МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

учреждение образования

«Гродненский государственный университет имени Янки Купалы»

Факультет математики и информатики

**Кафедра современных технологий программирования**

ЛАТОШ ЕВГЕНИЙ АЛЕКСАНДРОВИЧ

**Игра «Pac-Man»**

Курсовая работа

по дисциплине «Основы алгоритмизации и программирования»

студента 1 курса специальности

1-40 01 01 «Программное обеспечение информационных технологий»

дневной формы получения образования

|  |  |
| --- | --- |
|  | Научный руководитель  Урбан Ольга Ивановна,  старший преподователь  кафедры современных технологий программирования |
|  |  |

Гродно, 2021

**РЕЗЮМЕ**

Латош Евгений Александрович

Курсовая работа – «Игра Pac-Man», 38 страниц, 31 иллюстраций, 4 использованных источников.

Ключевые слова – (алгоритм, библиотека, спрайт, привидение, игрок).

Цель исследования – разработка компьютерной игры «Pac-Man».

Объект исследования – алгоритмы поведения врагов, алгоритмы взаимодействия игрока с окружением, алгоритмы взаимодействия пользователя с интерфейсом игры.

Предмет исследования – реализация алгоритмов поведения врагов, взаимодействия игрока с окружением, взаимодействия пользователя с интерфейсом игры для создания компьютерной игры «Pac-Man».

Методы исследования – анализ алгоритмов оригинальной игры «Pac-Man» и синтез оптимальных алгоритмов для проекта.

**Авторская характеристика работы.**

Программа написана на языке высокого уровня С++. В ней реализованы общие алгоритмы функционирования игры. Основной задачей данной работы является создание законченного продукта, готового к использованию. Так же реализованные функции могут найти применение в создании новых игр или приложений.

**SUMMARY**

Latosh Yauheni Alexandrovich

The course work: «Pac-Man game», 38 pages, 31 pictures, 4 references used.

Keywords: algorithm, library, sprite, ghost, player.

The purpose of the research is the development of a computer game «Pac-Man».

Object of research: algorithms for the behavior of enemies, algorithms for the player's interaction with the environment, algorithms for user interaction with the game interface.

Subject of research: the implementation of algorithms for the behavior of enemies, player interaction with the environment, user interaction with the game interface to create a computer game «Pac-Man».

Research methods - analysis of algorithms of the original game «Pac-Man» and synthesis of optimal algorithms for the project.

**Author's characteristics of the work.**

The program is written in the high-level C ++ language. It implements general algorithms for the functioning of the game. The main task of this work is to create a finished product, ready to use. Also, the implemented functions can be used in the creation of new games or applications.

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ.................................................................................................................. 6

ГЛАВА 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ.................................……............. 8

1.1 История игры «Pac-Man»………………………………….…………............. 8

1.2 Обзор существующих решений….………………………………..…............. 9

1.3 Вывод по главе 1……………………………………………….……............. 11

ГЛАВА 2 ПРОЕКТИРОВАНИЕ ИГРЫ «PAC-MAN»……………..…………….. 12

2.1 Проектирование логики…………………………………………………….. 12

2.1.1 Алгоритм передвижения игрока…………………………….………..…….. 12

2.1.2 Алгоритм передвижения привидения…………………….……………..…. 12

2.1.3 Подсчет очков и алгоритм рекордов……………………….………………. 14

2.1.4 Алгоритм регистрации…………………………………….………………... 15

2.2 Проектирование взаимодействия пользователя с интерфейсом игры…… 16

2.2.1 Окно регистрации…………………………………………………………… 16

2.2.2 Главное меню………………………………………………………………... 17

2.2.3 Таблице рекордов…………………………………………………………… 18

2.2.4 Игровое меню……………………………………………………………….. 18

2.3 Вывод по главе 2…………………………………………….………………. 19

ГЛАВА 3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ИГРЫ «PAC-MAN»…….……….. 20

3.1 Программная реализация алгоритмов……………………………………... 20

3.1.1 Алгоритм передвижения игрока…………………………….………..…….. 20

3.1.2 Алгоритм передвижения привидения……………………….…………..…. 22

3.1.3 Подсчет очков и алгоритм рекордов……………………….………………. 25

3.1.4 Алгоритм регистрации……………………………………….……………... 26

3.2 Программная реализация взаимодействия пользователя с интерфейсом игры…………………………………………………………………………………. 27

3.2.1 Окно регистрации…………………………………………………………… 27

3.2.2 Главное меню………………………………………………………………... 30

3.2.3 Таблице рекордов…………………………………………………………… 32

3.2.4 Игровое меню……………………………………………………………….. 35

3.3 Вывод по главе 3…………………………………………….………………. 36

ЗАКЛЮЧЕНИЕ…………………………………………………….………………. 37

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ……………….……………… 38

**ВВЕДЕНИЕ**

Индустрия компьютерных игр - один из самых молодых и быстро развивающихся секторов новых медиа. Его экономическое значение было широко признано на региональном, национальном и международном уровнях.

Мировой рынок видеоигр быстро превратился из ниши в массовый рынок, обслуживающий различные игровые платформы, и все это лежит в основе постоянных технологических усовершенствований. На самом деле, сегодняшняя индустрия видеоигр далеко не экономически легковесна по сравнению с другими творческими отраслями, с глобальными доходами, соперничающими с доходами от производства фильмов или музыкальных изданий. Другие примеры многогранного спектра творческих индустрий включают литературу, рекламу или моду. Общим для этих отраслей является то, что по своей сути они включают нематериальные и символически насыщенные товары и услуги, одновременно смешивая коммерческие и культурные интересы.

Индустрия компьютерных игр неспокойна. Технологические циклы становятся короче, инвестиции в новые продукты сопряжены с риском, необходимость быстрого вывода на рынок огромна, а бизнес-модели быстро устаревают. Требуемой квалифицированной рабочей силы часто не хватает.

В данной курсовой работе были реализованы алгоритмы функционирования игры. При создании алгоритмов, я провел анализ алгоритмов оригинальной игры. В результате этого анализа были созданы более оптимальные алгоритмы, которые я использовал в своем проекте. Данные алгоритмы могут быть использованы при создании игр. Например, реализация передвижения объектов, создание меню, реализация поведения врагов и т.д. Также данные алгоритмы могут быть реализованы в различных приложениях. Например, в приложениях, которые используют регистрацию пользователей, рейтинг или таблицу рекордов. Также реализованные алгоритмы могут быть использованы при создании меню приложения.

Цель курсовой работы – разработка игры «Pac-Man». Игрок управляет круглым желтым существом, состоящим из рта и глаз. Задача игрока – собрать все точки на уровне. Уровень закончится, когда будут съедены все точки. Но в этом игроку будут мешать привидения, которые охотятся на него. В начале уровня, привидения находятся в центре экрана в небольшой комнате, откуда они выходят по одному, а одно привидение начинает уровень вне комнаты. Если игрок «съест» большую точку, то на следующие несколько секунд привидения становятся уязвимыми и могут быть съедены игроком. Но в это время привидения избегают встречи с ним. Съеденные в такой ситуации привидения не исчезают, но возвращаются в центр экрана, а игроку начисляются дополнительные очки. Если войти в коридор, ведущий направо или налево за границу экрана, то игрок выйдет с противоположной стороны, но это могут сделать и привидения.

Предусмотрено решение следующих задач:

1) Создать карту, на которой будет происходить игровой процесс

2) Создать объекты персонажей – Игрока и Привидений

3) Разработать алгоритм передвижения объектов по карте

4) Разработать алгоритм поедания точек и привидений, а также смерть

5) Разработать алгоритм подсчета очков и реализация рекордов

6) Создать меню

7) Создать графическое изображение объектов

**ГЛАВА 1**

**АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

**1.1 История игры «Pac-Man»**

Оригинальная аркадная игра Pac-Man была написана в Японии, но только когда компания Midway Games из Чикаго стала североамериканским дистрибьютором игры, ведение голодного желтого круга через лабиринт точек стало международным явлением. С момента своего выпуска в 1980 году игра породила различные продолжения и множество лицензионных товаров, что сделало ее одной из самых кассовых медиа-франшиз всех времен. В торговом центре Woodfield в пригороде Шаумбург есть даже ресторан и галерея, посвященные бренду.



Рис. 1.1

Созданный программистом Тору Иватани в 1980 году и выпущенный Namco в Японии и на полпути в США, желтый главный персонаж Pac-Man в форме пиццы и четыре красочных призрака привнесли индивидуальность в видеоигры. Игра-лабиринт захватила воображение миллионов людей и стала самой продаваемой аркадной видеоигрой за всю историю.

В то же время Pac-Man сам стал первым культовым послом эры видеоигр - одновременно символизируя видеоигры и превзойдя их, когда он перешел в массовую культуру. Игра вызвала первое массовое увлечение лицензированием видеоигр, стимулировавшее продажу домашних консолей, портативных устройств, игрушек, одежды и даже предметов домашнего обихода. ABC транслировала анимационное телешоу, а песня «Pac-Man Fever» в марте 1982 года попала в топ-10 чарта Billboard Hot 100. «Pac-Mania» подготовила почву для более поздних икон видеоигр, таких как Марио и Соник.

С момента своего выпуска Pac-Man и его многочисленные вариации и сиквелы проложили себе путь в бесчисленное количество аркад, домов и новых цифровых пространств. В 2013 году Namco и Disney запустили Pac-Man and the Ghostly Adventures, новый мультсериал, видеоигры и линейку игрушек, которые представили Pac-Man новому поколению.

Очарование Pac-Man в сочетании с захватывающей игрой делают его игрой, в которую каждое поколение сможет по-новому насладиться.

**1.2 Обзор существующих решений**

При написании данной курсовой работы, я рассмотрел несколько решений по созданию игры «Pac-Man». Далее мной будет описано одно из этих решений.

В самом начале данного решения было размещено изображение игрового поля со всеми стенками, которым была придана дополнительная жесткость. Это означает что через эти стенки нельзя пройти. Можно только упереться в них.

После этого был добавлен спрайт игрока. Также игроку добавили две анимации, анимация передвижения и анимация смерти игрока. Позднее были добавлены события обработки нажатия клавиш. Каждой кнопке соответствует свое направление.

Следующим шагом надо было заставить игрока двигаться и поворачивать. Для более чёткого координированного движения и чтобы не цеплялся за стенки, в каждый перекрёсток было поставлено по спрайту, координирующему направление. В точке пересечения координат игрока и данного спрайта, игрок может произвести поворот.

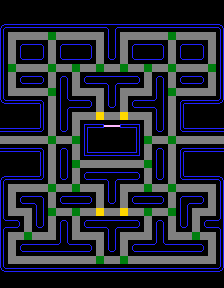


Рис. 1.2

Следующим шагом было создание врагов и оформление их поведения. Созданы они по образу игрока, с той лишь разницей, что на перекрестках направление выбирается случайно. Что напрочь лишило их каких-либо признаков интеллекта, зато добавило непредсказуемости в игре.

После были добавлены точки, которые являются целью игрока. При столкновении игрока с ними, точки не уничтожаются, а просто переходят в состояние невидимости. Но если призрак столкнется с невидимыми точками, они снова станут видимыми.

Также были добавлены два счётчика, текущее и лучшее количество очков.

Хоть данное решение и позволяет создать игру «Pac-Man», но в нем есть несколько недостатков.

Во-первых, игрок и приведения могут поворачивать только на специальных спрайтах. Это позволяет не цепляться за стены при повороте, но при этом сильно усложняет создание карты.

Во-вторых, привидения при повороте выбирают направление случайно. Это хоть и добавляет непредсказуемости в игру, но делает игру слишком простой. Игроку становится легче избегать встречи с призраками.

Также данное решение может хранить только один лучший рекорд. Кроме того, лучший рекорд содержит только количество очков. Этого недостаточно для создания таблицы рекордов.

В данном решении также не предусмотрено наличие меню. Это делает взаимодействие пользователя с интерфейсом игры менее удобным.

**1.3 Вывод по главе 1**

В первой главе была рассмотрена история игры «Pac-Man». Она описывает значимость игры для всей игровой индустрии. Также было рассмотрено существующее решение и были выявлены его недостатки.

На основе рассмотренного решения можно выделить основные требования к разработке игры «Pac-Man»:

1. Создать меню
2. Разработать улучшенный метод подсчета и хранения рекордов
3. Разработать улучшенный алгоритм передвижения привидений. Сделать их передвижение менее хаотичным
4. Улучшить алгоритм поворотов игрока и привидений

**ГЛАВА 2**

**ПРОЕКТИРОВАНИЕ ИГРЫ «PAC-MAN»**

**2.1 Проектирование логики**

**2.1.1 Алгоритм передвижения игрока**

В начале каждого раунда игрок появляется в определенной точке карты. Игрок начнет движение только после нажатия клавиши (вверх, вниз, вправо, влево). Игрок продолжает движение в том же направлении, даже если клавиша не нажата, и останавливается только после столкновения с препятствием.

Повороты при движении осуществляются нажатием клавиши клавиатуры. При этом поворот может быть произведен только в том направлении, в котором нет препятствия. Если же игрок выбрал направление, где стоит препятствие, то поворот не производится, а игрок продолжает двигаться в том же направлении.

Также должна быть реализована телепортация в случае если игрок заходит в коридор, ведущий за границу карты. В момент, когда игрок выходит за приделы карты, он должен выйти с противоположной стороны при этом сохраняя свои координаты по оси Oy.

**2.1.2 Алгоритм передвижения привидения**

В начале каждого раунда, привидения находятся в центре экрана в небольшой комнате, откуда они выходят по одному, а одно привидение начинает уровень вне комнаты.

После начала движения, каждое привидение направляется в свой угол карты и только после того как привидение достигнет своей цели, оно направляется в погоню за игроком. Но если игрок съест большую точку, находящуюся на карте, привидение будет убегать от игрока.

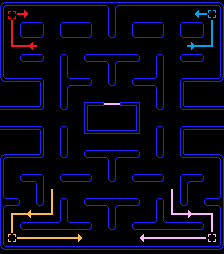


Рис. 2.1

Но привидения не всегда преследуют игрока. Время от времени они меняют цель преследования с игрока на свои углы. Это позволяет им не находится долго на одинаковых координатах. Также это позволяет сделать их поведение менее предсказуемым.

Изначально привидение движется в заданном направлении. Оно может изменить свое направление только когда оно полностью находится в одной клетке сетки, на которую разбита карта. При выборе направления привидение выбирает идти в сторону той клетки, которая ближе к цели, но оно не может повернуть в противоположную сторону.

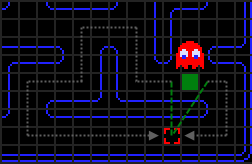


Рис. 2.2

В случае, когда игрок съедает привидение, оно не умирает, а направляется в комнату, в центре карты, там оно восстанавливается и отправляется вновь преследовать игрока.

**2.1.3 Подсчет очков и алгоритм рекордов**

В игре Pac-Man реализована система подсчета очков. Игрок может зарабатывать очки несколькими способами. Когда игрок съедает маленькую точку, он получает 10 очков. В случае, когда он съедает большую точку, он получает 50 очков и с этого момента он может съесть привидение. Если игроку удалось съесть приведение, он получает 200 очков.

Когда игрок проиграет (у него закончатся все 3 жизни) или нажмет на клавишу Escape, он сможет сохранить свой рекорд. В этом случае начинает работать алгоритм рекордов.

Сначала алгоритм проверит существует ли уже такой рекорд. Если уже такой рекорд, от этого пользователя, записан, то он не будет записан повторно. В случае если такого рекорда нет, алгоритм начнет запись нового рекорда в файл. Для этого он создаст вектор, в который запишет все рекорды, хранящиеся в файле, и в конец этого вектора он запишет новый рекорд.



Record

2

New record

Record

4

Record

3

Record

1

Рис. 2.3

Далее для каждого элемента он подсчитает финальное количество очков, по которому он потом будет их сравнивать. Финальное количество очков складывается из суммы очков, подсчитанных за игру, и номера уровня, на котором умер игрок, умноженного на 1000.

Final points = Score + Level \* 1000

Рис. 2.4

Такой способ сравнения позволяет наиболее честно распределить игроков в таблице рекордов. После этого алгоритм сортирует рекорды в векторе. Если количество рекордов в векторе меньше 7, то он запишет их всех, если нет, то будут записаны только первые 7 рекордов.

Таким образом в файл не будут записаны лишние рекорды. Также при выводе рекордов на экран, достаточно вывести всё, что хранится в файле.

**2.1.4 Алгоритм регистрации**

При запуске игры, пользователя встречает окно, в котором ему предлагается войти под существующим аккаунтом или создать новый. Но чтобы войти в аккаунт или создать новый, пользователю нужно вписать имя в первую строку и пароль во вторую. Если одна из строк будет пустая, то пользователю алгоритм покажет ошибку.

Если пользователь решил войти в свой аккаунт, то алгоритм начнет проверять есть ли аккаунт с таким именем и паролем в файле. Если такой аккаунт есть, то пользователь сможет войти в игру, но если такого аккаунта нет или пароль не совпадает, то алгоритм покажет ошибку.

Если пользователь решил создать новый аккаунт, то алгоритм будет проверять только имя. Если такое имя уже есть, то алгоритм покажет ошибку, но если такого имени еще нету, то пользователь сможет войти в игру и новый аккаунт запишется в файл.

**2.2 Проектирование взаимодействия пользователя с интерфейсом игры**

**2.2.1 Окно регистрации**

При запуске игры, пользователя встречает окно, в котором ему предлагается войти под существующим аккаунтом или создать новый. Данное окно показано на рисунке 2.5.

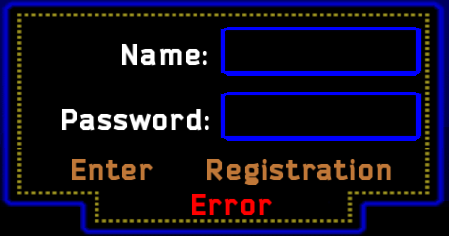


Рис. 2.5

В этом окне находятся две кнопки и две строки. В строку, на против которой расположена надпись Name, нужно вписать игровое имя, а в строку напротив которой расположена надпись Password, нужно вписать пароль от аккаунта. Для ввода текста в одну из данных строк требуется нажать левой кнопкой мыши во внутрь синей рамки, которая окружает нужную строку. После чего пользователь может производить ввод текста в данную строку. Для того чтобы стереть ненужный символ из строки нужно нажать на клавишу BackSpace.

После того как строки были заполнены, пользователь может произвести попытку входа или регистрации. Для того что бы произвести вход в игру, нужно навести мышкой на кнопку Enter и нажать на левую кнопу мыши. Для регистрации, требуется навести курсором на надпись Registration и нажать на левую кнопу мыши. В случае возникновения ошибки, внизу окна появится красная надпись Error.

**2.2.2 Главное меню**

После осуществления входа в игру. Перед пользователем появляется главное меню игры. Данное меню показано на рисунке 2.6.



Рис. 2.6

В самом низу главного меню белым цветом написано игровое имя того аккаунта, под которым был произведен вход в игру. Выше над этим именем расположены кнопки, которые активируются при наведении на них курсором и нажатием левой кнопкой мыши.

При нажатии на кнопку Play, главное меню сворачивается и запускается игра. Управление персонажем происходит при нажатии клавиш вверх, вниз, влево, вправо. Также при нажатии на клавишу Escape игра ставится на паузу и открывается игровое меню.

Если нажать на кнопку Record list, поверх меню откроется таблица рекордов. Если же нажать на кнопку Change user откроется окно регистрации, которое было описано выше. При нажатии на кнопку Exit происходит закрытие всей игры.

**2.2.3 Таблице рекордов**

Если нажать на кнопку Record list в главном меню, то откроется окно таблицы рекордов. Данное оно показано на рисунке 2.7.



Рис. 2.7

В самом низу этого окна расположена кнопка Menu, при нажатии на которую таблица рекордов закрывается и открывается окно главного меню.

Над этой кнопкой располагаются рекорды, которые были поставлены пользователями. Рекорды выводятся в отсортированном виде. Каждая строка рекордов содержит номер в таблице рекордов, игровое имя пользователя, количество очков, собранных в процессе игры, и номер уровня, до которого дошел пользователь.

**2.2.4 Игровое меню**

При нажатии на клавишу Escape, во время игры, или после проигрыша игрока, открывается игровое меню. Игровое меню, открытое при помощи клавиши Escape, показано на рисунке 2.8, а меню, открытое после проигрыша игрока, показано на рисунке 2.9.

В самом верху игрового меню располагаются количество очков, игровое время с начала раунда и уровень в данный момент игры. Ниже располагаются три кнопки. Две кнопки в обоих случая одинаковы, а третья отличается.

Рис. 2.9

Рис. 2.8





При нажатии на кнопку Menu, игра заканчивается и открывается главное меню при этом сохранение рекордов не происходит. Поэтому предусмотрена кнопка Save record, при нажатии на которую производится сохранение рекорда на данный момент игры.

Третья кнопка отличается в зависимости от ситуации. Если меню было вызвано кнопкой Escape, то третьей является кнопка Continue, при нажатии на которую игра начнется с того момента на котором она остановилась. Если же меню было вызвано проигрышем игрока, то продолжить игру невозможно. Поэтому кнопка Continue заменяется на кнопку Restart, при нажатии на которую игра начинается заново.

**2.3 Вывод по главе 2**

Во второй главе были приведены основные алгоритмы, которые будут использоваться в игре. Некоторые из этих алгоритмов были созданы на основе существующих, но также были доработаны под задачи данного проекта. Также были приведены возможности пользователя при взаимодействии с интерфейсом игры. После того, как мы разработали основные алгоритмы и возможности меню можно приступать к их реализации программным кодом.

**ГЛАВА 3**

**ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ИГРЫ «PAC-MAN»**

**3.1 Программная реализация алгоритмов**

**3.1.1 Алгоритм передвижения игрока**

Алгоритм передвижения игрока хранится в библиотеке Player. То есть ниже описанные алгоритмы написаны в файле Player.cpp. На рисунке 3.1 показана функция, которая организует передвижение персонажа при нажатии клавиши клавиатуры.

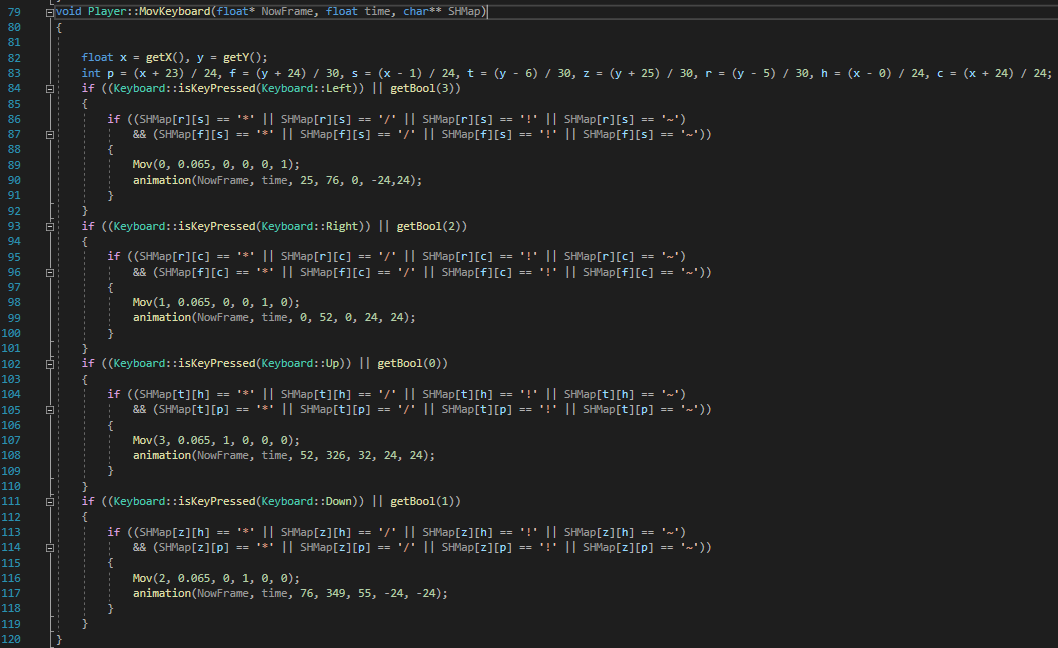


Рис. 3.1

В строке 82 переменным x и y присваиваются координаты игрока. Далее в строке 83 от этих координат вычисляются точки и их индексы в массиве, в котором хранится карта. Основываясь на этих данных, будет вычисляться окружение вокруг игрока.

В строках 84, 93, 102, 111 проверяется нажата ли клавиша клавиатуры или была ли она нажата последней. Проверка на то, нажата ли клавиша или нет, реализовывается при помощи функции библиотеки SFML, а проверка на то, была ли она нажата последней, реализовывается при помощи функции getBool, которая возвращает переменную типа bool, привязанную к данной клавише.

В строках 86-87, 95-96, 104-105, 113-114 реализована проверка, которая определяет есть ли препятствие в том направлении, куда хочет повернуть игрок. Это определяется при помощи переменных описанных в строке 83.

В строках 89, 98, 107, 116 происходит вызов функции, которая определяет направление, скорость, и переменную типа bool, определяющую какая клавиша была нажата. Структура данной функции показана на рисунке 3.2.



Рис. 3.2

В строках 90, 99, 108, 117 происходит вызов функции, которая отвечает за анимацию персонажа. Данная функция принимает счетчик кадров, игровое время и координаты изображения, использованного для нового кадра. Структура данной функции показана на рисунке 3.3.

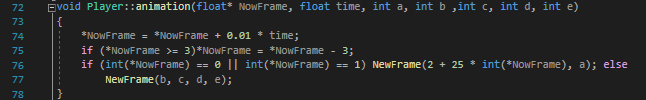


Рис. 3.3

В строках 74-75 вычисляется номер кадра, зависящего от времени. Далее в строках 76-77 вычисляется нужное изображение в зависимости от данного кадра.

Часть кода, отвечающая за телепортацию игрока при выходе за границы экрана, прописана в файле main.cpp. На рисунке 3.4 показан этот участок кода.

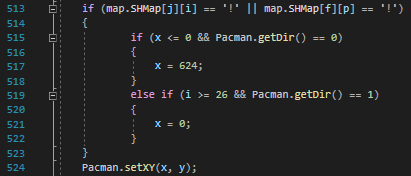


Рис. 3.4

В строке 513 реализована проверка, которая определяет находится ли игрок на особом символе, который находится за картой. Далее в строках 515 и 519 реализована проверка, которая определяет за какую границу (правую, левую) вышел игрок и в каком направлении он движется. В строках 517, 521 присваивается новое значение временной переменной x. После этого в строке 524 игроку присваиваются новые координаты.

**3.1.2 Алгоритм передвижения привидения**

Алгоритм передвижения привидения хранится в библиотеке Player. То есть ниже описанные алгоритмы написаны в файле Player.cpp. На рисунке 3.5 показана первая часть функции, которая организовывает передвижение персонажа.

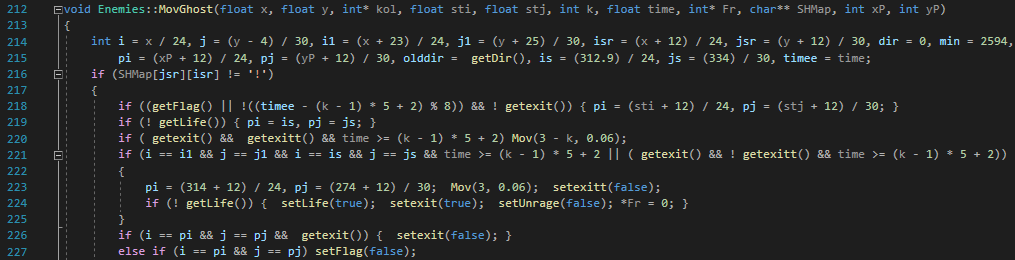


Рис. 3.5

В строках 214-215 координаты привидения переводятся в индексы массива карты, вычисляются вспомогательные точки, определяется направление движения и координаты игрока помечаются как цель привидения.

В строке 216 прописано условие, которое проверяет на то, что привидение не должно телепортироваться в данный момент.

В строке 218 определяется целью преследования угол карты, если привидение только начало свое движение или если наступил момент, когда привидение должно поменять цель с игрока на угол карты.

В строке 219 целью становиться комната в центре карты, если привидение мертво.

В строках 220-225 прописан алгоритм выхода привидения из комнаты. В строке 220 привидениям, которые строят по краям комнаты, сообщается направление, куда идти чтобы дойти до центра комнаты, если данному привидению пора выходить из комнаты. Далее в строке 221 проверяется, находится ли привидение в центре комнаты. Если это условие верно, то привидению присваивается новое направление движения, но если привидение при этом мертво, то оно восстанавливается.

В строках 226 и 227 убираются режимы выхода из комнаты и режима преследования угла, если цель была достигнута.

Далее в рисунке 3.6 будет описана вторая часть этой функции, которая отвечает за выбор нового направления для привидения.

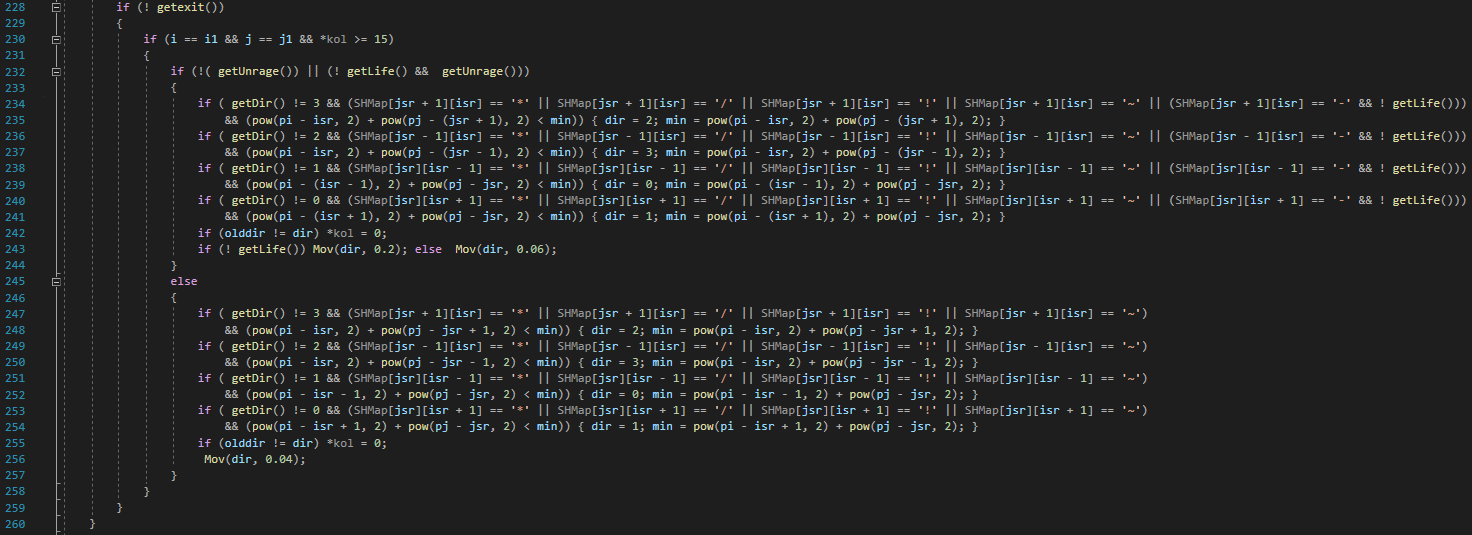


Рис 3.6

В строках 228 и 230 проверяется, что привидение не выходит из комнаты и что оно может выбирать направление соответственно. Далее в строке 232 проходит проверка на то, что приведение не может быть съедено игроком или что оно мертво. Если это условие верно, то в строках 234-241 определяется направление, которое ведет привидение к цели, но если данное условие не верно, то в строках 247-254 определяется направление, которое ведет привидение от цели.

В строках 243, 256 привидениям присваивается новое направление и скорость, определяемая в зависимости от его состояния.

В рисунке 3.7 описан алгоритм телепортации привидения, но он похож на алгоритм телепортации игрока.

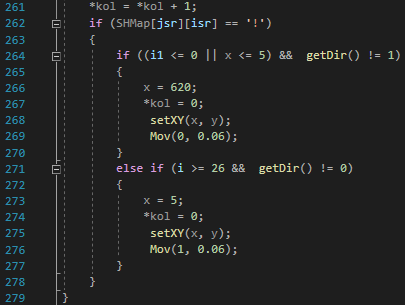


Рис. 3.7

**3.1.3 Подсчет очков и алгоритм рекордов**

Алгоритм подсчета очков и алгоритм рекордов прописаны в файле main.cpp. В рисунке 3.8 описана функция, которая отвечает за запись рекордов.

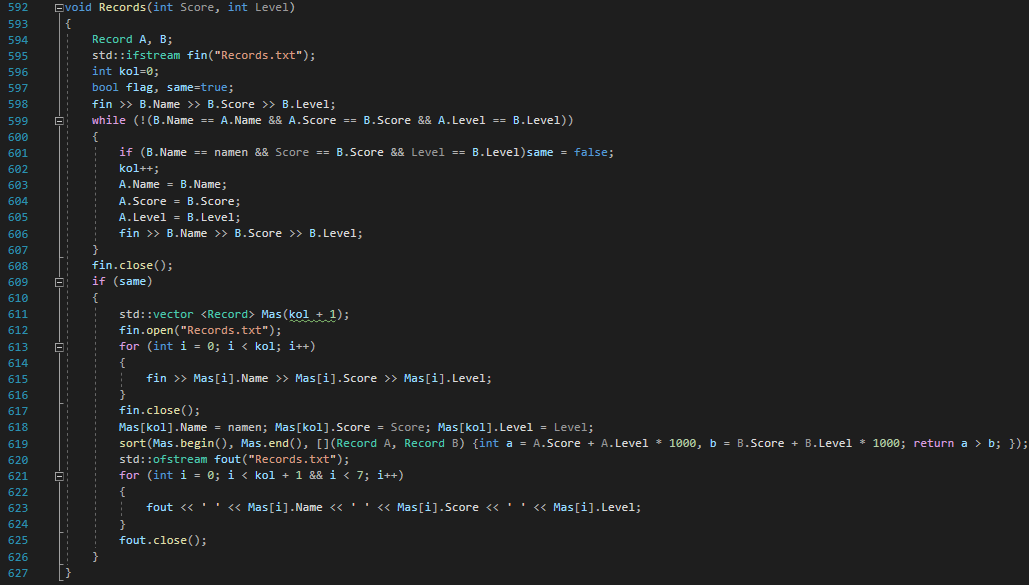


Рис. 3.8

В строках 594-597 объявляются вспомогательные объекты и переменные, а также объявляется поток, которым мы будем считывать информацию из файла с рекордами. Далее в строках 598-608 проверяется есть ли такой рекорд или нет, а также закрывается поток.

В строках 609-618 создается вектор размером на один больше чем количество рекордов в файле. Далее он заполняется рекордами из файла и в конец записывается новый рекорд. После этого в строках 619-625 данный вектор сортируется и записывается обратно в файл.

**3.1.4 Алгоритм регистрации**

Алгоритм регистрации прописан в файле main.cpp. В рисунке 3.9 прописан алгоритм входа под существующим аккаунтом.

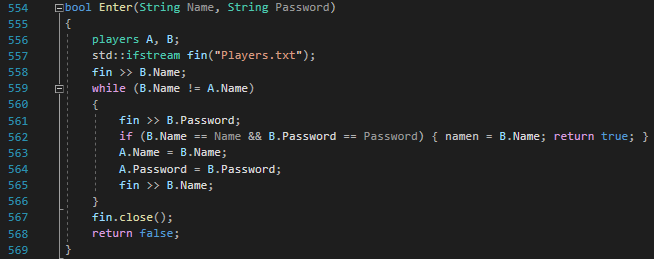


Рис. 3.9

В строках 556-557 объявляется вспомогательные объекты и открывается поток для чтения из файла игроков. Далее в строках 558-566 проводится поиск нужного аккаунта. Если такой аккаунт существует, то алгоритм разрешит пользователю войти. Если такого аккаунта нет, то в строке 568 алгоритм запретит вход в аккаунт.

Также пользователь может создать новый аккаунт. В рисунке 3.10 прописан алгоритм создания нового аккаунта.

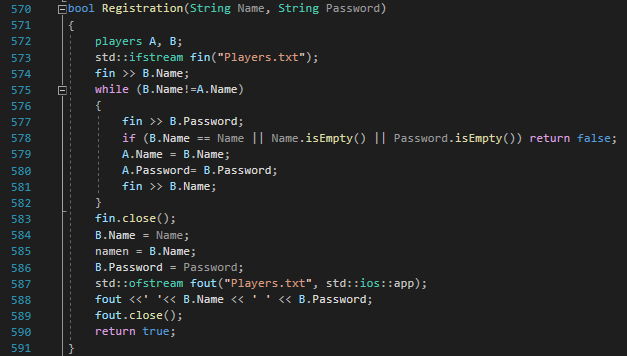


Рис. 3.10

В строках 572-573 объявляется вспомогательные объекты и открывается поток для чтения из файла игроков. Далее в строках 574-583 проводится поиск аккаунта с таким же именем. Если аккаунт с таким именем существует, то алгоритм запретит вход и новый аккаунт не запишется. Но если аккаунта с таким именем нет, то в строках 584-586 имя и пароль сохранятся. Также в строках 587-591 новый аккаунт запишется в файл и алгоритм разрешит пользователю войти.

**3.2 Программная реализация взаимодействия пользователя с интерфейсом игры**

**3.2.1 Окно регистрации**

Функция реализующая окно регистрации прописана в файле main.cpp. Первая часть данной функции прописана на рисунке 3.11.



Рис. 3.11

В строках 188-199 объявляется текст, который будет использоваться в окне регистрации. Также ему присваиваются основные настройки. В строках 200-209 объявляются изображения, которые используются в данном окне. Также определяются их текстуры и их положение в окне. Для этого используются объекты из графической библиотеки SFML.

В строках 197, 210, 211 объявляются вспомогательные переменные, которые будут использованы для функционирования интерфейса.

На рисунке 3.12 прописана часть функции организующая ввод текста.

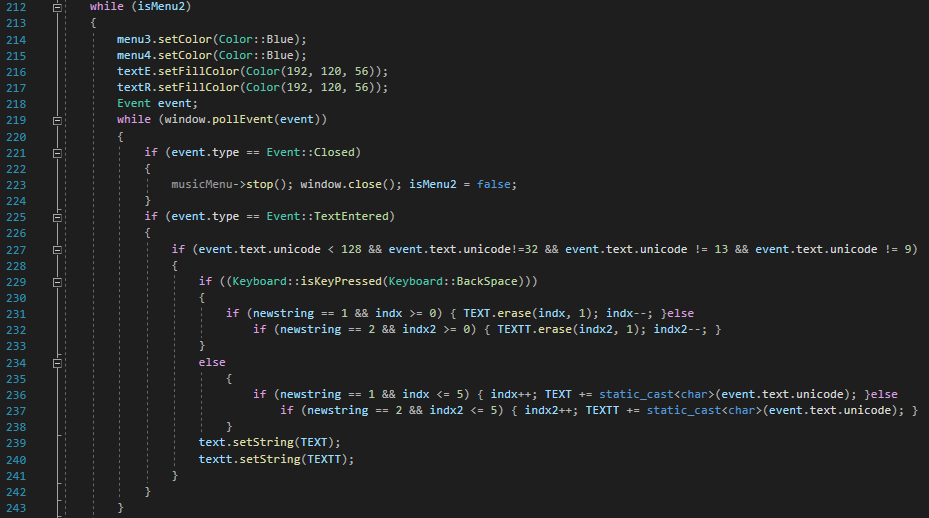


Рис. 3.12

В строке 212 прописано начало цикла, реализующего работу окна.

В строках 214-218 определяются настройки текста и изображения, к которым они должны возвращаться. Также объявляется объект event, который отвечает за обработку событий.

Далее в строках 219-243 реализован цыкал обработки событий. В строках 221-224 реализована обработка закрытия окна. В строках 225-242 реализована обработка ввода текста с клавиатуры. В строке 227 прописано условие, ограничивающее вводимые символы. Далее в строках 229-233 организовано стирание символа при нажатии клавиши BackSpace, а в строках 234-238 организовано добавление разрешенного символа в строку. В строках 239-240 строка переводится в текст, который будет выводится на экран.

На рисунке 3.13 прописана часть функции, которая отвечает за выбор строки, нажатия кнопки и прорисовки всего окна.



Рис. 3.13

В строке 244 происходит очистка окна. Далее в строках 245-250 определяется на какую кнопку или строку направлен курсор мыши. При этом цвет кнопки или рамки строки изменяется.

В строках 251-258 обрабатывается нажатие левой кнопки мыши по кнопке или по строке. Таким образом выполняется функционал кнопки или выбирается строка, в которую будет вводится текст.

В строках 253, 259 прописана проверка, чтобы клик засчитывался только в момент нажатия кнопки мыши.

В строках 260-275 реализована прорисовка объектов и вывод их на дисплей.

**3.2.2 Главное меню**

После того как вход в аккаунт был произведен, перед пользователем появляется главное меню. Функция реализующая данное меню прописана в файле main.cpp. Первая часть данного меню показана на рисунке 3.14.



Рис. 3.14

В строке 34 происходит смещение камеры по определенным координатам.

В строках 35-39 объявляется и запускается музыка, которая играет в главном меню.

В строках 40-47 объявляется текст, который будет выводится на экран. Также определяются его цвет, шрифт, позиция, размер. Далее в строках 48-50 определяется текстура и спрайт заднего фона меню.

В строках 51-53 объявляются вспомогательные переменные, которые используются при функционировании меню. А также объявляется объект, который отвечает за время внутри игры.

В строке 54 определяется позиция спрайта заднего фона в меню.

На рисунке 3.15 показана реализация работы главного меню.

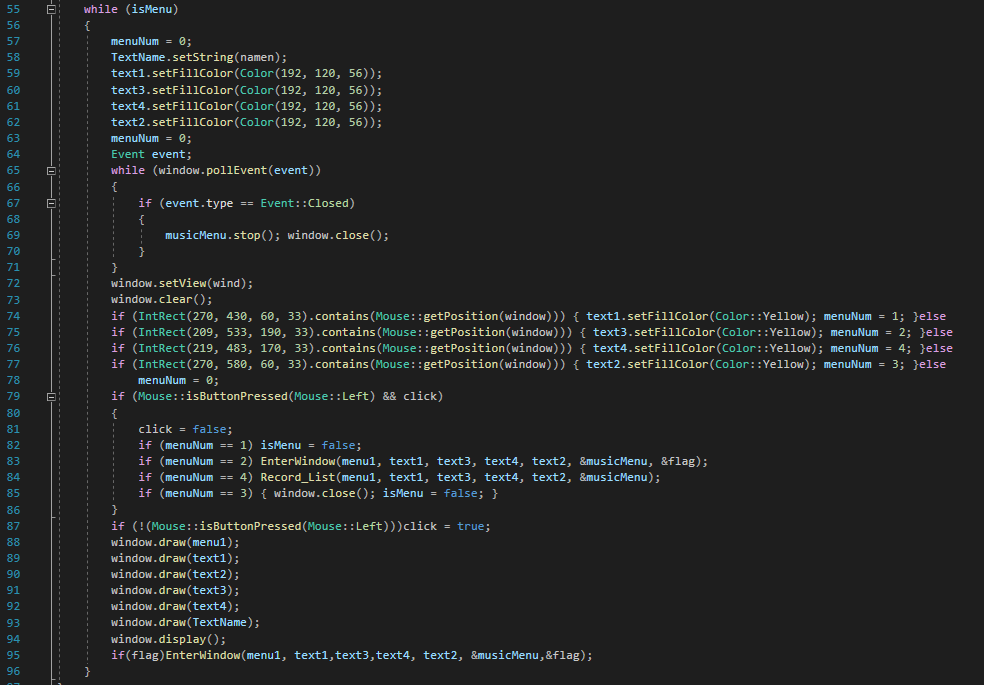


Рис. 3.15

В строке 55 прописано начало цикла, реализующего работу главного меню.

В строках 57-63 прописаны настройки текста, к которым он должен возвращаться. А также значения переменных.

В строках 64-71 объявляется объект event и обрабатывается закрытие окна.

В строках 72, 73 экран смещается и очищается. Далее в строках 74-78 определяется на какую кнопку направлен курсор мыши. В строках 79-87 обрабатывается нажатие на кнопку, на которую направлен курсор.

В строках 88-94 реализована прорисовка объектов и вывод их на дисплей. Далее в строке 95 вызывается окно регистрации, если меню запустилось впервые.

**3.2.3 Таблице рекордов**

Окно таблицы рекордов прописано в файле main.cpp. На рисунке 3.16 показана реализация первой части функции, которая создает таблицу рекордов.

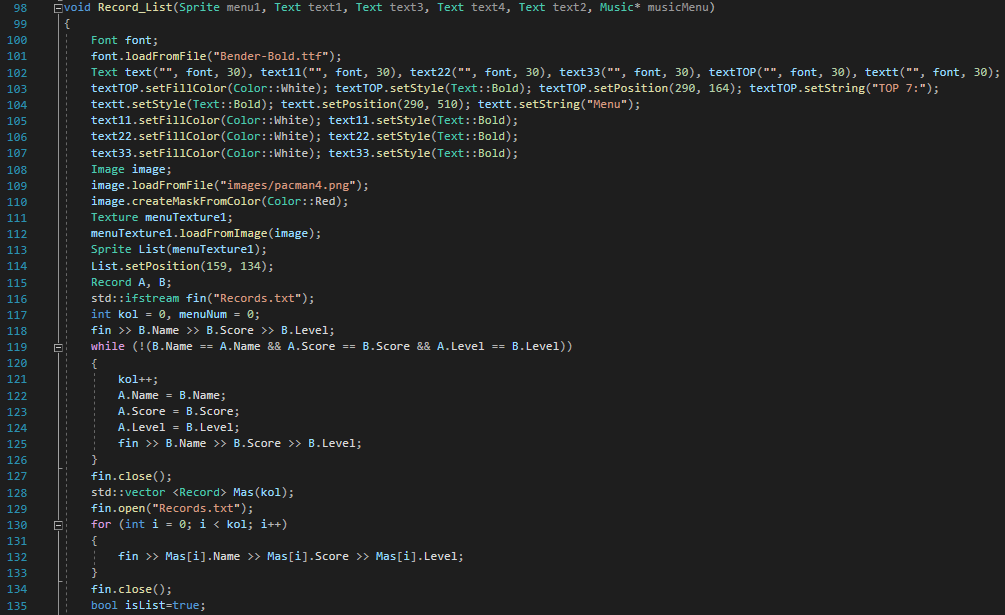


Рис. 3.16

В строках 100-107 объявляется текст, который будет выводится в окне таблицы рекордов. Также объявляются его основные характеристики.

В строках 108-114 объявляется изображение, его текстура и спрайт. Также определяется его положение на экране.

В строке 115 объявляются вспомогательные объекты. После этого в строках 116-127 подсчитывается количество рекордов в файле. Далее в строках 128-134 создается вектор, который заполняется рекордами. В строке 135 определяется вспомогательная переменная.

На рисунке 3.17 показана реализация второй части данной функции.



Рис. 3.17

В строке 136 прописано начало цикла, в котором реализована таблица рекордов. Далее в строках 139-144 создается объект event и обрабатывается закрытие окна.

В строках 145-147 реализована проверка на то, что курсор наведен на кнопку. Далее в строках 148-153 реализуется нажатие на эту кнопку.

В строках 154-162 реализована очистка окна и прорисовка объектов. Далее на рисунке 3.18 показана прорисовка рекордов.

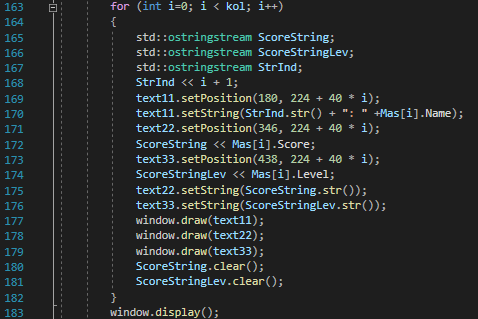


Рис. 3.18

В строках 163-182 реализуется прорисовка рекордов на экран. В строках 165-167 объявляются потоки перевода чисел в строку. В строках 168, 172, 174 номер, количество очков и номер уровня переводятся в строку. Далее в строках 169, 171, 173 определяется положение текста. В строках 170, 175, 176 строки переводятся в текст, который будет выведен на экран.

В строках 177-179 прорисовывается текст. Далее в строках 180-181 очищаются строки.

В строке 183 все, что было прорисовано, выводится на экран.

**3.2.4 Игровое меню**

Игровое меню прописано в файле main.cpp. На рисунке 3.19 показана первая часть функции, которая реализовывает игровое меню.

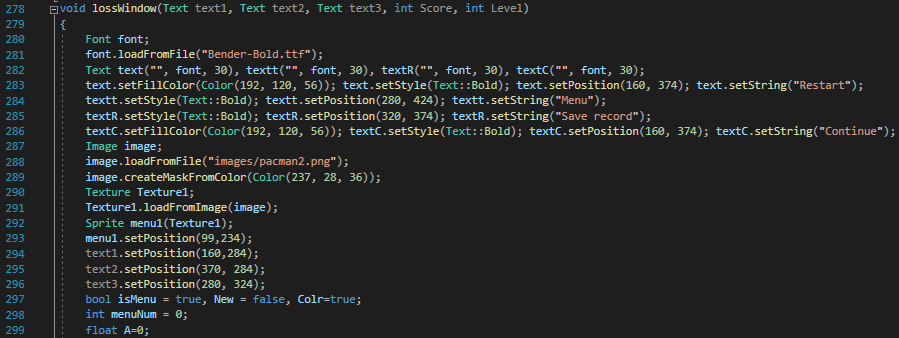


Рис. 3.19

В строках 280-286 объявляется текст, который будет выводится в окне игрового меню. Также определяются его основные характеристики.

В строках 287-292 объявляется изображение, текстура и спрайт. Далее в строках 293-296 определяется положение изображения и текста.

В строках 297-299 объявляются вспомогательные переменные, при помощи которых функционирует данное меню.

На рисунке 3.20 показана вторая часть функции, в которой описано функционирование игрового меню.

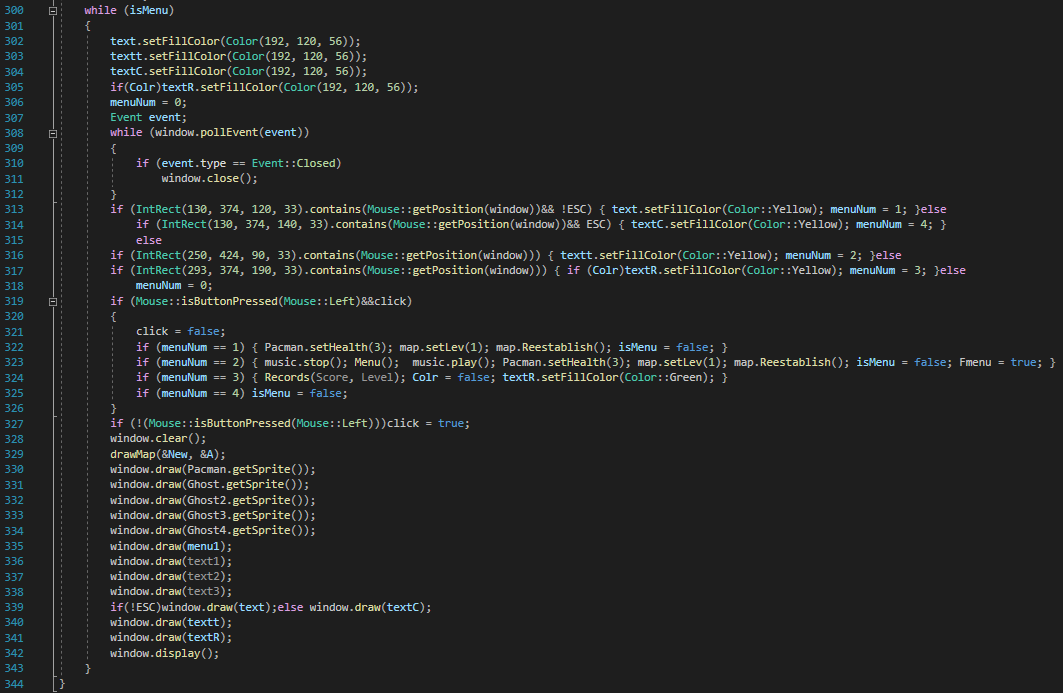


Рис. 3.20

В строках 302-306 определяется изначальный цвет кнопок. Далее в строках 307-312 объявляется объект event и обрабатывается закрытие окна.

В строках 313-318 определяется на какую кнопку направлен курсор мыши. Далее в строках 319-327 обрабатывается нажатие клавиши мыши.

В строке 328 происходит очистка окна. Далее в строках 329-342 происходит прорисовка объектов и вывод их на экран.

**3.3 Вывод по главе 3**

Таким образом мы разобрались с функционалом программы и ее реализацией. В данной главе была описана реализация основных алгоритмов данного проекта, а также была описана реализация взаимодействия пользователя с интерфейсом игры. Наглядно показаны функции игры.

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы были изучены алгоритмы оригинальной игры Pac-Man и на их основе были разработаны алгоритмы, более подходящие для данного проекта. На этапе разработки программы были получены практические навыки создания игры. Благодаря использованию дополнительной учебной литературы, особых трудностей при написании работы не возникло.

Дальнейшую модификацию программы вижу в возможности дополнения ее новыми функциями и возможностями.

В первой главе курсовой работы была описана история игры «Pac-Man». Также было рассмотрено существующее решение и были выявлены его недостатки.

Во второй главе были спроектированы основные алгоритмы, использованные в курсовой работе. А также был спроектирован интерфейс игры.

В третьей главе была описана реализация логики игры, а также реализация взаимодействия пользователя с интерфейсом игры.

В заключении хочется отметить, что реализованные функции могут быть применены не только в данном проекте. Данные алгоритмы можно использовать как в других играх, так и в различных приложениях.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Чукич И. Функциональное программирование на языке C++/ И. Чукич; пер. с англ. В. Ю. Винника, А. Н. Киселева. – Москва: ДМК Пресс, 2020. -360 с.
2. Новости разработки игр, руководства и многое другое [Электронный ресурс]: Более пристальный взгляд на SFML. -URL: <https://gamefromscratch.com/a-closer-look-at-sfml/>(дата обращения: 11.03.2021)
3. SFML [Электронный ресурс]: Простая и быстрая мультимедийная библиотека. -URL: <https://www.sfml-dev.org/>(дата обращения: 30.02.2021)
4. База знаний Хабр [Электронный ресурс]: Алгоритм поведения привидений в игре Pac-Man. -Москва, 2010. -URL: <https://habr.com/ru/post/109406/> (дата обращения: 15.03.2021)