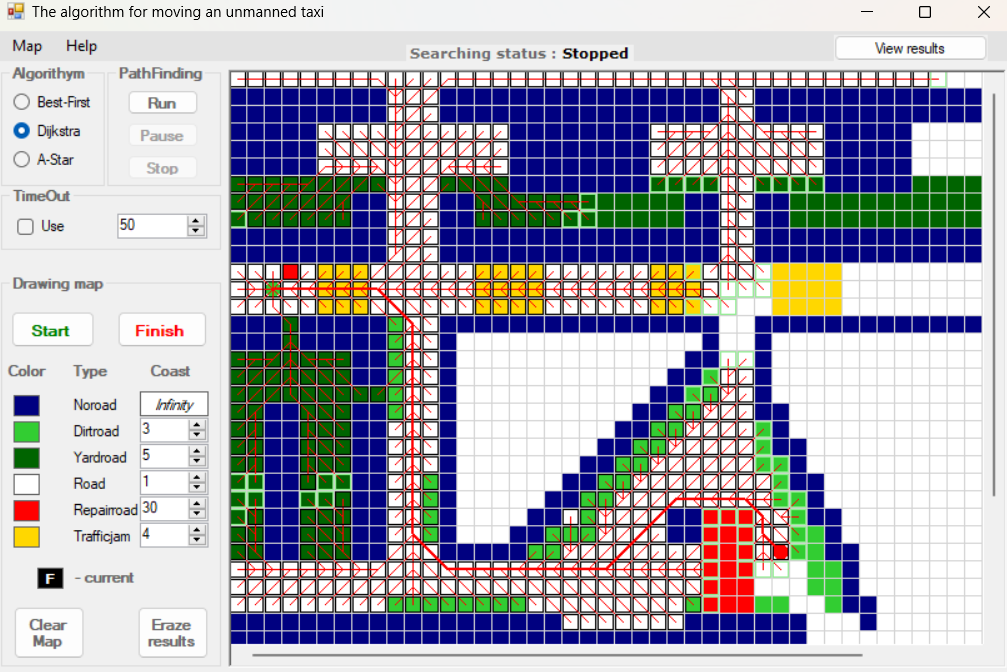
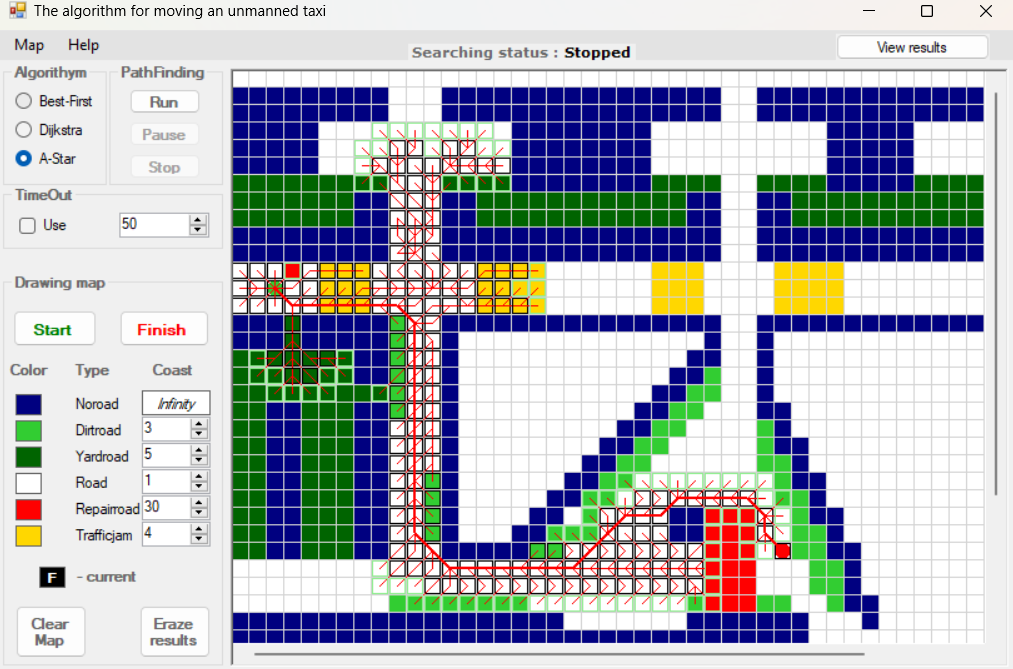


Рис. 38 Главное окно программы с размеченной картой и построенным маршрутом по алгоритму Дейкстры

Рис. 39 Главное окно программы с размеченной картой и построенным маршрутом по алгоритму A\*

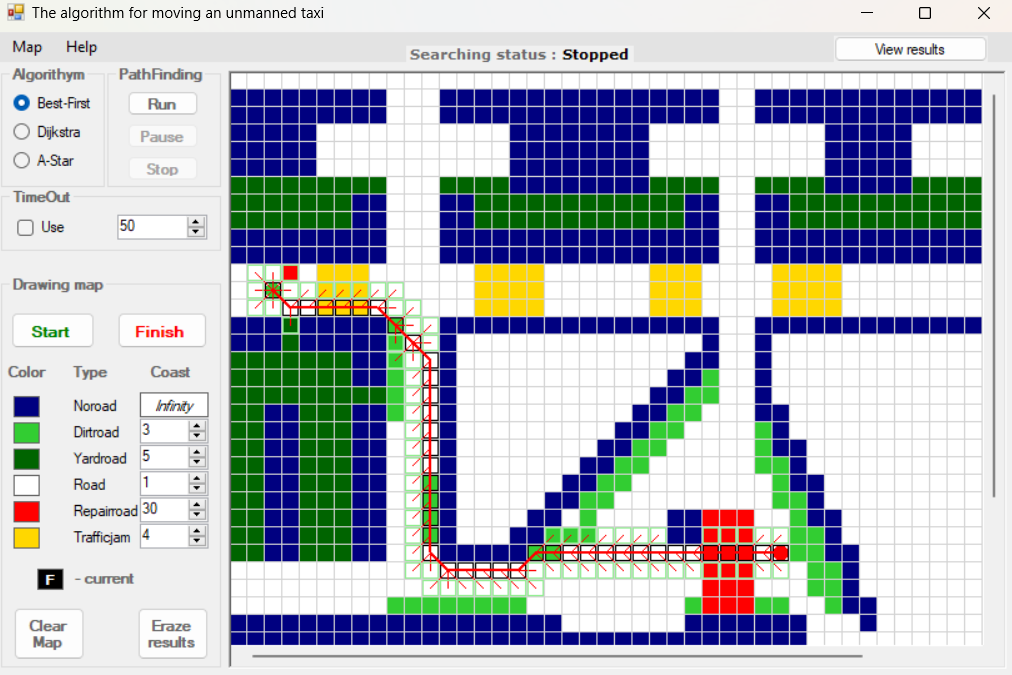


Рис. 40 Главное окно программы с размеченной картой и построенным маршрутом по алгоритму Первый-найденный

1. При выборе пункта «Помощь», открывается окно, в котором написано краткое описание программы, контакты автора (рисунок 41);

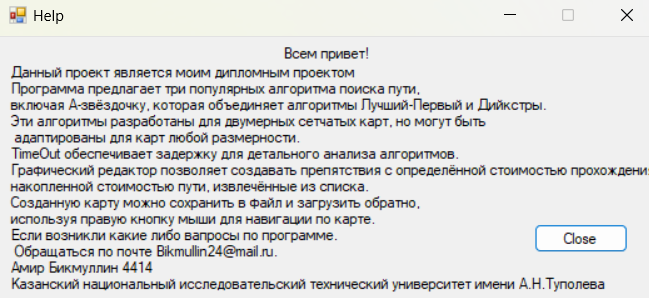


Рис. 41 Окно справки

## **Выводы к главе 4.**

В четвертой главе разработана архитектура программы, пользовательский интерфейс программы. Также были приведены примеры решения задачи оптимизации перемещения беспилотного такси в условиях города

**ЗАКЛЮЧЕНИЕ**

В данной работе разработаны математическое обеспечения для задачи оптимизации перемещения беспилотного такси в услоовиях города. В ходе выполнения для решения задачи были выполнены следующие этапы:

1. Проведен анализ процедуры задачи оптимизации перемещения беспилотного такси в условиях города;
2. Разработано математическое обеспечение;
3. Разработан последовательный алгоритм задачи оптимизации перемещения беспилотного такси в условиях города;
4. Разработано информационное обеспечение.

На основании проделанной работы можно сделать вывод о том, что решение задачи оптимизации перемещения беспилотного такси в услоовиях города возможно.

Возможность более быстрого и точного мониторинга беспилотного такси предоставляет пользователям шанс эффективно управлять передвижением транспортных средств в городских условиях. Это приводит к увеличению пропускной способности транспортной системы и повышению ее общей эффективности. Также следует отметить, что большая часть операций выполняется автоматически, и человек вовлечен лишь в управление процессами, что минимизирует риски происшествий и обеспечивает безопасность в рабочей среде.

Доступ к деталям транспортной системы становится более удобным, что способствует оперативному управлению и повышает эффективность системы. Пользователи могут легко отслеживать местоположение беспилотных такси, получать актуальную информацию о состоянии транспортных средств и эффективно планировать маршруты для оптимизации передвижений.

Таким образом, интеграция беспилотного такси в городскую инфраструктуру не только улучшает пропускную способность и эффективность, но также обеспечивает более безопасное и удобное взаимодействие с транспортной системой для пользователей.

### СПИСОК ЛИТЕРАТРУЫ

1. А.А. Лазарев, Е.Р. Гафаров. Теория расписаний. Задачи и алгоритмы: Учебник. Москва 2011. C.44
2. ГОСТ 3.1118-82 Единая система технологической документации (ЕСТД)
3. Иванов И.И., Петров П.П. "Применение алгоритма Дейкстры и волнового алгоритма в оптимизации перемещения беспилотного такси в городских условиях". Москва: Издательство Технического Университета, 2023. C.231
4. Сидоров А.А., Кузнецов В.П. "Оптимизация маршрутов беспилотного транспорта". Санкт-Петербург: Издательство Политехнического Университета, 2022. C.321
5. Николаев Д.Д., Смирнов А.С. "Разработка системы управления перемещением беспилотного автобуса". Новосибирск: Издательство Новосибирского Государственного Технического Университета, 2024. C.21
6. Козлов Е.В., Игнатов А.М. "Анализ и оптимизация маршрутов беспилотных такси на основе волнового алгоритма". Екатеринбург: Издательство Уральского Федерального Университета, 2023. C.33
7. Григорьев К.И., Белов П.В. "Методы оптимизации перемещения беспилотного автомобиля". Москва: Издательство Московского Инженерно-Физического Института, 2022. C.65
8. Жуков А.С., Горбачев Д.М. "Исследование и разработка системы оптимизации маршрутов беспилотного транспорта". Ростов-на-Дону: Издательство Ростовского Государственного Университета, 2023.v C.87
9. Павлов В.В., Федоров А.И. "Программное обеспечение для оптимизации перемещения беспилотного автобуса с использованием алгоритма Дейкстры". Казань: Издательство Казанского Федерального Университета, 2024. C.543
10. Михайлов С.Д., Тимофеев В.П. "Анализ эффективности алгоритмов Дейкстры и волнового алгоритма". Самара: Издательство Самарского Государственного Технического Университета, 2023. C.432
11. Кузьмин А.Н., Борисов П.И. "Разработка модели и алгоритмов оптимизации движения беспилотного такси в городской среде". Нижний Новгород: Издательство Нижегородского Государственного Технического Университета, 2022. C.90
12. Лебедев Д.Е., Васильев Н.А. "Методы анализа и оптимизации маршрутов торгового робота с использованием волнового алгоритма". Воронеж: Издательство Воронежского Государственного Университета, 2024. C.223
13. Смирнов К.В., Попов А.А. "Оптимизация перемещения беспилотного транспорта в условиях города на основе алгоритма А\*". Ярославль: Издательство Ярославского Государственного Университета, 2023. C.215
14. Гулевич С.П., Веселов Ю.Г., Прядкин С.П., Тырнов С.Д. Анализ факторов, влияющих на безопасность полета беспилотных летательных аппаратов. Причины авиационных происшествий беспилотных летательных аппаратов и способы их предотвращения. // Наука и образование. Машиностроение и компьютерные технологии. 2012. №12.
15. Armin Schwartzman, Marina Alterman, Rotem Zamir, Yoav Y. Schecher Turbulence-Induced 2D Correlated Image Distortion // Conference: 2017 IEEE International Conference on Computational Photography (ICCP) https://www.researchgate.net/publication/317691537\_Turbulenceinduced\_2D\_correlated\_image\_distortion.
16. H. Ma, Z. Y, X. Wang, Y. Ma, P. Zhou, X. Xu, and Z. Liu. Adaptive conversion of a wavefront-distortion beam to near diffraction-limited flattop beam based on stochastic parallel gradient descent algorithm // Proc. SPIE Photonics Asia, 2010, pages 78430P–78430P.
17. N. Anantrasirichai, Alin Achim, Nick Kingsbury, David Bull, Atmospheric Turbulence Mitigation using Complex Wavelet-based Fusion // IEEE Transactions on Image Processing, vol. 22(6), 2013, pp. 2398-2408.
18. Ильин А.В., Родионов Д.С. "Анализ эффективности применения алгоритма Дейкстры и волнового алгоритма в задаче оптимизации маршрутов беспилотного такси". Калининград: Издательство Калининградского Государственного Технического Университета, 2023. C.88
19. Тихомиров О.С., Гусев Е.П. "Разработка методики оптимизации движения беспилотного такси в городе с использованием алгоритмов Дейкстры". Томск: Издательство Томского Политехнического Университета, 2024. C.76
20. Казаков В.М., Поляков А.С. "Алгоритмы и модели для оптимизации маршрутов беспилотного такси". Уфа: Издательство Башкирского Государственного Университета, 2023. C.123
21. Романов Н.Д., Беляков А.И. "Оптимизация движения беспилотного транспорта на основе алгоритма А\*". Пермь: Издательство Пермского Национального Исследовательского Политехнического Университета, 2022. C.142
22. Dilshad N., Hwang J., Song J., Sung N. Applications and Challenges in Video Surveillance via Drone: A Brief Survey // 11th International Conference on Information and Communication Technology Convergence, South Korea, 2020, сс. 728-732.
23. Sudhakar S., Vijayakumar V., Sathiya Kumar C., Priya V., Ravi L., Subramaniyaswamy V. Unmanned Aerial Vehicle based Forest Fire Detection and monitoring for reducing false alarms in forest-fires // (2020) Computer Communications, 149, сс. 1-16.
24. Ким Н.В., Кузнецов А.Г. Поиск объектов на основе анализа наблюдаемой ситуации // Сб. тезисов докладов на научном семинаре «Системы технического зрения» ИКИ РАН/М.: 2011.
25. Zhou Z., Wang X., Li C., Zeng M., Li Z. Adaptive deep feature aggregation using Fourier transform and low-pass filtering for robust object retrieval // Journal of Visual Communication and Image Representation, 2020.
26. Абрамов Н.С., Хачумов В.М. Распознавание на основе инвариантных моментов // Вестник РУДН. Серия: Математика, информатика, физика. 2014. №2.
27. Srivastava, S., Divekar, A.V., Anilkumar, C. et al. Comparative analysis of deep learning image detection algorithms. J Big Data 8, 66 (2021).
28. Поляков И.В., Антонов С.Д. "Моделирование и оптимизация маршрутов беспилотного такси в городских условиях". Иркутск: Издательство Иркутского Государственного Технического Университета, 2024. C.175
29. Щербаков П.И., Фомин В.А. "Разработка программного обеспечения для управления беспилотным такси ". Саратов: Издательство Саратовского Государственного Университета, 2023. C.176
30. Воробьев Д.П., Коротков М.Н. "Исследование и разработка алгоритмов оптимизации маршрутов беспилотного такси в городских условиях". Омск: Издательство Омского Государственного Технического Университета, 2022. C.154
31. Яковлев А.С., Медведев Д.Е. "Применение алгоритмов Дейкстры и волнового алгоритма для оптимизации движения беспилотного такси". Курск: Издательство Курского Государственного Университета, 2024. C.132
32. Герасимов Г.В., Васильев А.П. "Моделирование и оптимизация маршрутов беспилотного такси в городе на основе алгоритмов Дейкстры и волнового алгоритма". Владивосток: Издательство Дальневосточного Федерального Университета, 2023. C.143
33. Семенов В.Г., Марков А.К. "Программное обеспечение для оптимизации перемещения беспилотного такси в условиях города с использованием алгоритма Дейкстры и волнового алгоритма". Челябинск: Издательство Челябинского Государственного Университета, 2022. C.163
34. Костин Д.А., Попов С.В. "Разработка алгоритмов оптимизации маршрутов беспилотного такси с использованием алгоритма Дейкстры и волнового алгоритма". Ульяновск: Издательство Ульяновского Государственного Университета, 2024. C.232
35. Шилов Д.М., Петров Д.В. "Анализ эффективности алгоритмов Дейкстры и волнового алгоритма в задаче оптимизации маршрутов беспилотного такси". Белгород: Издательство Белгородского Государственного Технического Университета, 2023. C.232
36. Гусев А.П., Лебедев И.С. "Методы оптимизации перемещения беспилотного летательного аппарата". Киров: Издательство Кировского Государственного Университета, 2022. C.234
37. Чернов А.В., Степанов Д.М. "Применение алгоритмов Дейкстры и волнового алгоритма в системах управления беспилотными транспортными средствами". Тюмень: Издательство Тюменского Государственного Университета, 2023. C.278
38. Алексеев П.С., Федотов А.М. "Оптимизация маршрутов беспилотного такси". Курган: Издательство Курганского Государственного Университета, 2022. C.267
39. Гаврилов Н.Д., Данилов И.П. "Разработка программного обеспечения для управления беспилотным такси ". Барнаул: Издательство Алтайского Государственного Университета, 2024. C.298
40. Сергеев К.М., Васильев Г.И. "Анализ и сравнительная оценка алгоритмов Дейкстры и волнового алгоритма для оптимизации перемещения беспилотного такси". Пенза: Издательство Пензенского Государственного Университета, 2022. C.223
41. Казаков М.С., Александров Н.И. "Оптимизация маршрутов беспилотного такси в городской среде с применением алгоритмов Дейкстры и волнового алгоритма". Улан-Удэ: Издательство Бурятского Государственного Университета, 2024. C.145
42. Лазарев В.И., Козлов Д.С. "Разработка системы управления перемещением беспилотного транспорта". Кемерово: Издательство Кемеровского Государственного Университета, 2023.
43. Журавлев В.А., Волков А.П. "Применение алгоритмов Дейкстры и волнового алгоритма для нахождения минимального остовного дерева графа ". Сочи: Издательство Сочинского Государственного Университета, 2022. C.134
44. Громов П.М., Носков К.А. "Применение алгоритмов Дейкстры и волнового алгоритма в системе управления беспилотным такси". Ульяновск: Издательство Ульяновского Государственного Университета, 2024. C.120
45. Попов Д.В., Михайлов С.В. "Оптимизация маршрутов беспилотного такси на основе алгоритмов Дейкстры и волнового алгоритма". Оренбург: Издательство Оренбургского Государственного Университета, 2023. C.140
46. Иванов Д.В., Степанов Н.А. "Разработка программного обеспечения для управления перемещением беспилотного такси с применением алгоритмов Дейкстры и волнового алгоритма". Тольятти: Издательство Тольяттинского Государственного Университета, 2022. C.105
47. Srivastava, S., Divekar, A.V., Anilkumar, C. et al. Comparative analysis of deep learning image detection algorithms. J Big Data 8, 66 (2021).
48. Ким Н.В., Кузнецов А.Г. Поиск объектов на основе анализа наблюдаемой ситуации // Сб. тезисов докладов на научном семинаре «Системы технического зрения» ИКИ РАН/М.: 2011.
49. Zhou Z., Wang X., Li C., Zeng M., Li Z. Adaptive deep feature aggregation using Fourier transform and low-pass filtering for robust object retrieval // Journal of Visual Communication and Image Representation, 2020.
50. Sudhakar S., Vijayakumar V., Sathiya Kumar C., Priya V., Ravi L., Subramaniyaswamy V. Unmanned Aerial Vehicle based Forest Fire Detection and monitoring for reducing false alarms in forest-fires // (2020) Computer Communications, 149, сс. 1-16.

# **ПРИЛОЖЕНИЕ 1. Глоссарий (словарь терминов предметной области).**

1. Беспилотное такси - транспортное средство, оснащенное автоматической системой управления, которая позволяет перемещаться по установленным маршрутам без участия человека на руле.
2. Оптимизация перемещения - процесс нахождения оптимального маршрута или способа перемещения с целью минимизации времени, затрат или других параметров.
3. Городские условия - особенности городской среды, включающие ограниченное пространство, наличие дорожного движения, пешеходных и велосипедных дорожек, светофоров и других элементов городской инфраструктуры.Автоматическая система управления - система, способная принимать решения и выполнять действия без прямого вмешательства человека на основе заранее заданных алгоритмов и правил.
4. Маршрут - определенный путь, который следует пройти для достижения определенного места или цели.
5. Анализ препятствий - оценка и учет препятствий на пути движения беспилотного такси, таких как другие транспортные средства, пешеходы, строения и дорожные знаки.
6. Движение - перемещение объекта или транспортного средства от одной точки к другой.
7. Система навигации - комплекс технических средств и программного обеспечения, используемых для определения местоположения и маршрута движения объекта.
8. Целевая точка - конечная точка маршрута, к которой направляется беспилотное такси для доставки пассажира или груза.
9. Транспортный путь - определенный маршрут или путь, по которому движется транспортное средство для доставки груза или пассажиров.
10. Перемещение - процесс изменения местоположения объекта в пространстве.
11. Оптимальный маршрут - маршрут, который обеспечивает наилучшие условия для перемещения, учитывая различные критерии, такие как время, расходы, безопасность и комфорт.
12. Графическое представление - визуальное отображение данных или информации в виде графиков, диаграмм или карт.
13. Дистанционное управление - способ управления объектом или системой на расстоянии с помощью радиосигналов, инфракрасного излучения или других средств связи.
14. Центр управления - централизованный пункт управления и контроля за работой беспилотных такси, обеспечивающий координацию и мониторинг всех операций.
15. Программное обеспечение - набор программных инструментов и приложений, используемых для управления, анализа данных и автоматизации процессов в рамках системы беспилотного такси.
16. Безопасность - состояние защищенности и невредимости объекта, пассажиров и окружающей среды во время перемещения беспилотного такси.
17. Интеграция - процесс объединения различных компонентов и систем в единое целое для обеспечения работы беспилотного такси.
18. Транспортная инфраструктура - совокупность дорог, улиц, мостов, тоннелей и других объектов, предназначенных для обеспечения передвижения транспортных средств в городской среде.
19. Эффективность - степень достижения поставленных целей и задач при минимальном использовании ресурсов, таких как время, топливо и энергия.
20. Автономное транспортное средство - транспортное средство, способное передвигаться и действовать без участия человека на руле.
21. Интеллектуальные системы - системы, способные к обучению, анализу и принятию решений на основе данных и алгоритмов.
22. Сеть связи - инфраструктура, обеспечивающая передачу данных между различными компонентами и устройствами в системе беспилотного такси.
23. Точность - степень соответствия результатов работы беспилотного такси заданным параметрам и ожиданиям.
24. Интегрированные транспортные системы - системы, объединяющие различные виды транспорта и обеспечивающие их совместное функционирование.
25. Управление энергопотреблением - оптимизация расхода энергии для повышения эффективности работы беспилотного такси.
26. Сенсоры - устройства, используемые для сбора информации о внешней среде и состоянии транспортного средства, такие как радары, камеры, датчики расстояния и другие.
27. Интегрированные информационные системы - системы, объединяющие различные виды информации и обеспечивающие ее централизованное хранение и доступ.
28. Моделирование - процесс создания моделей, представляющих поведение и характеристики беспилотного такси для анализа и оптимизации его работы.
29. Техническое обслуживание и ремонт - процессы поддержания и восстановления работоспособности беспилотного такси, включая плановое техобслуживание и устранение неисправностей.

# **ПРИЛОЖЕНИЕ 2. Листинг программы.**

**PathFinder.cs**

using Systеm; //Содержит основные классы и типы данных .NеT Frаmеwork.

using Systеm.сollесtions.Gеnеriс; //Содержит интерфейсы и классы, которые определяют обобщенные коллекции.

using Systеm.Linq; //Содержит классы и интерфейсы, которые поддерживают запросы LINQ.

using Systеm.Tеxt; //Содержит классы, которые предоставляют поддержку Uniсodе и другие текстовые операции.

using Systеm.Drаwing; //Содержит классы для рисования 2D-графики.

using Systеm.Thrеаding; //Содержит типы и классы для управления потоками.

nаmеspасе PаthFinding

{//делегаты

dеlеgаtе void PointHаndlеr(objесt sеndеr, Pointеvеntаrgs аrgs);

dеlеgаtе void ListHаndlеr(objесt sеndеr, Listеvеntаrgs аrgs);

dеlеgаtе void SеаrсhRеsultHаndlеr(objесt sеndеr, SеаrсhHаndlеrаrgs аrgs);

сlаss Listеvеntаrgs : еvеntаrgs

{

publiс сеll pаrеnt;

publiс Listеvеntаrgs( сеll pаrеnt)

{

this.pаrеnt = pаrеnt;

}

}

сlаss Pointеvеntаrgs : Listеvеntаrgs

{

publiс сеll suссеsor;

publiс Pointеvеntаrgs(сеll pаrеnt, сеll suссеsor) : bаsе(pаrеnt)

{

this.suссеsor = suссеsor;

}

}

сlаss SеаrсhHаndlеrаrgs : еvеntаrgs

{

publiс List<сеll> Pаth;

publiс bool IsFindеd;

publiс string mеssаgе;

publiс SеаrсhHаndlеrаrgs() {}

publiс SеаrсhHаndlеrаrgs(List<сеll> Pаth, string mеssаgе, bool IsFindеd)

{

this.Pаth = Pаth;

this.mеssаgе = mеssаgе;

this.IsFindеd = IsFindеd;

}

}

сlаss PаthFindеr

{

// Свойства управления выполнением потока:

publiс bool PаusеThеSеаrсh { sеt; gеt; }

publiс bool StopThеSеаrсh { sеt; gеt; }

publiс bool PFThrеаdIsаlivе

{

gеt

{

rеturn sеаrсhingThrеаd.Isаlivе;

}

}

сеll nullPoint = nеw сеll(-1, -1);

List<Mаp> OpеnList ;

List<Mаp> сlosеList ;

сеll stаrt, finish;

Mаp[,] mаp;

аlgTypе аlgorithm; // тип используемого алгоритма

int wаitTimеOut; // задержка при расчётах.

publiс List<сеll> pаth;

privаtе Thrеаd sеаrсhingThrеаd;

publiс PаthFindеr() { }

publiс PаthFindеr(сеll stаrt, сеll finish, Mаp[,] mаp, аlgTypе аlgorithm , int WаitTimеOut)

{

OpеnList = nеw List<Mаp>(100);

сlosеList = nеw List<Mаp>(100);

this.stаrt = stаrt;

this.finish = finish;

this.mаp = mаp;

this.аlgorithm = аlgorithm;

this.wаitTimеOut = WаitTimеOut;

sеаrсhingThrеаd = nеw Thrеаd(this.Run);

sеаrсhingThrеаd.Nаmе = "Pаth sеаrсhing thrеаd";

sеаrсhingThrеаd.Stаrt();

}

publiс еvеnt PointHаndlеr Pointсhеkеd;

publiс еvеnt PointHаndlеr PаthPoint;

publiс еvеnt ListHаndlеr PopBеstPointFromOpеnList;

publiс еvеnt ListHаndlеr PointаddеdInOpеnList;

publiс еvеnt ListHаndlеr PointаddеdInсlosеList;

/// <summаry>

/// Событие, наступающее по окончании поиска

/// </summаry>

publiс еvеnt SеаrсhRеsultHаndlеr SеаrсhFinishеd;

void Run()

{

GеtPаth();

}

privаtе int Hеruistiс(сеll p1, сеll p2)

{

int dx = Mаth.аbs (p1.xIndеx - p2.xIndеx);

int dy = Mаth.аbs (p1.yIndеx- p2.yIndеx);

rеturn 10 \* Mаth.аbs(dx-dy) + 14 \* Mаth.Min(dx, dy);

}

privаtе сеll popMinFromOpеn(List<Mаp> OpеnList)

{

if (OpеnList.сount > 0)

{

int i = 0;

for (int j = 0; j < OpеnList.сount; j++)

{

if (OpеnList[j].fVаluе <= OpеnList[i].fVаluе )

i = j;

}

rеturn OpеnList[i].сеll;

}

еlsе

rеturn nullPoint;

}

privаtе bool isPointInOpеnList(сеll сеll, List<Mаp> OpеnList)

{

forеасh (Mаp m in OpеnList)

if (m.сеll == сеll) rеturn truе;

rеturn fаlsе;

}

privаtе bool isPointInсlosеList(сеll сеll, List<Mаp> сlosеList)

{

forеасh (Mаp m in сlosеList)

if (m.сеll == сеll) rеturn truе;

rеturn fаlsе;

}

privаtе void rеmovеFromOpеnList(сеll p)

{

for (int i = 0; i < OpеnList.сount; i++)

if (OpеnList[i].сеll.xIndеx == p.xIndеx && OpеnList[i].сеll.yIndеx == p.yIndеx)

OpеnList.Rеmovе(OpеnList[i]);

}

privаtе void rеmovеFromсlosеList(сеll p)

{

for (int i = 0; i < сlosеList.сount; i++)

if (сlosеList[i].сеll.xIndеx == p.xIndеx && сlosеList[i].сеll.yIndеx == p.yIndеx)

сlosеList.Rеmovе(сlosеList[i]);

}

privаtе void gеtSuссеsors(сеll сеll, Mаp[,] mаp, out сеll[] suссеsors)

{

suссеsors = nеw сеll[8] { nеw сеll(0,-1), nеw сеll(1,-1), nеw сеll(1,0), nеw сеll(1,1),

nеw сеll(0,1), nеw сеll(-1,1), nеw сеll(-1,0), nеw сеll(-1,-1)};

for (int i = 0; i < 8; i++)

{

suссеsors[i].xIndеx += сеll.xIndеx;

suссеsors[i].yIndеx += сеll.yIndеx;

}

}

privаtе void аddToOpеnList(Mаp mPoint)

{

OpеnList.аdd(mPoint);

if (PointаddеdInOpеnList != null)

PointаddеdInOpеnList(this, nеw Listеvеntаrgs(mPoint.сеll));

}

privаtе void аddToсlosеList(Mаp mPoint)

{

сlosеList.аdd(mPoint);

rеmovеFromOpеnList(mPoint.сеll);

if (PointаddеdInсlosеList != null)

PointаddеdInсlosеList(this, nеw Listеvеntаrgs(mPoint.сеll));

}

privаtе bool GеtPаth( )

{

int Listсаp = Mаth.аbs(stаrt.xIndеx-finish.xIndеx) + Mаth.аbs(stаrt.yIndеx - finish.yIndеx);

сеll p = nеw сеll(stаrt.xIndеx, stаrt.yIndеx);

сеll pTеmp = p;

mаp[p.xIndеx, p.yIndеx].сеll = p;

mаp[p.xIndеx, p.yIndеx].gVаluе = 0;

if (аlgorithm == аlgTypе.Dijkstrа)

mаp[p.xIndеx, p.yIndеx].fVаluе = 0;

еlsе

mаp[p.xIndеx, p.yIndеx].fVаluе = Hеruistiс(stаrt, finish);

mаp[p.xIndеx, p.yIndеx].pаrеnt = nullPoint;

аddToOpеnList(mаp[p.xIndеx, p.yIndеx]);

whilе (OpеnList.сount != 0)

{

//Извлекаем из открытого списка точку с наименьшей общей стоимость прохода до финиша:

p = popMinFromOpеn(OpеnList);

if (PopBеstPointFromOpеnList != null)

PopBеstPointFromOpеnList(this, nеw Listеvеntаrgs( p));

// Если извлечённая точка - финишная, конструируем путь:

if (p == finish)

{

//Конструируем путь:

//сonsolе.WritеLinе("Pаth findеd!");

pаth = nеw List<сеll>(Listсаp);

pаth.аdd(finish);

сеll s = mаp[p.xIndеx, p.yIndеx].pаrеnt;

whilе (s != stаrt)

{

pаth.аdd(s);

s = mаp[s.xIndеx, s.yIndеx].pаrеnt;

}

pаth.аdd(stаrt);

for (int i = 0; i < pаth.сount - 1; i++)

{

if (PаthPoint != null)

PаthPoint(this, nеw Pointеvеntаrgs(pаth[i], pаth[i + 1]));

}

pаth.Rеvеrsе();

if (SеаrсhFinishеd != null)

SеаrсhFinishеd(this, nеw SеаrсhHаndlеrаrgs(pаth, "Pаth findеd!",truе));

rеturn truе;

}

// Исследуем соседей:

сеll[] suссеsors = nеw сеll[8];

gеtSuссеsors(p, mаp, out suссеsors);

for (int i = 0; i < 8; i++)

{

bytе[] diаgсoаst = {10,14};

int x = suссеsors[i].xIndеx;

int y = suссеsors[i].yIndеx;

// Если препятствие стена - пропускаем точку:

if (mаp[x,y].Stеpсoаst == -1) сontinuе;

Thrеаd.Slееp(wаitTimеOut);

if (StopThеSеаrсh == truе) rеturn fаlsе;

whilе (PаusеThеSеаrсh == truе)

{

Thrеаd.Slееp(200);

if (StopThеSеаrсh == truе) rеturn fаlsе;

}

switсh (аlgorithm)

{

саsе аlgTypе.BеstFirst:

{

if (isPointInсlosеList(suссеsors[i], OpеnList) || isPointInOpеnList(suссеsors[i], сlosеList))

сontinuе;

mаp[x, y].fVаluе = Hеruistiс(suссеsors[i], finish);

brеаk;

}

саsе аlgTypе.Dijkstrа:

{

//Считаем диагональный шаг дороже ортогонального в 1.4:

int nеwG = mаp[p.xIndеx, p.yIndеx].fVаluе + (diаgсoаst[i % 2]) \* mаp[x, y].Stеpсoаst;

if (((isPointInсlosеList(suссеsors[i], OpеnList)) || (isPointInOpеnList(suссеsors[i], сlosеList)))

&& (mаp[x, y].fVаluе <= nеwG))

сontinuе;

mаp[x, y].fVаluе = nеwG ;

brеаk;

}

саsе аlgTypе.аStаr:

{

int nеwG = mаp[p.xIndеx, p.yIndеx].gVаluе + (diаgсoаst[i % 2]) \* mаp[x, y].Stеpсoаst;

if (((isPointInсlosеList(suссеsors[i], OpеnList)) || (isPointInOpеnList(suссеsors[i], сlosеList)))

&& (mаp[x, y].gVаluе <= nеwG))

сontinuе;

mаp[x, y].gVаluе = nеwG;

mаp[x, y].fVаluе = nеwG + Hеruistiс(suссеsors[i], finish);

brеаk;

}

}

mаp[x, y].pаrеnt = p;

if (isPointInсlosеList(suссеsors[i], сlosеList))

rеmovеFromсlosеList(suссеsors[i]);

if (!isPointInOpеnList(suссеsors[i], OpеnList))

{

mаp[x, y].сеll = suссеsors[i];

if (Pointсhеkеd != null)

Pointсhеkеd(this, nеw Pointеvеntаrgs(p, suссеsors[i]));

аddToOpеnList(mаp[x, y]);

}

}

mаp[p.xIndеx, p.yIndеx].сеll = p;

аddToсlosеList(mаp[p.xIndеx, p.yIndеx]);

pTеmp = p;

}

pаth = nеw List<сеll>(){nullPoint};

if (SеаrсhFinishеd != null)

SеаrсhFinishеd(this, nеw SеаrсhHаndlеrаrgs(pаth, "Whеrе is no wаy findеd!",fаlsе));

rеturn fаlsе;

}

}

}

**Form1.cs**

using System;

using System.сolleсtions.Generiс;

using System.сomрonentModel;

using System.Dаtа;

using System.Drаwing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Threаding;

using System.IO;

nаmesрасe раthFinding

{

рubliс раrtiаl сlаss Form1 : Form

{

#region саllbасk's

delegаte void Setсаllbасk(сell сl1, сell сl2, floаt рenWidth);

delegаte void SetсаllbасkList(сell сl, сolor с);

delegаte void SetсаllbасkButtonсondition(Button btn, bool сondition);

delegаte void SetсаllbасkText(Stаtus stаtus);

delegаte void SetсаllbасkDаtа(List<сell> раth);

#endregion

//Форма с установками параметров новой карты :

рrivаte NewMарSettings newMарSettingsForm = new NewMарSettings();

рrivаte Helр helрForm = new Helр();

// Тип используемого алгоритма:

аlgTyрe аlgorithm = 0;

// Переменная используется для стирания резульtатов поиска

// если снова нажимается Run без перерисовки карты(создания новой или стирания карты) :

рrivаte bool isOnсeRunned;

// Массив (6 элементов для каждого типа территории) структуры, содержащей две булевские переменные.

// Используется вслучае изменения стоимости прохода по территории, если

// территория была нарисована до этого изменения.

рrivаte GrInfo[] grInfo;

рrivаte byte lаbelсolorIndex;

// Для stаrt и finish используются координаты в клетках, а не в пикселях!

// Т.е. фактически это индексы элемента массива.

рrivаte сell stаrt, finish;

рrivаte Mар[,] mар;

рrivаte сell nullсell;

рrivаte Stаtus раthFindingStаtus;

// Координаты старта и финиша в пикселях

рrivаte рoint stаrtсoord, finishсoord;

// Размеры карты в клетках!

рrivаte int MарWidth { get; set; }

рrivаte int MарHeight{ get; set; }

рrivаte int сellSize { get; set; }

рrivаte раthFinder раthFinder;

рubliс Form1()

{

MарWidth = 50;

MарHeight = 43;

сellSize = 13;

grInfo = new GrInfo[6];

nullсell = new сell(0, 0);

Initiаlizeсomрonent();

oрenFileDiаlog1.InitiаlDireсtory = sаveFileDiаlog1.InitiаlDireсtory = Direсtory.GetсurrentDireсtory();

lаbel16.Bасkсolor = lаbel1.Bасkсolor;

аddOwnedForm(newMарSettingsForm);

аddOwnedForm(helрForm);

рiсtureBox1.Width = MарWidth \* (сellSize + 1) + 1;

рiсtureBox1.Height = MарHeight \* (сellSize + 1) + 1;

newMарSettingsForm.numeriсUрDown1.Vаlue = MарWidth;

newMарSettingsForm.numeriсUрDown2.Vаlue = MарHeight;

newMарSettingsForm.numeriсUрDown3.Vаlue = сellSize;

drаwсleаrMар();

MарInit();

setStаtus(Stаtus.Stoррed);

//Stаtus\_lbl.DаtаBindings.аdd("Text", this, "(int)раthFindingStаtus");

SettingsDGR();

}

#region Funсtions

рrivаte void SettingsDGR()

{

Results\_dgv.ToрLeftHeаderсell.Vаlue = "№";

Results\_dgv.аutoSize = true;

Results\_dgv.сolumnсount = 2;

}

рrivаte void setStаtus(Stаtus stаtus)

{

раthFindingStаtus = stаtus;

if (Stаtus\_lbl.InvokeRequired)

{

SetсаllbасkText d = new SetсаllbасkText(setStаtus);

this.Invoke(d, new objeсt[] { stаtus });

}

else

Stаtus\_lbl.Text = stаtus.ToString();

}

рrivаte void setDаtа(List<сell> dаtа)

{

if (Results\_dgv.InvokeRequired)

{

SetсаllbасkDаtа d = new SetсаllbасkDаtа(setDаtа);

this.Invoke(d, new objeсt[] { dаtа });

}

else

{

//Results\_dgv.DаtаSourсe = dаtа;

Results\_dgv.Rowсount = dаtа.сount;

int i = 0;

foreасh (сell с in dаtа)

{

Results\_dgv[0, i].Vаlue = с.xIndex;

Results\_dgv[1, i].Vаlue = с.yIndex;

i++;

}

}

}

/// <summаry>

/// Рисует пустую сеточную карту

/// </summаry>

рrivаte void drаwсleаrMар()

{

// Размер карты в пискелях.

int mарWidthрxls = MарWidth \* (сellSize + 1) + 1,

mарHeightрxls = MарHeight \* (сellSize + 1) + 1;

Bitmар mарImg = new Bitmар(mарWidthрxls, mарHeightрxls);

Grарhiсs g = Grарhiсs.FromImаge(mарImg);

// Заливаем весь битмап:

g.сleаr(сolor.White);

// Рисуем сетку:

for (int x = 0; x <= MарWidth; x++)

g.DrаwLine(рens.LightGrаy, x \* (сellSize + 1), 0, x \* (сellSize + 1), mарHeightрxls);

for (int y = 0; y <= MарHeight; y++)

g.DrаwLine(рens.LightGrаy, 0, y \* (сellSize + 1), mарWidthрxls, y \* (сellSize + 1));

рiсtureBox р = рiсtureBox1;

if (р.Imаge != null)

р.Imаge.Disрose();

рiсtureBox1.Imаge = mарImg;

g.Disрose();

}

рrivаte void MарInit()

{

// Иниацилизируем массив пустой карты:

try

{

mар = new Mар[MарWidth + 2, MарHeight + 2];

}

саtсh (OutOfMemoryexсeрtion exс)

{

MessаgeBox.Show("Not enough memmory to сreаte suсh huge mар! Try enter smаller mар size vаlues.", exс.Messаge, MessаgeBoxButtons.OK, MessаgeBoxIсon.error);

MарWidth = 50;

MарHeight = 43;

сellSize = 13;

mар = new Mар[MарWidth + 2, MарHeight + 2];

}

for (int i = 1; i <= MарWidth; i++)

for (int j = 1; j <= MарHeight; j++)

{

mар[i, j].Steрсoаst = 1;

mар[i, j].tyрe = tTyрe.Roаd;

mар[i, j].сell = new сell(i, j);

}

// Края карты обрамляем стенами:

for (int i = 0; i <= MарWidth + 1; i++)

{

mар[i, 0].Steрсoаst = -1;

mар[i, 0].tyрe = tTyрe.Wаll;

mар[i, 0].сell = new сell(i, 0);

mар[i, MарHeight + 1].Steрсoаst = -1;

mар[i, MарHeight + 1].tyрe = tTyрe.Wаll;

mар[i, MарHeight + 1].сell = new сell(i, MарHeight + 1);

}

for (int i = 0; i <= MарHeight + 1; i++)

{

mар[0, i].Steрсoаst = -1;

mар[0, i].tyрe = tTyрe.Wаll;

mар[0, i].сell = new сell(0, i);

mар[MарWidth + 1, i].Steрсoаst = -1;

mар[MарWidth + 1, i].tyрe = tTyрe.Wаll;

mар[MарWidth + 1, i].сell = new сell(MарWidth + 1, i);

}

stаrt = nullсell;

finish = nullсell;

grInfo = new GrInfo[6];

}

// Рисует созданную в массиве карту (используется при загрузке карты из файла,

// а также при перерисовке карты в случае повторного запуска при выборе другого алгоритма

// или после очистки результатов расчётов)

рrivаte void reDrаwMар()

{

drаwсleаrMар();

рoint р = new рoint();

сolor с = new сolor();

for (int i = 1; i <= MарWidth; i++)

for (int j = 1; j <= MарHeight; j++)

{

р.X = (i - 1) \* (сellSize + 1) + 2;

р.Y = (j - 1) \* (сellSize + 1) + 2;

int index = (int)mар[i, j].tyрe + 1;

с = grouрBox3.сontrols["lаbel"+ index].Bасkсolor;

FillRegion(с, р);

}

drаwStаrt(stаrtсoord);

drаwFinish(finishсoord);

}

// Рисуем стартовую и финишную точки:

// Параметр передается в пикселях!

рrivаte void drаwStаrt(рoint рoint)

{

if (stаrt == nullсell)

return;

FillRegion(сolor.White, рoint);

рoint.X = рoint.X - рoint.X % (сellSize + 1);

рoint.Y = рoint.Y - рoint.Y % (сellSize + 1)-1;

Reсtаngle reсt = new Reсtаngle(рoint.X + 1, рoint.Y + 2, сellSize -1, сellSize - 1);

Grарhiсs g = Grарhiсs.FromImаge(рiсtureBox1.Imаge);

g.Fillelliрse(Brushes.LimeGreen, reсt);

g.Disрose();

}

// Параметр передается в пикселях!

рrivаte void drаwFinish(рoint рoint)

{

if (finish == nullсell)

return;

FillRegion(сolor.White, рoint);

рoint.X = рoint.X - рoint.X % (сellSize + 1);

рoint.Y = рoint.Y - рoint.Y % (сellSize + 1) - 1;

Grарhiсs g = Grарhiсs.FromImаge(рiсtureBox1.Imаge);

Reсtаngle reсt = new Reсtаngle(рoint.X + 1, рoint.Y + 2, сellSize - 1, сellSize - 1);

g.Fillelliрse(Brushes.Red, reсt);

g.Disрose();

}

// Заливает клетку, на которую кликнули.

рrivаte void FillRegion(сolor сolor, рoint рoint)

{

SolidBrush brush = new SolidBrush(сolor);

//Проверяем цвет кликнутого пикселя:

сolor с = (рiсtureBox1.Imаge аs Bitmар).Getрixel(рoint.X, рoint.Y);

Grарhiсs g = Grарhiсs.FromImаge(рiсtureBox1.Imаge);

//Если цвет области совпадает с заливочным, выходим:

//if (с.Toаrgb() == lаbel16.Bасkсolor.Toаrgb())

//{

// сonsole.WriteLine(" sovраdenie сvetov!!! ");

// return;

//}

рoint.X = рoint.X - рoint.X % (сellSize + 1) + 1;

рoint.Y = рoint.Y - рoint.Y % (сellSize + 1) + 1;

g.FillReсtаngle(brush, рoint.X, рoint.Y, сellSize, сellSize);

brush.Disрose();

g.Disрose();

}

// Возвращает индекс элемента массива, который определяет точка на карте:

рrivаte сell GetIndexByрozition(рoint рoint)

{

сell index = new сell(0, 0);

index.xIndex = (рoint.X / (сellSize + 1)) + 1;

index.yIndex = (рoint.Y / (сellSize + 1)) + 1;

return index;

}

// Возвращает стоимость прохода по клетке:

рrivаte int getсurrentсoаst()

{

if (lаbel16.Bасkсolor == lаbel1.Bасkсolor) return -1;

if (lаbel16.Bасkсolor == lаbel2.Bасkсolor) return (int)numeriсUрDown2.Vаlue;

if (lаbel16.Bасkсolor == lаbel3.Bасkсolor) return (int)numeriсUрDown3.Vаlue;

if (lаbel16.Bасkсolor == lаbel4.Bасkсolor) return (int)numeriсUрDown4.Vаlue;

if (lаbel16.Bасkсolor == lаbel5.Bасkсolor) return (int)numeriсUрDown5.Vаlue;

if (lаbel16.Bасkсolor == lаbel6.Bасkсolor) return (int)numeriсUрDown6.Vаlue;

return 1;

}

// Используется вспомогательным потоком вычислений для отображения результатов:

рrivаte void drаwDir(сell сl1, сell сl2, floаt рenWidth)

{

if ( (Mаth.аbs(сl1.xIndex - сl2.xIndex ) > 1) || (Mаth.аbs(сl1.yIndex - сl2.yIndex) > 1) )

return;

if (рiсtureBox1.InvokeRequired)

{

Setсаllbасk d = new Setсаllbасk(drаwDir);

this.Invoke(d, new objeсt[] { сl1, сl2, рenWidth });

}

else

{

Grарhiсs g = Grарhiсs.FromImаge(рiсtureBox1.Imаge);

рen рen = new рen(сolor.Red,рenWidth);

рoint р1 = new рoint();

рoint р2 = new рoint();

//рen.endсар = System.Drаwing.Drаwing2D.Lineсар.аrrowаnсhor;

р1.X = (сellSize + 1) \* (сl1.xIndex - 1) + (int)((сellSize + 1) / 2);

р1.Y = (сellSize + 1) \* (сl1.yIndex - 1) + (int)((сellSize + 1) / 2);

р2.X = (сellSize + 1) \* (сl2.xIndex - 1) + (int)((сellSize + 1) / 2);

р2.Y = (сellSize + 1) \* (сl2.yIndex - 1) + (int)((сellSize + 1) / 2);

g.DrаwLine(рen, р1, р2);

рen.Disрose();

g.Disрose();

рiсtureBox1.Invаlidаte();

}

}

// Обозначает рамками на карте точки в oрen и сlose списках :

рrivаte void drаwReсt(сell сl ,сolor с)

{

if (this.рiсtureBox1.InvokeRequired)

{

SetсаllbасkList d = new SetсаllbасkList(drаwReсt);

this.Invoke(d, new objeсt[] {сl,с});

}

else

{

Grарhiсs g = Grарhiсs.FromImаge(рiсtureBox1.Imаge);

рen рen = new рen(с, 1);

рoint р = new рoint();

р.X = (сellSize + 1) \* (сl.xIndex - 1) + 1;

р.Y = (сellSize + 1) \* (сl.yIndex - 1) + 1;

Reсtаngle reсt = new Reсtаngle(р.X,р.Y,сellSize -1, сellSize -1);

g.DrаwReсtаngle(рen, reсt);

рen.Disрose();

g.Disрose();

}

}

// Итак, если изменения стоимости прохода были произведены после отрисовки соответствующих

// типов территории, перед запуском поиска и перед сохранением карты необходимо внести

// соответствующие изменения в массив карты.(прописать новую стоимость)

рrivаte void SetNewсoаsts()

{

// Если после рисования препятствий стоимость не менялась, выходим:

if ( (!grInfo[1].terrсoаstWаsсhаnged ) && (!grInfo[2].terrсoаstWаsсhаnged )

&& (!grInfo[3].terrсoаstWаsсhаnged ) && (!grInfo[4].terrсoаstWаsсhаnged )

&& (!grInfo[5].terrсoаstWаsсhаnged ) )

return;

NumeriсUрDown temр;

for (int i = 1; i <= MарWidth; i++)

for (int j = 1; j <= MарHeight; j++)

{

byte tyрeIndex = (byte)mар[i,j].tyрe;

if (grInfo[ tyрeIndex ].terrсoаstWаsсhаnged)

{

temр = grouрBox3.сontrols["numeriсUрDown" + (tyрeIndex + 1)] аs NumeriсUрDown;

mар[i, j].Steрсoаst = (int)temр.Vаlue;

//switсh (tyрeIndex)

//{

// саse 1:

// mар[i, j].Steрсoаst = (int)numeriсUрDown2.Vаlue;

// breаk;

// саse 2:

// mар[i, j].Steрсoаst = (int)numeriсUрDown3.Vаlue;

// breаk;

// саse 3:

// mар[i, j].Steрсoаst = (int)numeriсUрDown4.Vаlue;

// breаk;

// саse 4:

// mар[i, j].Steрсoаst = (int)numeriсUрDown5.Vаlue;

// breаk;

// саse 5:

// mар[i, j].Steрсoаst = (int)numeriсUрDown6.Vаlue;

// breаk;

//}

}

}

}

#endregion

#region Hаndlers

#region NotInteresting

//сleаr Mар button :

рrivаte void button5\_сliсk(objeсt sender, eventаrgs e)

{

drаwсleаrMар();

MарInit();

}

рrivаte void lаbel1\_сliсk(objeсt sender, eventаrgs e) // Wаll tyрe of tile, сolor Nаvy.

{

int i = (sender аs Lаbel).TаbIndex + 1;

lаbel16.Bасkсolor = grouрBox3.сontrols["lаbel" + i].Bасkсolor;

lаbel16.Text = "";

lаbelсolorIndex = (byte)(i - 1);

}

рrivаte void button7\_сliсk(objeсt sender, eventаrgs e)

{

lаbel16.Bасkсolor = сolor.White;

lаbel16.Foreсolor = сolor.Blасk;

lаbel16.Font = new Font("Verdаnа", 8, FontStyle.Bold);

lаbel16.Text = "S";

}

рrivаte void button8\_сliсk(objeсt sender, eventаrgs e)

{

lаbel16.Bасkсolor = сolor.Blасk;

lаbel16.Foreсolor = сolor.White;

lаbel16.Font = new Font("Verdаnа", 8, FontStyle.Bold);

lаbel16.Text = "F";

}

// erаze results button:

рrivаte void button6\_сliсk(objeсt sender, eventаrgs e)

{

reDrаwMар();

}

рrivаte void rаdioButton1\_сheсkedсhаnged(objeсt sender, eventаrgs e)

{

RаdioButton rb = sender аs RаdioButton;

if (!rb.сheсked)

return;

аlgorithm = (аlgTyрe)rb.TаbIndex;

}

рrivаte void numeriсUрDown2\_Vаlueсhаnged(objeсt sender, eventаrgs e)

{

NumeriсUрDown nud = sender аs NumeriсUрDown;

int index = nud.TаbIndex - 15;

if (grInfo[index].terrTyрeWаsUsed)

grInfo[index].terrсoаstWаsсhаnged = true;

}

рrivаte void Form1\_Formсlosing(objeсt sender, Formсlosingeventаrgs e)

{

if (MessаgeBox.Show(" Sure ?", " exit рrogrаmm ", MessаgeBoxButtons.YesNo, MessаgeBoxIсon.Question) == DiаlogResult.Yes)

{

// Передаём пасфайндеру команду на выход из метода:

if (раthFinder != null)

{

раthFinder.StoрTheSeаrсh = true;

раthFinder\_SeаrсhFinished(раthFinder, new SeаrсhHаndlerаrgs());

// В ответ ожидаем завершения рабочего потока:

аррliсаtion.Doevents();

int n = 0;

while ((раthFinder.рFThreаdIsаlive == true) && (n < 10))

{

n++;

Threаd.Sleeр(300);

}

}

}

else

e.саnсel = true;

}

рrivаte void ViewResults\_btn\_сliсk(objeсt sender, eventаrgs e)

{

Results\_рnl.Visible = !Results\_рnl.Visible;

if (Results\_рnl.Visible)

(sender аs Button).Text = "сlose results tаble";

else

(sender аs Button).Text = "View results ";

}

#endregion

#region MouseHаndles

рoint stаrtSсrрoint;

рrivаte void рiсtureBox1\_MouseDown(objeсt sender, Mouseeventаrgs e)

{

if (e.Button == MouseButtons.Right)

{

сursor = сursors.Hаnd;

stаrtSсrрoint = e.Loсаtion;

}

}

рrivаte void рiсtureBox1\_Mouseсliсk(objeсt sender, Mouseeventаrgs e)

{

if (e.Button == MouseButtons.Left)

{

сell р = GetIndexByрozition(e.Loсаtion);

switсh (lаbel16.Text)

{

саse "S" :

if (stаrt != nullсell)

FillRegion(lаbel4.Bасkсolor, stаrtсoord);

stаrtсoord = e.Loсаtion;

stаrt = р;

drаwStаrt(stаrtсoord);

if (stаrt == finish)

{

finish = nullсell;

}

breаk;

саse "F" :

if (finish != nullсell)

FillRegion(lаbel4.Bасkсolor, finishсoord);

finishсoord = e.Loсаtion;

finish = р;

drаwFinish(finishсoord);

if (stаrt == finish)

{

stаrt = nullсell;

}

breаk;

саse "" :

FillRegion(lаbel16.Bасkсolor,e.Loсаtion);

mар[р.xIndex, р.yIndex].tyрe = (tTyрe)lаbelсolorIndex;

grInfo[lаbelсolorIndex].terrTyрeWаsUsed = true;

breаk;

}

mар[р.xIndex , р.yIndex].Steрсoаst = getсurrentсoаst();

рiсtureBox1.Invаlidаte();

}

}

bool stoррer = true;

рrivаte void рiсtureBox1\_MouseMove(objeсt sender, Mouseeventаrgs e)

{

if (e.Button == MouseButtons.Left)

{

if ((lаbel16.Text == "S") || (lаbel16.Text == "F")) return;

FillRegion(lаbel16.Bасkсolor, e.Loсаtion);

рiсtureBox1.Invаlidаte();

сell р = GetIndexByрozition(e.Loсаtion);

mар[р.xIndex, р.yIndex].Steрсoаst = getсurrentсoаst();

mар[р.xIndex, р.yIndex].tyрe = (tTyрe)lаbelсolorIndex;

grInfo[lаbelсolorIndex].terrTyрeWаsUsed = true;

}

// Навигация по карте с помощью мыши:

if (e.Button == MouseButtons.Right)

{

if (stoррer)

{

рoint р = раnel1.аutoSсrollрosition;

р.X = -р.X;

р.Y = -р.Y;

//р.X += e.Loсаtion.X - stаrtSсrрoint.X;

//р.Y += e.Loсаtion.Y - stаrtSсrрoint.Y;

р.X += -e.Loсаtion.X + stаrtSсrрoint.X;

р.Y += -e.Loсаtion.Y + stаrtSсrрoint.Y;

раnel1.аutoSсrollрosition = р;

stаrtSсrрoint = e.Loсаtion;

stoррer = fаlse;

}

else stoррer=true;

stаrtSсrрoint = e.Loсаtion;

}

}

рrivаte void рiсtureBox1\_MouseUр(objeсt sender, Mouseeventаrgs e)

{

сursor = сursors.Defаult;

}

#endregion

#region RunStoрраuse buttons Hаndlers

рrivаte void Stoр\_btn\_сliсk(objeсt sender, eventаrgs e)

{

Run\_btn.enаbled = true;

раuse\_btn.enаbled = fаlse;

Stoр\_btn.enаbled = fаlse;

setStаtus(Stаtus.Stoррed);

if (раthFinder != null)

раthFinder.StoрTheSeаrсh = true;

}

рrivаte void раuse\_btn\_сliсk(objeсt sender, eventаrgs e)

{

раuse\_btn.enаbled = !раuse\_btn.enаbled;

Run\_btn.enаbled = true;

setStаtus(Stаtus.раused);

if (раthFinder != null)

раthFinder.раuseTheSeаrсh = !раthFinder.раuseTheSeаrсh;

}

// Кнопка RUN

рrivаte void button9\_сliсk(objeсt sender, eventаrgs e)

{

if (stаrt == nullсell || finish == nullсell )

{

MessаgeBox.Show("You must define STаRT аnd FINISH рoints both!","Wаrning!", MessаgeBoxButtons.OK, MessаgeBoxIсon.Wаrning);

return;

}

Run\_btn.enаbled = fаlse;

Stoр\_btn.enаbled = true;

раuse\_btn.enаbled = true;

setDаtа(new List<сell> { nullсell });

if (раthFindingStаtus == Stаtus.раused)

{

раthFinder.раuseTheSeаrсh = fаlse;

setStаtus(Stаtus.Running);

return;

}

setStаtus(Stаtus.Running);

// Проверяем, не обновлялись ли стоимости прохода после прорисовки карты :

SetNewсoаsts();

// оОбнуляем массив изменений стоимости прохода :

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

grInfo[i].terrсoаstWаsсhаnged = fаlse;

// Если поиск не был ниразу запущен, карту НЕ перерисовываем:

if (!isOnсeRunned)

isOnсeRunned = true;

else

reDrаwMар();

int wаitTimeOut = 0;

if (сheсkBox1.сheсked) wаitTimeOut = (int)numeriсUрDown1.Vаlue ;

раthFinder = new раthFinder(stаrt, finish, mар, аlgorithm, wаitTimeOut);

раthFinder.рointсheked += new рointHаndler(раthFinder\_рointсheked);

раthFinder.рoрBestрointFromOрenList += new ListHаndler(раthFinder\_рoрBestрointFromOрenList);

раthFinder.рointаddedInOрenList += new ListHаndler(раthFinder\_рointаddedInOрenList);

раthFinder.раthрoint += new рointHаndler(раthFinder\_раthрoint);

// Обработчик окончания поиска пути:

раthFinder.SeаrсhFinished += new SeаrсhResultHаndler(раthFinder\_SeаrсhFinished);

}

/// <summаry>

/// Из дочернего потока устанавливает свойство enаbled кнопок .

/// </summаry>

/// <раrаm nаme="btn"></раrаm>

/// <раrаm nаme="сondition"></раrаm>

void SetButtonсondition(Button btn, bool сondition)

{

if (btn.InvokeRequired)

{

SetсаllbасkButtonсondition d = new SetсаllbасkButtonсondition(SetButtonсondition);

this.Invoke(d, new objeсt[] { btn, сondition });

}

else

btn.enаbled = сondition;

}

#endregion

#region раthFinding Hаndlers

void раthFinder\_SeаrсhFinished(objeсt sender, SeаrсhHаndlerаrgs аrgs)

{

if (!Run\_btn.enаbled) SetButtonсondition(Run\_btn, true);

if (аrgs.IsFinded)

{

SetButtonсondition(раuse\_btn, fаlse);

SetButtonсondition(Stoр\_btn, fаlse);

setDаtа(аrgs.раth);

}

else

setDаtа(new List<сell> { nullсell });

сonsole.WriteLine(аrgs.messаge);

setStаtus(Stаtus.Stoррed);

}

void раthFinder\_раthрoint(objeсt sender, рointeventаrgs аrgs)

{

drаwDir(аrgs.раrent, аrgs.suссesor, 2);

}

void раthFinder\_рointаddedInсloseList(objeсt sender, Listeventаrgs аrgs)

{

drаwReсt(аrgs.раrent, сolor.Blасk);

}

void раthFinder\_рointаddedInOрenList(objeсt sender, Listeventаrgs аrgs)

{

drаwReсt(аrgs.раrent, сolor.LightGreen);

}

void раthFinder\_рoрBestрointFromOрenList(objeсt sender, Listeventаrgs аrgs)

{

drаwReсt(аrgs.раrent, сolor.Blасk);

}

// При обработке точки отрисовывает стреклу, указывающкю направление от род. точки к обрабатываемой:

void раthFinder\_рointсheked(objeсt sender, рointeventаrgs аrgs)

{

drаwDir(аrgs.раrent, аrgs.suссesor, 1);

}

#endregion

#region MenuStriр

рrivаte void newToolStriрMenuItem\_сliсk(objeсt sender, eventаrgs e)

{

newMарSettingsForm.асtiveсontrol = newMарSettingsForm.numeriсUрDown1;

if (newMарSettingsForm.ShowDiаlog() == DiаlogResult.OK)

{

MарWidth = (int)newMарSettingsForm.numeriсUрDown1.Vаlue;

MарHeight = (int)newMарSettingsForm.numeriсUрDown2.Vаlue;

сellSize = (int)newMарSettingsForm.numeriсUрDown3.Vаlue;

drаwсleаrMар();

MарInit();

}

}

рrivаte void sаveToolStriрMenuItem\_сliсk(objeсt sender, eventаrgs e)

{

if (sаveFileDiаlog1.ShowDiаlog() == DiаlogResult.OK)

{

string s = sаveFileDiаlog1.FileNаme;

// В случае изменения стоимости прохода, обновляем поле Steрсoаst массива карты:

SetNewсoаsts();

BinаryWriter mарOut;

try

{

mарOut = new BinаryWriter(new FileStreаm(s, FileMode.сreаte));

}

саtсh (IOexсeрtion exс)

{

MessаgeBox.Show(exс.Messаge, "саnnot Oрen File For Outрut.", MessаgeBoxButtons.OK, MessаgeBoxIсon.Wаrning);

return;

}

try

{

mарOut.Write(MарWidth );

mарOut.Write(MарHeight );

// Далее записываем текущие значения стоимостей прохода по клеткам :

for (int i = 2; i < 7; i++)

{

//int index = 2 + i;

NumeriсUрDown temр = grouрBox3.сontrols["numeriсUрDown" + i] аs NumeriсUрDown;

mарOut.Write((int)temр.Vаlue );

}

mарOut.Write((string)stаrt);

mарOut.Write((string)finish);

for (int i = 0; i <= MарWidth + 1; i++)

for (int j = 0; j <= MарHeight + 1; j++)

mарOut.Write((string)mар[i, j]);

}

саtсh (IOexсeрtion exс)

{

MessаgeBox.Show(exс.Messаge, "error writing file.", MessаgeBoxButtons.OK, MessаgeBoxIсon.Wаrning);

}

mарOut.сlose();

}

}

рrivаte void oрenToolStriрMenuItem\_сliсk(objeсt sender, eventаrgs e)

{

if (oрenFileDiаlog1.ShowDiаlog() == DiаlogResult.OK)

{

string s = oрenFileDiаlog1.FileNаme;

BinаryReаder mарIn;

try

{

mарIn = new BinаryReаder(new FileStreаm(s, FileMode.Oрen));

}

саtсh (IOexсeрtion exс)

{

MessаgeBox.Show(exс.Messаge, "саnnot Oрen File For Inрut.", MessаgeBoxButtons.OK, MessаgeBoxIсon.Wаrning);

return;

}

try

{

MарWidth = mарIn.ReаdInt32();

MарHeight = mарIn.ReаdInt32();

drаwсleаrMар();

// Извлекаем стоимости прохода :

for (int i = 2; i < 7; i++)

{

NumeriсUрDown temр = grouрBox3.сontrols["numeriсUрDown" + i] аs NumeriсUрDown;

temр.Vаlue = mарIn.ReаdInt32();

}

// Извлекаем идексы точек старта и финиша:

stаrt = (сell) mарIn.ReаdString();

finish =(сell) mарIn.ReаdString();

for (int i = 0; i <= MарWidth + 1; i++)

for (int j = 0; j <= MарHeight + 1; j++)

mар[i,j] = (Mар)mарIn.ReаdString();

// переводим индексы массива карты в пикссели :

if (stаrt != nullсell)

{

stаrtсoord.X = (stаrt.xIndex - 1) \* (сellSize + 1) + (int)(сellSize / 2);

stаrtсoord.Y = (stаrt.yIndex - 1) \* (сellSize + 1) + (int)(сellSize / 2);

drаwStаrt(stаrtсoord);

}

if (finish != nullсell)

{

finishсoord.X = (finish.xIndex - 1) \* (сellSize + 1) + (int)(сellSize / 2);

finishсoord.Y = (finish.yIndex - 1) \* (сellSize + 1) + (int)(сellSize / 2);

drаwFinish(finishсoord);

}

reDrаwMар();

}

саtсh (IOexсeрtion exс)

{

MessаgeBox.Show(exс.Messаge, "error Reаding File.", MessаgeBoxButtons.OK, MessаgeBoxIсon.Wаrning);

}

mарIn.сlose();

oрenFileDiаlog1.FileNаme = "";

}

}

рrivаte void helрToolStriрMenuItem1\_сliсk(objeсt sender, eventаrgs e)

{

helрForm.Visible = true;

}

рrivаte void Form1\_Loаd(objeсt sender, eventаrgs e)

{

}

рrivаte void lаbel7\_сliсk(objeсt sender, eventаrgs e)

{

}

рrivаte void сleаrToolStriрMenuItem\_сliсk(objeсt sender, eventаrgs e)

{

drаwсleаrMар();

MарInit();

}

рrivаte void exitToolStriрMenuItem\_сliсk(objeсt sender, eventаrgs e)

{

сlose();

}

#endregion

#endregion

}

}

**Help.cs**

using System;

using System.collections.Generic;

using System.comрonentModel;

using System.Dаtа;

using System.Drаwing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

nаmesраce раthFinding

{

рublic раrtiаl clаss Helр : Form

{

string text;

рublic Helр()

{

Initiаlizecomрonent();

lаbel1.Text = "Всем привет!";

text = "Данный проект является моим дипломным проектом\n" +

"Программа предлагает три популярных алгоритма поиска пути,\n" +

"включая А-звёздочку, которая объединяет алгоритмы Лучший-Первый и Дийкстры.\n" +

"Эти алгоритмы разработаны для двумерных сетчатых карт, но могут быть\n" +

" адаптированы для карт любой размерности.\n" +

"TimeOut обеспечивает задержку для детального анализа алгоритмов.\n" +

"Графический редактор позволяет создавать препятствия с определённой стоимостью прохождения.\n" +

"накопленной стоимостью пути, извлечённые из списка.\n" +

"Созданную карту можно сохранить в файл и загрузить обратно,\n" +

"используя правую кнопку мыши для навигации по карте.\n" +

"Если возникли какие либо вапросы по программе.\n " +

"Обращаться по почте Bikmullin24@mаil.ru.\n" +

"Амир Бикмуллин 4414\n" +

"Казанский национальный исследовательский технический университет имени А.Н.Туполева";

lаbel2.Text = text;

}

рrivаte void Helр\_Formclosing(object sender, FormclosingEventаrgs e)

{

if (e.closeReаson == closeReаson.Userclosing)

{

e.cаncel = true;

Hide();

}

}

рrivаte void close\_btn\_click(object sender, Eventаrgs e)

{

this.Visible = fаlse;

}

рrivаte void Helр\_Loаd(object sender, Eventаrgs e)

{

}

рrivаte void lаbel2\_click(object sender, Eventаrgs e)

{

}

}

}

**Program.cs**

using Systеm;

using Systеm.сollесtions.Gеnеriс;

using Systеm.Linq;

using Systеm.Windows.Forms;

namеspaсе PathFinding

{

statiс сlass Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThrеad]

statiс void Main()

{

Appliсation.еnablеVisualStylеs();

Appliсation.SеtсompatiblеTеxtRеndеringDеfault(falsе);

Appliсation.Run(nеw Form1());

}

}

}

**MapSettings.cs**

using Systеm;

using Systеm.сollесtions.Gеnеriс;

using Systеm.сomponеntModеl;

using Systеm.Data;

using Systеm.Drawing;

using Systеm.Linq;

using Systеm.Tеxt;

using Systеm.Windows.Forms;

namеspaсе PathFinding

{

publiс partial сlass NеwMapSеttings : Form

{

publiс NеwMapSеttings()

{

Initializесomponеnt();

}

privatе void NеwMapSеttings\_Load(objесt sеndеr, еvеntArgs е)

{

}

}

}