СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc532204135)

[1 АНАЛИЗ ЗАДАНИЯ И ПОСТАНОВКА ЗАДАЧ 5](#_Toc532204136)

[2 ПРОЕКТИРОВАНИЕ ПРОГРАММЫ 7](#_Toc532204137)

[3 РЕАЛИЗАЦИЯ ПРОГРАММЫ 9](#_Toc532204138)

[4 ТЕСТИРОВАНИЕ ПРОГРАММЫ 15](#_Toc532204139)

[ЗАКЛЮЧЕНИЕ 19](#_Toc532204140)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 20](#_Toc532204141)

[ПРИЛОЖЕНИЯ 21](#_Toc532204142)

# Введение

Компьютерная игра – компьютерная программа, служащая для организации игрового процесса (геймплея), связи с партнёрами по игре, или сама выступающая в качестве партнёра.

В настоящее время в ряде случаев вместо «компьютерная игра» может использоваться видеоигра, то есть данные термины могут употребляться как синонимы и быть взаимозаменяемыми. В компьютерных играх, как правило, игровая ситуация воспроизводится на экране дисплея или обычного телевизора, но в то же время компьютерная игра может быть звуковой, телетайповой и др.

Темой данного курсового проекта является разработка игровой программы «Pacman». Pac-Man (с англ. — «Па́кман») — аркадная видеоигра, разработанная японской компанией Namco и вышедшая в 1980 году. Задача игрока — управляя Пакманом, съесть все точки в лабиринте, избегая встречи с привидениями, которые гоняются за героем.

К программе предъявляются следующие требования:

1. аппаратная архитектура – PC совместимый компьютер;
2. операционная система – MS Windows 7/8/8.1/10;
3. язык программирования – C++;
4. среда разработки – Microsoft Visual Studio 2017;
5. вид приложения – графическое приложение.

Для написания программы была выбрана среда программирования Visual Studio 2017 C++, основанная на языке программирования C++. Данная среда выгодно отличается эффективностью и надёжностью.

Язык С++ является универсальным языком программирования с достаточно большими возможностями работы с ОС и аппаратной частью электронных вычислительных машин. Поэтому, строго говоря, он позволяет решить практически любую задачу программирования. Тем не менее, в силу разных причин (не всегда технических) для каких-то типов задач данный язык употребляется чаще, а для каких-то – реже.

Также программа включает в себя библиотеку SFML (Simple and Fast Multimedia Library). Данная библиотека содержит функции, предназначенные для разработки игровых приложений. SFML содержит ряд модулей для простого программирования игр и мультимедиа приложений.

# 1 Анализ задания и постановка задач

Целью курсового проекта является создание игровой программы «Pacman», с реализацией системы подсчета количества набранных очков, а также редактирование карты. Под стандартным типом понимается реализация игровой программы с использованием текстур и моделей, а также звукового сопровождения.

Тематика игровой программы Pacman предполагает наличие нескольких ключевых элементов, определяющих данную игру. Поэтому было принято решение о включении в игровую программу следующих основных игровых элементов:

1. Pacman – игровой объект, обладающий уникальной текстурой, который находятся под управлением игрока и способен передвигаться вверх\вниз\влево\вправо, при этом взаимодействовать с другими объектами. Также он будет иметь свою реакцию, которая будет рассчитываться при столкновении\взаимодействии с тем или иным объектом.
2. Задний фон – выбран черного цвета. Так как очень важно выбрать правильный цвет заднего фона что бы не было наложений или слияния текстур друг с другом. Задним фоном был выбран черный цвет, так как это цвет подходит по всем параметрам.
3. Враги – игровые объекты, обладающие уникальными текстурами, которые способны перемещаться по всему полю. Враги могут взаимодействовать с Pacman. Они предназначены для того, чтобы усложнить игру игроку.
4. Стенки карты – динамические изображения линий, содержащие текстуру игрового поля, на котором будет происходить игра. Также на данном поле будут находиться статические объекты, с которыми может взаимодействовать Pacman, при взаимодействии будут начисляться очки.
5. Телепорты – статичные игровые объекты, обладающие уникальной текстурой. При попадании в телепорт игрок (Pacman) телепортируется в другой портал.
6. Панель информации – левая часть окна игровой программы, содержащая количество очков, которое набрал пользователь. Располагается эта информация за картой.
7. Монетки – игровые объекты, обладающие уникальной текстурой которые удаляются с карты при поедании их Pacman’ом, а также добавляются к игровому счету игрока.
8. Редактор карт – отдельный пункт меню, который будет удобен пользователям. В нем должно быть реализовано редактирование каждой текстуры, а также игра на этой карте.
9. Настройки звука – отдельный пункт, который позволяет удобно изменять громкость звуковых эффектов и музыкального сопровождения в игре.

Важной частью игрового процесса является аудио-звуковое сопровождение. При помощи музыки игра «оживает».

Музыку в игровой программе можно разделить на два главных типа:

1. музыка главного меню;
2. музыка игрового процесса.

Звуковое сопровождение программы позволит пользователю получить отклик от взаимодействия с ней. Можно выделить следующие типы звуков в игровой программе:

1. звук нажатия указателям мыши по клавише в главном меню;
2. звук попадания Pacman’a во врага;
3. звук, как Pacman погибает;
4. звук движения Pacman’a;
5. звук поедания монеток.

Одним из преимуществ данной игровой программы будет меню. В данном меню будет присутствовать 4 вида кнопок: «Новая игра», «Редактор карт», «Настройки», «Выход».

«Новая игра» – при нажатии данной кнопки меню начинается новая игра.

«Редактор карт» – при нажатии данной кнопки меню игроку выведется редактор, в котором вы можете создать карту на свой вкус.

«Настройки» – при нажатии данной кнопки меню игрок переходит в отдельную вкладку с настройками, где предлагается взаимодействие с музыкой и звуковыми эффектами.

«Выход» – при нажатии данной кнопки меню программа завершает работу.

# 2 Проектирование программы

При проектировании программы в первую очередь требуется сделать простой и удобной интерфейс, обеспечить работоспособность игры, корректное отображение данных.

Данная игровая программа будет написана с помощью функций SFML. Опираясь на данную библиотеку, можно сказать, что за всю визуальную составляющую проекта и за считывание нажатий клавиш будет отвечать SFML.

При проектировании программы главным этапом является выделение её ключевых составных частей, а точнее модулей. Каждая такая часть ответственна за определенные задачи, решаемые в процессе выполнения программы.

В процессе проектирования будут введены следующие основные три модуля:

* main – главный модуль разрабатываемого средства.
* Menu – модуль, отвечающий за реализацию интерфейса в программе.
* game – модуль, отвечающий за реализацию игровой составляющей и её изображения.

Предполагается, что будущий проект будет иметь следующую модель подключения модулей (рисунок 2.1).



**Рисунок 2.1** – Схема подключения модулей

В главном модуле main предполагается только одна функция:

* main – данная функция является основной. В данной функции будет реализовано скрытие консольного меню, создание главного окна, в котором будет происходить все основные действия, а также будет реализовано ограничение кадров отображения.

В модуле Menu предполагается несколько функций:

* menu – данная функция будет представлять собой создание главного меню. В данной функции будет происходить загрузка текстур для меню, отображение данных текстур и отрисовка окна, будет отслеживаться нажатие кнопок, а далее по нажатию одной из клавиш будет происходить действие, связанное с кнопкой, которую выберет пользователь.
* editMap – данная функция будет предоставлять возможность создать собственную карту и сыграть на ней.
* settings – функция, отвечающая за прорисовку меню настроек и обработку нажатия клавиш. Самое главное в данной функции – это логика, которая в последующих действиях будет менять настройки игры.
* playSE – функция должна будет производить звук при определенных нажатиях кнопок.

Один из самых важных модулей — это game. Данный модуль будет отвечать за игровую составляющую всего проекта. В данном модуле будет несколько функций, которые будут отвечать за ее работоспособность и корректность игрового процесса.

* game – данная функция будет содержать в себе всю механику игрового процесса. В функции будут присутствовать все вызовы, связанные с созданием игрового поля для игры. Будет происходить загрузка игровой музыки, которая в дальнейшем будет воспроизводиться при определенных условиях. Также в функции будет происходить загрузка текстур игрового поля и игровых объектов. Одно из важных свойств функции – это отслеживание нажатий соответствующих клавиш; после будет выполняться действия, связанные с данной кнопкой, которую нажал пользователь. Для удобства в данной функции реализован вывод в панель информации количества очков, которое заработал игрок.

# 3 Реализация программы

Данная программа представляет собой игровое приложение, в котором реализован процесс игры по типу Pacman. Программа позволяет использовать следующие возможности: играть, настраивать звуковые эффекты и музыкальное сопровождение, редактировать карту.

Большая часть программы разработана с помощью SFML. SFML ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Simple and Fast Multimedia Library – простая и быстрая мультимедийная библиотека) – свободная [кроссплатформенная](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) мультимедийная [библиотека](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)).

SFML содержит ряд модулей для простого программирования игр и мультимедиа приложений. Исходный код библиотеки предоставляется под лицензией [zlib/png license](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D1%86%D0%B5%D0%BD%D0%B7%D0%B8%D1%8F_zlib).

Первым модулем, который будет рассматриваться, — это модуль main. Данный модуль имеет только одну функцию – main. В данной функции происходит определение окна всей игровой программы с помощью функции RenderWindow window. При запуске программы будет создаваться окно размером 800х600 пикселей, размеры которого строго фиксированы. С окном можно производить только базовые действия: свернуть/развернуть окно, перетаскивать по всей области экрана. Также выполняется скрытие консольного окна. Далее выставляется лимит кадров в секунду и подключается русский язык. После подключения всех параметров, выполняется вызов меню. Ниже представлен листинг 3.1 с функцией main.

**Листинг 3.1** – Функция main

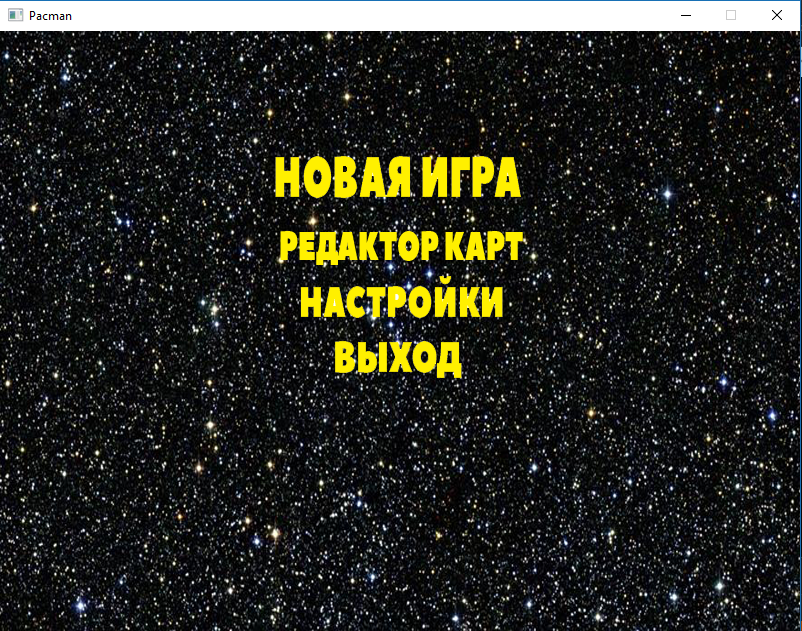
1. int main()
2. {
3. ShowWindow(GetConsoleWindow(), SW\_HIDE); //Скрытие консольного окна
4. RenderWindow window(VideoMode(800, 600), "Pacman",Style::None|Style::Close);
5. window.setFramerateLimit(70);
6. menu(window);
7. return 0;
8. }

После того, как окно создано и все параметры заданы, можно перейти к следующему модулю. Все, что связанно с описанием интерфейсных настроек, редактором карт, музыкальным сопровождением и управлением звуковыми эффектами, находится в модуле Menu. Данный модуль содержит несколько функций.

Первой из функций, которая также вызывается из main, является void menu(RenderWindow &window).Её основная задача – это создание, отображение меню. Благодаря данной функции имеется возможность выбрать несколько действий: Новая игра, Редактор карт, Настройки, Выход. Сама функция принимает лишь единственную переменную window. Это переменная является окном, на котором будет прорисовываться фоновое изображение и все элементы кнопок.

Первоначально в данной функции создаются переменные для управления громкостью звука volume\_sound\_effect и volume\_music. Далее создается буфер для звука SoundBuffer tapBuffer. В данный буфер загружается один из звуков tapBuffer.loadFromFile("tap.ogg"), который в последующем будет активироваться при нажатии одной из кнопок. Следующим действием создается событие звука, в который загружается звук из буфера Sound tap(tapBuffer). Для управления программой существует две переменные: bool exitMenu, int numberMenu. Переменная exitMenu — это логическая переменная, она отслеживает момент игры, когда пользователь нажимает кнопку Выход. Переменная numberMenu — это целочисленная переменная. В данную переменную записывается число, каждое число соответствует какой-то кнопке, при нажатии на левую часть мыши происходит перенаправление на следующее действие, связанное с той кнопкой, на которую нажал пользователь.

Также в данной функции присутствует цикл while (exitMenu && window.isOpen()), который позволяет отрисовывать всё меню. В данном цикле первоначально идет проверка на фокус окна, после идет проверка расположения курсора мыши мониторе, и если курсор находится в зоне действия одной из кнопок, то данная кнопка окрашивается в красный цвет, при нажатии кнопки происходит перенаправление, а точнее действие, связанное с одной из кнопок. Затем все элементы меню отрисовываются в окне (рисунок 3.1).



**Рисунок 3.1** – Внешний вид окна menu

При нажатии кнопки Редактор карт, осуществляется переход в функцию void editMap(RenderWindow & window, Sound tap). Данная функция принимает две переменные. Одной из них является наша переменная окна window, на котором будут изображаться все графические элементы. Вторая — это переменная звука tap, которая будет производиться в момент нажатия клавиши. В первую очередь идёт загрузка текстур для редактирования игрового поля. Для управления программой есть две переменные bool exitMenuTablicaRecords, int numberMenuRecord. Переменная exitMenuTablicaRecords — это логическая переменная, которая отслеживает событие нажатия кнопки «Выход». Переменная numberMenuRecord — это целочисленная переменная. В данную переменную записывается число, которое соответсвует определенной кнопке, при нажатии на левую кнопку мыши происходит перенаправление на следующее действие, связанное с той кнопкой, которую выбрал пользователь.

При нажатии кнопки Настройки, осуществляется переход в функцию void setting(RenderWindow & window, Sound tap). Данная функция принимает две переменные. Одной из них является переменная окна window, на котором будут изображаться все графические элементы. Вторая переменная — это переменная звука tap, которая будет воспроизводиться в момент нажатия клавиши. В данной функции описаны настройки игры, которые связаны с музыкальным сопровождением и звуковыми эффектами. Пользователю будет доступна возможность увеличить/уменьшить громкость музыки/звуковых эффектов (рисунок 3.2).



**Рисунок 3.2** – Меню настроек

Также в данной функции присутствует цикл while (exitMenuSettings && window.isOpen()), который позволяет отрисовывать графическую часть настроек. Данный цикл имеет точно такую же функциональность, что и в меню.

Присутствуют также функция void playSE(Sound tap).

Функция void playSE(Sound tap) отвечает за звуковые эффекты. В данной функции сначала устанавливается уровень громкости, а затем воспроизводится звуковой эффект.

Модуль game представляет собой всю механику игровой программы. Первой функцией, которая вызывается из меню, является int game(RenderWindow & window). Данная функция принимает одну переменную, а именно переменную окна window, на котором будут изображаться все графические элементы игры и всей игровой механика.

Первоначально в функции создается буфер музыки и осуществляется загрузка текстур. Дальше в цикле while (!back && window.isOpen()) происходит считывание кнопки, на которую нажал пользователь. В программе присутствует проверка на наличие препятствия перед Pacman’ом (монетки/стенки/телепорт). Если Pacman встречается с монеткой, то ему прибавляются очки. Если перед Pacman'ом стенка, то его движение остановится. Если Pacman переместился на телепорт, то происходит перемещение к другому телепорту. Ниже представлен листинг 3.2 телепортации Pacman’а одного из телепортов.

**Листинг 3.2** – Телепортация Pacman’а

1. if (TileMap[i][cellX] == 'p')
2. {
3. x = 35; y = 325;
4. herosprite.setPosition(x, y);
5. break;
6. }

В самом конце цикла идет проверка на выигрыш/проигрыш. Первоначально, если игрок выиграл либо проиграл, происходит обнуление карты. Затем игра загружает текстуры с надписями «Вы выиграли!» или «Вы проиграли!», а также текстура текста «Для продолжения нажмите клавишу Enter». Затем программа ожидает нажатия клавиши Enter и выходит в меню. Ниже представлена реакция игры на проигрыш (рисунок 3.3) и на выигрыш (рисунок 3.4).



**Рисунок 3.3** – Реакция игры на проигрыш



**Рисунок 3.4** – Реакция игры на выигрыш

Во время создания карты, вызываются функции, в которых происходит создание объектов карты. Сперва создаются стенки, которые будут ограничивать игровое поле. После этого создается Pacman.

Затем, когда были созданы стенки карты и сам Pacman, происходит загрузка текстур, которые будут соответствовать определенным объектам на карте. Вся игровая поверхность будет делиться на две области: игровая область – область, на которой будет происходить вся игровая механика, и информационная область – область, на которой будет выводиться количество набранных очков игроком.

Одной из важных функций является функция void updateMap(). Она обновляет все переменные, которые были использованы. Эти переменные, отвечающие за музыку старта, музыку заднего фона и звуковые эффекты, а также переменная, отвечающая за количество набранных очков. Это требуется для нормальной работы игровой механики. Далее представлен листинг 3.3 с обновлением переменных.

**Листинг 3.3** – функция updateMap

1. void UpdateVariable()
2. {
3. for (int i = 0; i < HEIGHT\_MAP; i++)
4. for (int j = 0; j < WIDTH\_MAP; j++) {
5. if (TileMap[i][j] == ' ')
6. {
7. TileMap[i][j] = 'm';
8. }
9. }

Все модули тесно связаны между собой и дополняют друг друга, при удалении одного из модулей игровая программа перестанет работать.

# 4 Тестирование программы

Тестирование программы – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и ожидаемыми результатами на конечном наборе тестов, выбранных определенным образом.

Тестирование программы проводится с помощью метода белого ящика.

Тестирование методом белого ящика (также прозрачного, открытого, стеклянного ящика; основанное на коде или структурном тестировании) – метод тестирования программного обеспечения, который предполагает, что внутренняя структура системы известна тестировщику. Выбираются входные значения, основанные на знании кода, который будет их обрабатывать. Также известно, каким должен быть результат этой обработки. Тестирование белого ящика – углубление во внутренне устройство системы, за пределы ее внешних интерфейсов.

**Преимущества:**

* + - тестирование может производиться на ранних этапах: нет необходимости ждать создания пользовательского интерфейса;
    - можно провести более тщательное тестирование, с покрытием большого количества путей выполнения программы.

**Недостатки:**

* + - для выполнения тестирования белого ящика необходимо большое количество специальных знаний;
    - при использовании автоматизации тестирования на этом уровне, поддержка тестовых скриптов может оказаться достаточно сложной, если программа часто изменяется.

Тестирование программы и результаты представлены в таблице 4.1

**Таблица 4.1** – Тестирование программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Действие** | **Ожидаемый результат** | **Полученный результат** | **Метка** |
| 1 | Запуск программы | Первоначально осуществляется создание окна приложения и его отрисовка, а также происходит скрытие консольного окна. | При запуске программы осуществляется создание окна с заданными параметрами, отрисовка в данном окне всех текстур, также скрывается консольное окно. | Выполнено |

Продолжение таблицы 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Действие** | **Ожидаемый результат** | **Полученный результат** | **Метка** |
| 2 | Наведение курсора мыши на любую кнопку | При наведении на любую из кнопок в меню, она должна окраситься в красный оттенок. | При наведении на любую кнопку в меню, она окрашивается в красный оттенок, если убрать курсор, кнопка принимает стандартный вид. | Выполнено |
| 3 | Нажатие на кнопку «Новая игра» | Должен воспроизвестись звуковой сигнал, а затем происходит переход в игру. | При нажатии на кнопку “Новая игра” воспроизводится звуковой сигнал. После происходит переход в саму игру. | Выполнено |
| 4 | Нажатие на кнопку «Редактор карт» | Должен произойти звуковой сигнал, затем переход в редактор карт. | При нажатии на кнопку «Редактор карт» воспроизводится звуковой сигнал. После происходит переход в редактор карт. | Выполнено |
| 5 | Работа с редактором карт | Пользователю предлагается выбор текстур для их отображения на карте. | При переходе в редактор карт пользователю предлагается выбор текстур для отображения их на карте. | Выполнено |
| 6 | Нажатие на кнопку «Настройки» | Должен произойти звуковой сигнал, а потом произойти переход в настройки игру. | При нажатии на кнопку «Настройки» воспроизводится звуковой сигнал. После происходит переход в сами настройки игры. | Выполнено |

Продолжение таблицы 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Действие** | **Ожидаемый результат** | **Полученный результат** | **Метка** |
| 7 | Изменение настроек | Пользователю должны предоставляться изменение громкости звуковых эффектов и музыкального сопровождения. Предоставлена возможность изменять музыкального сопровождения. Изменение громкости, каждая часть желтых делений соответствует процентному соотношению самой музыки. | При переходе в настройки пользователю предоставляется возможность изменить звуковые эффекты и музыкальное сопровождение. При изменении громкости каждое желтое деление соответствует процентному соотношению громкости музыкального сопровождения, само музыкальное сопровождение играет на заднем плане игровой программы. При отключении музыки вопроизведение музыкальной дорожки останавливается. | Выполнено |
| 8 | Проверка на ходьбу при нажатии клавиш W A S D или стрелочек | При нажатии на W A S D или стрелочек Pacman должен поворачиваться в соответствующую сторону. | При нажатии на W или «стрелочка вверх», S или «стрелочка вниз», A или «стрелочка влево», D или «стрелочка вправо» Pacman поворачивается в соответствующую сторону. | Выполнено |

Окончание таблицы 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Действие** | **Ожидаемый результат** | **Полученный результат** | **Метка** |
| 9 | Столкновение со стенками | При столкновении со стенками игра должна остановить Pacman’a. | При столкновении со стенками Pacman останавливается. | Выполнено |
| 10 | Столкновение Pacman’а с бонусами | При столкновении Pacman’а с бонусными объектами, игроку должно начисляться некоторое количество очков и воспроизводиться звуковой эффект. | При столкновении Pacman’а с бонусными объектами игроку начисляются очки. Также при столкновении с объектами воспроизводится звуковой эффект. | Выполнено |
| 11 | Вывод меню на экран | После запуска меню происходит прорисовка фонового изображения и элементов меню на рабочей области окна. | Фоновое изображение и все элементы главного меню выводятся корректно и в правильном формате. | Выполнено |

В ходе тестирования программы не было замечено сбоев или аварийного завершения работы игрового приложения, что свидетельствует о его полной работоспособности.

# Заключение

Была поставлена задача по реализации игровой программы «Pacman», которая была успешно решена в среде разработки Visual Studio 2017. Удалось реализовать игровую программу, представляющую собой игру на подобие Pacman, в которой реализовано движение Pacman’a и некоторая часть функциональности привычной игры. Также был реализован редактор карт с возможностью создания и игры на этой карте.

В процессе работы были реализованы все возможные функции. Кроме основных функций, также были реализованы вспомогательные функции, такие как управление громкостью звуков и музыки, обновление переменных.

В программе был соблюден принцип модульности программы и дружественный интерфейс. На каждом этапе осуществляется проверка на корректность введенных данных.

В качестве вспомогательной библиотеки для работы с графикой и звуком используется популярная библиотека SFML.

Проведено полное тестирование программы с разбором ее основных возможностей. Библиотека SFML предназначена для работы с графикой и звуком на языке C++ и OpenGL. С ее помощью можно достаточно просто реализовать графику и анимацию в игре, а также использовать систему событий окна для лучшей отзывчивости управления и оптимизации потребления ресурсов компьютера [7].

В результате игровая программа была реализована со всеми требованиями: была реализована сама игра, также был реализован редактор карт. Сама игровая программа работает стабильно. Поставленная задача выполнена в полной мере.

# Список использованных источников

1. Шиманович Е. Л. C/C++ в примерах и задачах. - Минск: Новое знание, 2016, - 528с.
2. Лаффе Р. Объектно-ориентированное программирование в С++. Классика Computer Science. – СПб.: Питер, 2016 – 928 c.
3. Гилберт Стивен, Маккартни Билл. Самоучитель Visual C++ 6 в примерах. - К. : ООО "ТИД ДС", 2015. - 496с.
4. Эккель Б., Эллисон Ч. Философия С++. Практическое программирование. – СПб.: Питер, 2016. – 608 с.
5. SFML URL: https://www.sfml-dev.org/tutorials/2.4/
6. Д.Э. Кнут Искусство программирования. Том 1. Основные алгоритмы.
7. Д.Э. Кнут Искусство программирования. Том 2. Получисленные алгоритмы.
8. Д.Э. Кнут Искусство программирования. Том 3. Сортировка и поиск.