Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕНОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ

«Санкт-Петербургский государственный университет   
промышленных технологий и Дизайна»

Институт информационных технологий и автоматизации

Специальность (направление) 09.03.03 Прикладная информатика (по областям)

Кафедра Информационных систем и компьютерного дизайна

КУРСОВАЯ РАБОТА

(курсовой проект)

на тему Разработка интерактивного графического приложения «Лабиринт»

по дисциплине Алгоритмизация и программирование

|  |  |  |
| --- | --- | --- |
| Исполнитель – обучающийся учебной группы | | 2-МВ-4 |
|  | | (группа) |
| Белоцерковский Дмитрий Сергеевич | | |
| (фамилия, имя, отчество, подпись) | | |
| Руководитель курсовой работы |  | |
| Бармина Анастасия Александровна | | |
| (ученая степень, звание, фамилия, имя, отчество, подпись) | | |

Санкт-Петербург

2024

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕНОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ

«Санкт-Петербургский государственный университет   
промышленных технологий и Дизайна»

|  |  |
| --- | --- |
| Утверждаю | Институт информационных  технологий и автоматизации |
| Заведующий кафедрой |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. В. Сошников  «\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. | Кафедра Информационных систем и компьютерного дизайна |

**ЗАДАНИЕ**

**на курсовую работу (проект)**

Студенту Белоцерковскому Дмитрию Сергеевичу

1. Тема курсовой работы (проекта)

Разработка интерактивного графического приложения «Лабиринт»

2. Срок сдачи студентом законченной курсовой работы (проекта)

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

3. Исходные данные по курсовой работе (проекту)

Компьютерные игры, программирование, языки программирование, C++

4. Перечень подлежащих разработке в курсовой работе (проекте) вопросов или краткое содержание работы

1. Введение

2. Цель

3. Используемые программные средства

4. Описание игры

5. Разработка игры

6. Заключение

7. Список литературы

8. Приложения

6. Дата выдачи задания \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись)

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

(дата) (подпись студента)

РЕФЕРАТ

Курсовая работа: 22 страницы, 9 рисунков, 8 источников, 1 приложение.

СОЗДАНИЕ КОМПЬЮТЕРНОЙ ИГРЫ, ИГРА ЛАБИРИНТ, МИНИ-ИГРА, C++, SDL, VISUAL STUDIO, ОФЛАЙН-ИГРА

Цель работы – создание компьютерной игры при помощи языка программирования C++ и библиотеки SDL.

Проект был написан при помощи программного обеспечения Microsoft Visual Studio, языка программирования общего назначения С++ и мультимедийной библиотеки SDL. В процессе написания работы были также рассмотрены функционал, преимущества и недостатки вышеперечисленных продуктов.

Конечным продуктом курсовой работы стала игра «Лабиринт» в жанре «Лабиринт».

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc165846211)

[1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 5](#_Toc165846212)

[1.1 Microsoft Visual Studio 5](#_Toc165846213)

[1.2 Язык программирования C++ 7](#_Toc165846214)

[1.3 Мультимедийная библиотека SDL 11](#_Toc165846215)

[2 ПРАКТИЧЕСКАЯ ЧАСТЬ 13](#_Toc165846216)

[2.1 Постановка задач 13](#_Toc165846217)

[2.2 Создание игры 13](#_Toc165846218)

[ЗАКЛЮЧЕНИЕ 20](#_Toc165846219)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 21](#_Toc165846220)

[ПРИЛОЖЕНИЕ А 22](#_Toc165846221)

ВВЕДЕНИЕ

В более широком смысле под программированием понимают весь спектр деятельности, связанный с созданием и поддержанием в рабочем состоянии программ – программного обеспечения ЭВМ. Иначе это называется «программная инженерия». В неё входят анализ и постановка задачи, проектирование программы, построение алгоритмов, разработка структур данных, написание текстов программ, отладка и тестирование программы, документирование, настройка, доработка и сопровождение.

Все вышеперечисленное важно и для поэтапного создания игры. Также для реализации идеи нужны сторонние средства, например, мультимедийная библиотека Simple DirectMedia Layer или кратко SDL, что упрощает разработку игр и мультимедийных приложений.

В данной курсовой работе были разобраны основные функции программы Microsoft Visual Studio, достоинства и недостатки, основы языка программирования С++, методы и функции, которые могут помочь в разработке приложений и игр, а также приведена информация о самой мультимедийной библиотеке.

В практической части была разработана сама игра. Игра написана на базе среды разработки Visual Studio.

1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

* 1. Microsoft Visual Studio

Microsoft Visual Studio – это узкоспециализированная линейка программных продуктов от компании Microsoft, которая включает в себя интегрированную среду разработка программного обеспечения и множество других средств для разработчиков.

Visual Studio ориентированно на разработку приложений, причем в Visual Studio можно создавать как консольные приложения, так и приложения с поддержкой графического интерфейса, и с поддержкой технологии Windows Forms.

Помимо этого в Visual Studio можно разрабатывать веб-сайты, веб-приложения, веб-службы, при этом разработанные приложения могут быть для любой платформы, которую поддерживают Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Microsoft Silverlight.

Мощнейший отладчик кода может работать как с машинным, так и с исходным кодами. Также Visual Studio предоставляет множество других инструментов для работы: многофункциональный редактор форм для создания графического интерфейса приложения, мощный веб-редактор, инструмент дизайна классов и инструмент дизайна схем баз данных.

Visual Studio обладает возможностью использовать множество плагинов для расширения функциональных возможностей системы практически на каждом уровне работы, включая подключение систем контроля версий исходного кода, таких как Perforce, Subversion и Visual SourceSafe, также можно добавить огромное множество новых инструментов для работы с визуальным проектированием программных продуктов.

Visual Studio включает один или несколько компонентов из следующих:

1. [Visual Basic .NET](https://ru.wikipedia.org/wiki/Visual_Basic_.NET), а до его появления – Visual Basic
2. [Visual C++](https://ru.wikipedia.org/wiki/Microsoft_Visual_C%2B%2B)
3. [Visual C#](https://ru.wikipedia.org/wiki/C_Sharp)(включён начиная с Visual Studio .NET)
4. [Visual F#](https://ru.wikipedia.org/wiki/F_Sharp)(включён начиная с Visual Studio 2010)
5. JavaScript
6. Python (включён начиная с Visual Studio 2019)
7. TypeScript
8. XAML

Многие варианты поставки также включают:

1. [Microsoft SQL Server](https://ru.wikipedia.org/wiki/Microsoft_SQL_Server) либо [Microsoft SQL Server Express](https://ru.wikipedia.org/wiki/Microsoft_SQL_Server#SQL_Server_Express_Edition)

В прошлом в состав Visual Studio также входили продукты:

1. [Visual InterDev](https://ru.wikipedia.org/wiki/Visual_InterDev)
2. [Visual J++](https://ru.wikipedia.org/wiki/Visual_J%2B%2B)
3. [Visual J#](https://ru.wikipedia.org/wiki/Visual_J_Sharp)
4. [Visual FoxPro](https://ru.wikipedia.org/wiki/Visual_FoxPro)
5. [Visual Source Safe](https://ru.wikipedia.org/wiki/Microsoft_Visual_SourceSafe) – файл-серверная [система управления версиями](https://ru.wikipedia.org/wiki/Система_управления_версиями).

Visual Studio предоставляет первоклассные возможности Git и GitHub. Такие функции, как проверка подлинности, клонирование и создание новых репозиториев, встроены в Visual Studio, что упрощает начало работы с Git и GitHub. Больше не нужно полагаться на внешние инструменты для управления системой управления версиями или быть экспертом по Git, чтобы иметь возможность использовать Git и GitHub в Visual Studio.

* 1. Язык программирования C++

C++ – это язык объектно-ориентированного программирования общего назначения (ООП), разработанный Бьярном Страуструпом и являющийся расширением языка Си. Следовательно, можно производить разработку программного обеспечения C++ в стиле Cи или в объектно-ориентированном стиле. В определенных сценариях на нем можно писать код любым способом и, таким образом, он является эффективным примером гибридного языка.

С++ считается языком промежуточного уровня, так как он включает в себя функции языка высокого и низкого уровня. Первоначально язык назывался «C с классами», так как он имел все свойства языка C с дополнительным понятием «классы». Тем не менее, он был переименован в C++ в 1983 году.

C ++ – один из самых популярных языков, в основном используемый с системным/прикладным программным обеспечением, драйверами, клиент – серверными приложениями и встроенным программным обеспечением.

Типы в С++:

1. Символьные: char, wchar\_t (char16\_t и char32\_t, в стандарте C++ 11).
2. Целочисленные знаковые: signed char, short int, int, long int (и long long int, в стандарте C++11).
3. Целочисленные беззнаковые: unsigned char, unsigned short int, unsigned int, unsigned long int (и unsigned long long int, в стандарте C++11).
4. С плавающей точкой: float, double, long double.
5. Логический: bool.

Операции сравнения возвращают тип bool. Выражения в скобках после if, while приводятся к типу bool.

Функции могут принимать аргументы по ссылке. Функции могут возвращать результат по ссылке. Ссылки сходны с указателями, со следующими особенностями: перед использованием ссылка должна быть инициализирована; ссылка всегда указывает на один и тот же адрес; в выражении ссылка обозначает непосредственно тот объект или ту функцию, на которую она указывает, обращение же к объекту или функции через указатель требует разыменование указателя.

C++ добавляет к C объектно-ориентированные возможности. Он вводит классы, которые обеспечивают три самых важных свойства ООП: инкапсуляцию, наследование и полиморфизм.

Методы класса – это функции, которые смогут применяться к экземплярам класса. Грубо говоря, метод – это функция, объявленная внутри класса и предназначенная для работы с его объектами. Методы объявляются в теле класса. Описываться могут там же, но могут и за пределами класса (внутри класса в таком случае достаточно представить прототип метода, а за пределами класса определять метод поставив перед его именем — имя класса и оператор ::). Методы и поля входящие в состав класса называются членами класса. При этом методы часто называют функциями-членами класса.

В C++ при наследовании одного класса от другого наследуется реализация класса, плюс класс-наследник может добавлять свои поля и функции или переопределять функции базового класса. Множественное наследование разрешено.

Конструктор наследника вызывает конструкторы базовых классов, а затем конструкторы нестатических членов-данных, являющихся экземплярами классов. Деструктор работает в обратном порядке.

Наследование бывает публичным, защищённым и закрытым.

Целью полиморфизма, применительно к объектно-ориентированному программированию, является использование одного имени для задания общих для класса действий. Выполнение каждого конкретного действия будет определяться типом данных.

Преимуществом полиморфизма является то, что он помогает снижать сложность программ, разрешая использование того же интерфейса для задания единого класса действий. Выбор же конкретного действия, в зависимости от ситуации, возлагается на компилятор. Полиморфизм может применяться также и к операторам.

Основным способом организации информации в C++ являются классы. В отличие от структуры (struct) языка C, которая может состоять только из полей и вложенных типов, класс (class) C++ может состоять из полей, вложенных типов и функций-членов. Инкапсуляция в С++ реализуется через указание уровня доступа к членам класса: они бывают публичными (public), защищёнными (protected) и закрытыми (private). В C++ структуры отличаются от классов тем, что по умолчанию члены и базовые классы у структуры публичные, а у класса – собственные.

В языке программирования C++ термин Стандартная Библиотека означает коллекцию классов и функций, написанных на базовом языке. Стандартная Библиотека поддерживает несколько основных контейнеров, функций для работы с этими контейнерами, объектов-функции, основных типов строк и потоков (включая интерактивный и файловый ввод-вывод), поддержку некоторых языковых особенностей, и часто используемые функции для выполнения таких задач, как, например, нахождение квадратного корня числа. Стандартная Библиотека языка C++ также включает в себя спецификации стандарта ISO C90 стандартной библиотеки языка Си. Функциональные особенности Стандартной Библиотеки объявляются внутри пространства имен std.

[Стандартная библиотека шаблонов](https://ru.bmstu.wiki/index.php?title=Стандартная_библиотека_шаблонов&action=edit&redlink=1) (STL) – подмножество стандартной библиотеки C++ и содержит контейнеры, алгоритмы, итераторы, объекты-функции и т. д.

Заголовочные файлы стандартной библиотеки C++ не имеют расширения «.h».

Стандартная библиотека C++ содержит последние расширения C++ стандарта ANSI (включая библиотеку стандартных шаблонов и новую библиотеку iostream). Она представляет собой набор файлов заголовков.

Отличия от Си:

1. поддержка объектно-ориентированного программирования;
2. поддержка обобщённого программирования через шаблоны;
3. дополнительные типы данных;
4. исключения;
5. пространства имён;
6. встраиваемые (inline) функции;
7. перегрузка операторов;
8. перегрузка функций;
9. ссылки и операторы управления свободно распределяемой памятью;
10. дополнения к стандартной библиотеке.

В С++ не разрешается:

1. вызывать функцию main() внутри программы, в то время как в C это действие правомерно.
2. неявное приведение типов между несвязанными типами указателей.
3. использовать функции, которые ещё не объявлены.

Достоинства:

1. Высокая совместимость с языком Си
2. Вычислительная производительность
3. Поддержка различных стилей программирования: структурное, объектно-ориентированное, обобщённое программирование, функциональное программирование, порождающее мета-программирование.
4. Автоматический вызов деструкторов объектов (в порядке обратном вызову конструкторов) упрощает и повышает надёжность управления памятью и другими ресурсами (открытыми файлами, сетевыми соединениями, т. п.).
5. Перегрузка операторов
6. Шаблоны (дают возможность построения обобщённых контейнеров и алгоритмов для разных типов данных)
7. Возможность расширения языка для поддержки парадигм, которые не поддерживаются компиляторами напрямую
8. Доступность. Для С++ существует огромное количество учебной литературы, переведённой на всевозможные языки.

Недостатки:

1. Плохо продуманный синтаксис сужает спектр применимости языка
2. Язык не содержит многих важных возможностей
3. Язык содержит опасные возможности
4. Производительность труда программистов на языке оказывается неоправданно низка
5. Громоздкость синтаксиса
6. Тяжелое наследие
7. Необходимость следить за памятью.

1.3 Мультимедийная библиотека SDL

SDL – свободная кроссплатформенная мультимедийная библиотека, реализующая единый программный интерфейс к графической подсистеме, звуковым устройствам и средствам ввода для широкого спектра платформ. Данная библиотека активно используется при написании кроссплатформенных мультимедийных программ (в основном игр).

Официально поддерживаемые операционные системы: Linux, Microsoft Windows, macOS, iOS и Android. SDL API доступны для языков: C, C++, C#, VB.NET, D, Ada, Vala, Eiffel, Haskell, Erlang, Euphoria, Java, Lisp, Lua, ML, Pascal, Perl, PHP, Pike, PureBasic, Python, Go, Rust и Ruby.

Основная часть SDL содержит базовый, весьма ограниченный спектр возможностей. Дополнительную функциональность обеспечивают библиотеки расширений, которые обычно входят в поставку SDL. Начиная с версии 2.0, SDL распространяется под условиями лицензии zlib; до этого использовалась лицензия GNU LGPL.

SDL создал Сэм Лантинга, будучи ведущим программистом компании Loki Entertainment Software. История SDL начинается со времён, когда Сэм работал над программой-эмулятором Microsoft Windows для Macintosh. Он заметил, что многие куски кода без проблем могут работать на Linux. И он решил создать небольшую библиотеку, чтобы ей могли воспользоваться другие программисты. В октябре 1997 был выпущен релиз библиотеки с версией 0.3, а позже эту библиотеку вовсю использовала компания Loki Software для портирования игр под Linux.

SDL сам по себе довольно прост. Его можно рассматривать как тонкую прослойку, обеспечивающую поддержку для 2D-операций над пикселями, звука, доступа к файлам, обработки событий и т. п. Он часто используется в дополнение к OpenGL, предоставляя поддержку мыши, клавиатуры и джойстиков.

Библиотека состоит из нескольких подсистем, таких как Video, Audio, CD-ROM, Joystick и Timer. В дополнение к этой базовой низкоуровневой функциональности, существует ряд стандартных библиотек, предоставляющих дополнительную функциональность:

* SDL\_image — поддержка различных растровых форматов
* SDL\_mixer — функции для организации сложного аудио, в основном, сведение звука из нескольких источников
* SDL\_net — поддержка сетевых функций
* SDL\_ttf — поддержка шрифтов TrueType
* SDL\_rtf — отрисовка текста в формате RTF (доступна только для SDL 1.2)

2 ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1 Постановка задач

Чтобы достичь цели курсовой работы необходимо было поставить перед собой следующие задачи:

1. Выработать алгоритм.
2. Написать код.
3. Отладить работу кода.

2.2 Создание игры

В качестве игры была выбрана тема игры «Лабиринт», в жанре «Лабиринт».

Код подробно сопровождён комментариями, и начинается с подключения библиотек, для обеспечения работоспособности различного функционала.

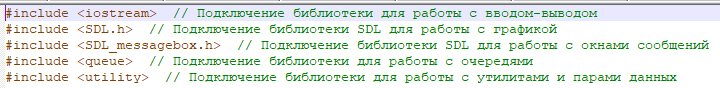


Рисунок 1 – подключение библиотек

**Класс Grid**

Далее идёт класс игрового поля «Grid», и его public раздел начинается с перечисления статических констант

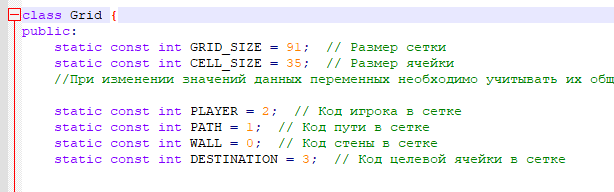


Рисунок 2 – класс Grid, статические константы.

Они определяют масштаб и пропорции игрового поля и переменные из логики перемещений игрока.

Далее идёт ряд необходимых функций для того, чтобы отрисовка игрового поля не требовала SDL\_Image и оперирования растровыми объектами.

Нам понадобится генератор случайных чисел для добавления доли случайности в величину отклонения от опорной матрицы стен и тропинок, вызов функции генерации новой матрицы поля, и установки стартовой позиции игрока.

При движении также понадобится проверка на то, что игрок не ходит по стенам, поэтому мы ставим булевую проверку на действительность тропинки.

Сама функиця генерации новой матрицы поля представляет из себя копипаст новой сетки из статической матрицы опорной, вызов функции проверки на гарантированность пути до финиша, последующий вызов функции перерисовки стен, и функцию смены позиции игрока.

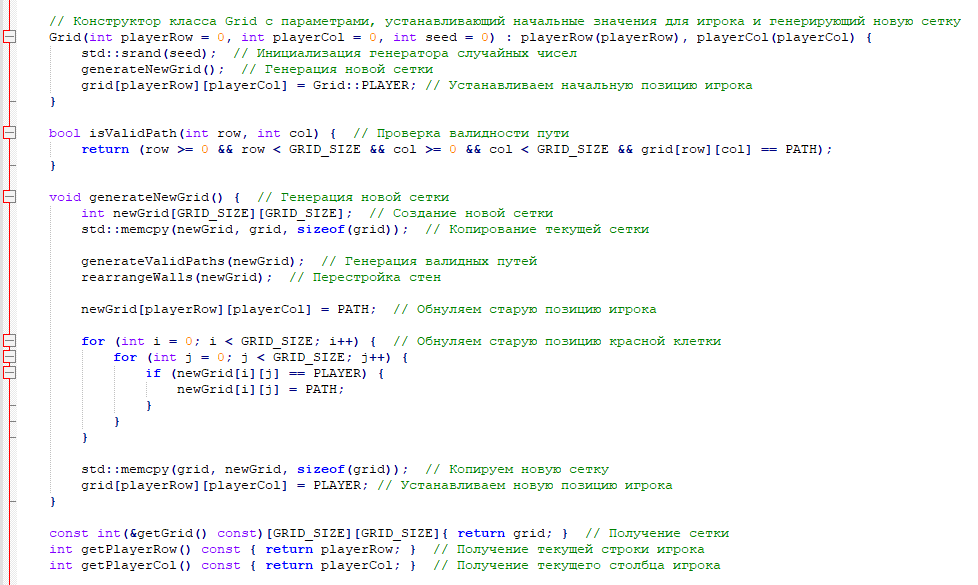


Рисунок 3 – различные функции отрисовки поля

Далее в коде идёт статическая опорная матрица.



Рисунок 4 – опорная матрица игрового поля.

Завершают класс Grid войды о генерации гарантированных путей до финиша и перестройки стен.

Также, для упрощения генерации уровня, было применено эмпирически подобранное отклонение для величины коэффициента тропинкостенозаменяемости (в функции rearrangeWalls), позволяющее сгенерировать визуально кажущийся отличающимся, и, достаточно непустой (на стены), новый уровень, целиком из статического опорного старого.

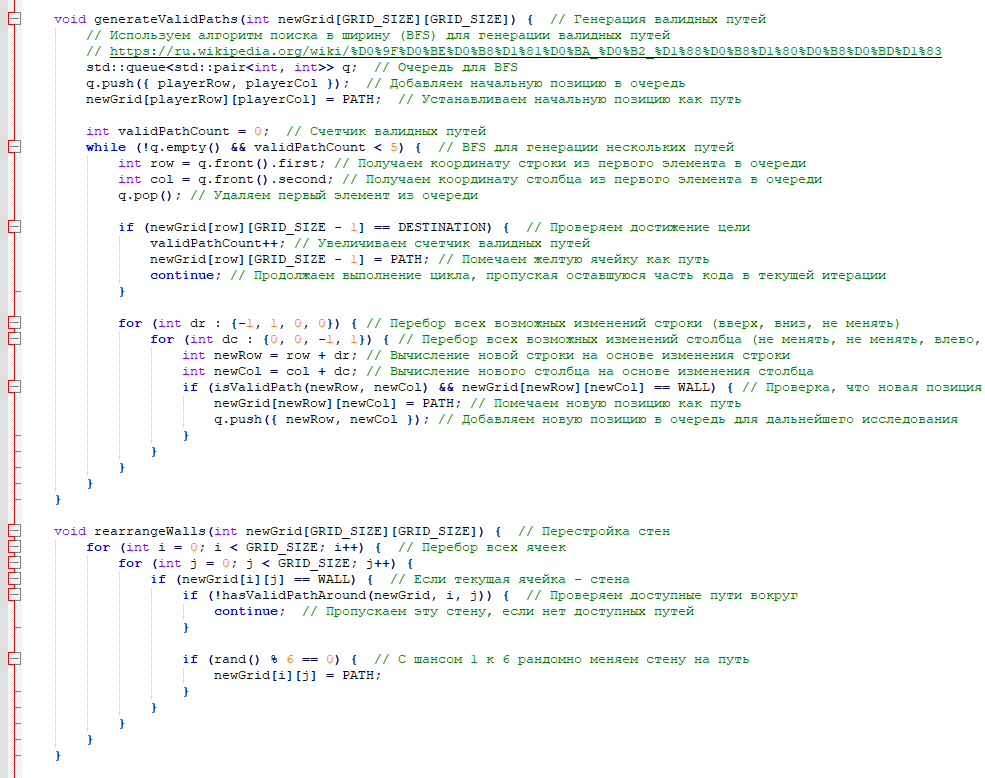


Рисунок 5 – функции гарантированных путей и перестройки стен

Ну, и функция, чтобы нельзя было ходить в стену.

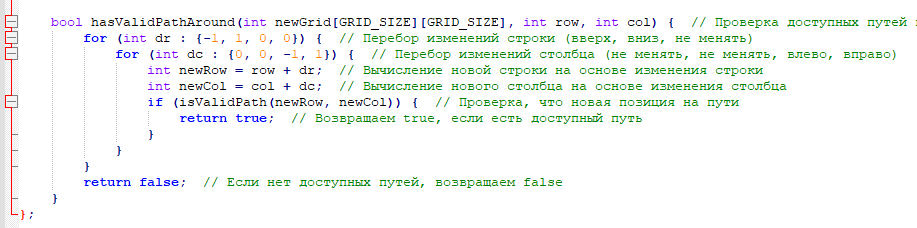


Рисунок 6 – функция проверки допустимости пути

**Класс Player**

Следующим идёт класс игрока, в котором описаны публичные константы и функции, такие как различные вспомогательные константы для управления ходом игрока, счётчика уровней игры, позиции игрока в двух осях, как её запроса, так и установки, войд для перемещения игрока, проверка выхода за границы, проверка достижения условия победы, инкремента уровня при достижении, вывода сообщения о победе с последующей выгрузкой ресурсов и дестроем рендера, в случае полной победы в игре, и проверкой на допустимый путь с обновлением позиции игрока, в случае её прохождения.

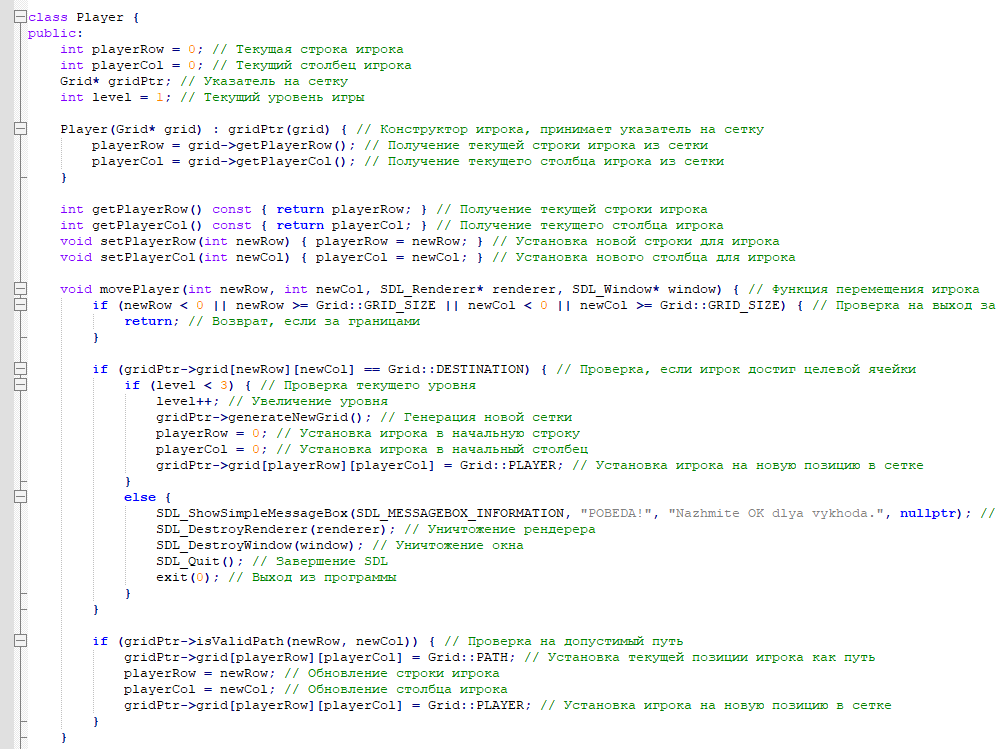


Рисунок 7 – класс игрока.

**Класс Renderer**

Далее, идёт класс рендера.

В классе рендера собраны различные SDL-операторы для вызова SDL-функций, таких как создание и отрисовка окна, установка цвета, размера, очистка рендера, для каждого из объектов (тропинка, стена, игрок, финиш), тильда-деструктор.

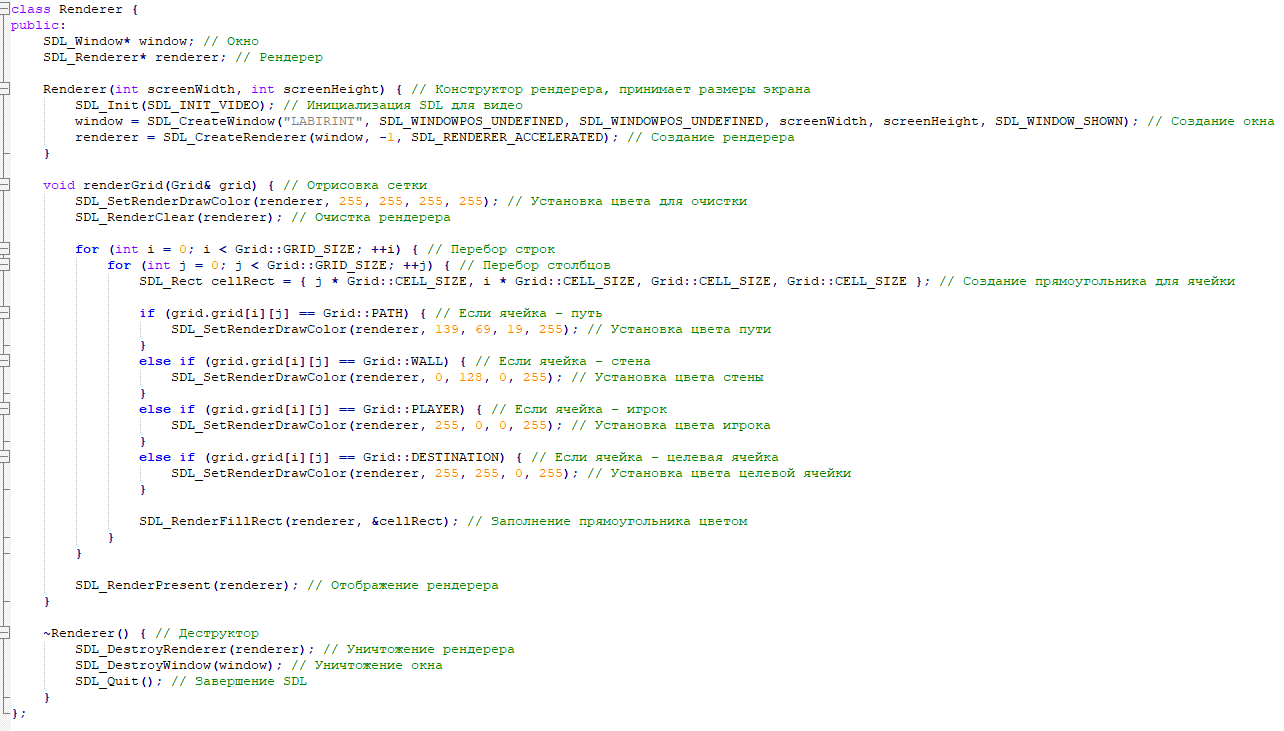


Рисунок 8 – класс рендера

**Основной цикл программы**

В конце, после описания всех классов, идёт основной цикл, в котором перечислены все вероятно могущие быть не повторёнными одинаково события, а именно – события нажатия клавиш при управлении персонажа игроком, в которых цепным образом вызываются все остальные, требуемые для игры функции, как то, отрисовки окна, смены позиции игрока и флага выхода из игры.

Отдельно стоит заметить, что в игре присутствует распределённая связность статического масштабирования игрового поля, опирающаяся на величины группы констант размеров пикселов и количества клеток поля, пропорционально рассчитываемых для сохранения логики построения поля вида, приемлемого для работоспособности игры. При изменении данных величин без учёта их связности, нарушится условия корректной отрисовки, и ожидаемого масштабирования не случится, для чего во всех связных местах в коде, оставлен соответствующий комментарий.

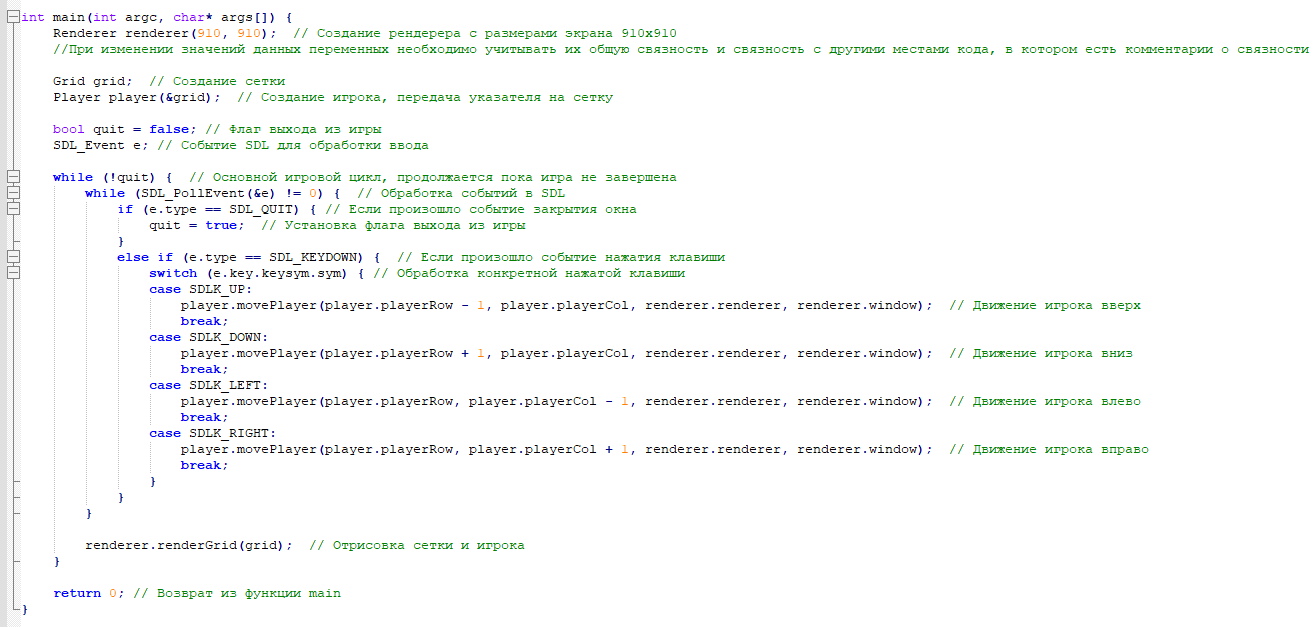


Рисунок 9 – основной цикл программы

ЗАКЛЮЧЕНИЕ

В процессе написания курсовой работы были повторно закреплены знания о языке программирования С++, появились знания о мультимедийной библиотеке под названием SDL.

Для практической части работы была создана игра «Лабиринт» в жанре «Лабиринт».

В реализации кода были использованы различные функции и методы: классы, булевые функции, счетчики, циклы, закрытия, открытия окна, свитчи, кейсы, воиды и так далее.

Всё это в совокупности помогло реализовать идею и написать игру «Лабиринт».

Билд и исходный код были направлены в личном кабинете на портале.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Microsoft Visual Studio. [Электронный ресурс] – 2024. – URL: http://www.kogorta.ru/microsoftvisualstudio (Дата обращения 08.03.2024)
2. Microsoft Visual Studio. [Электронный ресурс] – 2024. – URL: https://ru.wikipedia.org/w/index.php?title=Microsoft\_Visual\_Studio&stable=1 (Дата обращения 08.03.2024)
3. Обучение кодированию в Visual Studio. [Электронный ресурс] – 2024. – URL: https://visualstudio.microsoft.com/ru/vs/getting-started (Дата обращения 08.03.2024)
4. Программирование на С/С++. [Электронный ресурс] – 2024. – URL: https://roi4cio.com/categories/category/programmirovanie-na-cc/ (Дата обращения 08.03.2024)
5. Что такое C++. [Электронный ресурс] – 2024. – URL: http://web.spt42.ru/index.php/chto-takoe-c-plus-plus (Дата обращения 08.03.2024)
6. Язык программирования С++. [Электронный ресурс] – 2024. – URL: https://www.geeksforgeeks.org/operators-c-c/ (Дата обращения 08.03.2024)
7. Статья о мультимедийной библиотеке SDL. [Электронный ресурс] – 2024. – URL: https://ru.wikipedia.org/w/index.php?title=SDL&stable=1 (Дата обращения 08.03.2024)
8. Официальный сайт библиотеки SDL. [Электронный ресурс] – 2024. – URL: https://www.libsdl.org/ (Дата обращения 08.03.2024)

ПРИЛОЖЕНИЕ А

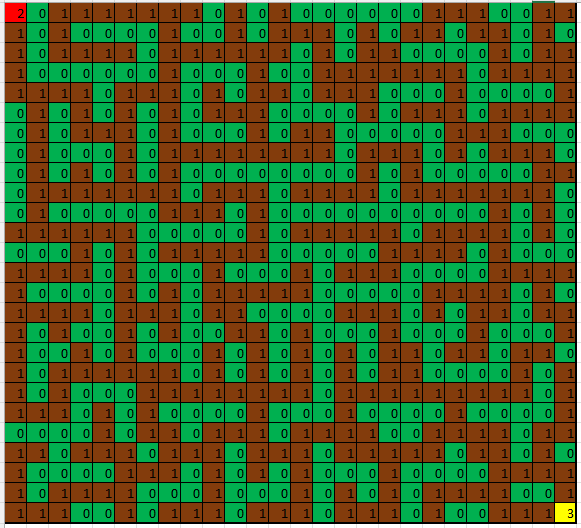


Рисунок А – схематическое наложение поля игры на её опорную матрицу, для наглядности способа генерации тропинок, стен, игрока и финиша