Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Программирование на языках высокого уровня

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

Игра “PACMAN”

БГУИР КП 1-40 02 01 301 ПЗ

Студент: Алборов Е.Е.

Руководитель: Ассистент кафедры ЭВМ

Марзалюк А. В.

МИНСК 2022

Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Программирование на языках высокого уровня

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

\_\_\_\_\_\_\_\_\_ Б. В. Никульшин

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2022 г.

ЗАДАНИЕ по курсовому проектированию

студенту Алборову Егору Евгеньевичу

**1.** Тема проекта: Игра”PACMAN”

**2.** Срок сдачи студентом законченного проекта: *10.12.2022*

**3.** Исходные данные к проекту:

***3.1*** *Операционная система: Windows*

***3.2*** *Язык программирования С++, графическая библиотека SFML.*

**4.** Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке): *Задание на курсовой проект. Введение. 1. Литература. 2.Структурное проектирование.3.Функциональное проектирование.  
4. Руководство пользователя. Заключение. Список используемых источников. Приложения.*

**5.** Перечень графического материала (с точным обозначением обязательных чертежей и графиков):

***5.1****Скриншоты работающей программы*

***5.2****Диаграмма классов*

***5.3*** *Блок-схемы алгоритмов*

**6.** Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапом):

*Введение, Задание на курсовой проект к 10 октября 2022 г.- 10%;*

*раздел 1, 2 к 10 ноября 2022 г. – 40 %;*

*разделы 3, 4, Заключение к 10 декабря 2022 г. – 40 %;*

*оформление пояснительной записки до 10 декабря 2022 г. – 10 %*

*Защита курсового проекта с 19 декабря 2022 г. по 30 декабря 2022 г.*

Дата выдачи задания: *16 сентября 2022 г.––––––*

РУКОВОДИТЕЛЬ \_\_\_\_\_\_\_\_\_\_\_\_\_– А.В. Марзалюк

(подпись)

ЗАДАНИЕ ПРИНЯЛ К ИСПОЛНЕНИЮ *­­*\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.Е. Алборов

(подпись)

**СОДЕРЖАНИЕ**

ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ……………………………...5

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ…………………………………………...6

ВВЕДЕНИЕ………………………………………………………………………..7

1 ОБЗОР НА ЛИТЕРАТУРУ…………………………………………….…….....8

2 СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ…………………………………….....9

2.1 Структура программы………………………………………………...9

2.2 Структура входных данных………………………..………………..10

2.3 Класс *Position………*…………………………………………………10

2.4 Класс *SimpleObject*…………………………………………………..11

2.5 Класс *Draw...*………………………………………………………....11

2.6 Класс *MovingObject…*………………………………………………..11

2.7 Класс *GhostManager…*………………………………………………12

2.8 Класс *Pacman*……………….………………………………………..12

2.9 Класс *Ghost…………*…………………………………………………13

2.10 Класс *Myqueue….*…………………………………………………...13

2.11 Класс *MainMenu*…………………………………………………....14

2.12 Класс *MenuRecords….*……………………………………………...15

3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ………………………………16

3.1 Алгоритм по шагам метода convert\_sketch………………………….16

3.2 Алгоритм по шагам метода IsInFileNewRecord………………….….16 3.3 Схемы алгоритмов………………………....………………………….17

4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ…………………………………………18

ЗАКЛЮЧЕНИЕ………………………………………………………………….21

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ…..…………………………….22

ПРИЛОЖЕНИЕ А……………………………………………………………….23

ПРИЛОЖЕНИЕ Б………………………………………………………………..25

ПРИЛОЖЕНИЕ В………………………………………………………………..27

ПРИЛОЖЕНИЕ Г………………………………………………………………..29

ПРИЛОЖЕНИЕ Д…..……………………………………………………………76

# **ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ**

ОС – операционная система.

ООП – Объектно-ориентированное программирование.

STL – (Standard Template Library) –Библиотека стандартных шаблонов.

ЭВМ – Электронно-вычислительная машина.

SFML – Simple and Fast Multimedia Library

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ**

Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Информация о рекордах игроков должна храниться в отдельном файле.   
 Реализация операций добавления, удаления рекордов. Разработка иерархии классов с использованием наследования (не меньше 3-х уровней наследования). Разработка и использование в программе классов контейнеров и алгоритмов (STL.) Обработка исключительных ситуаций.

**ВВЕДЕНИЕ**

Объектно-ориентированное программирование представляет собой технологию программирования, которая базируется на классификации и абстракции объектов. Одним из наиболее популярных средств объектно-ориентированного программирования, позволяющим разрабатывать программы, эффективные по объёму кода и скорости выполнения является С++.

C++ — мощный язык, содержащий средства создания эффективных программ практически любого назначения, от низкоуровневых утилит и драйверов до сложных программных комплексов самого различного назначения. В частности:

- предсказуемое выполнение программ является важным достоинством для построения систем реального времени. Весь код, неявно генерируемый компилятором для реализации языковых возможностей (например, при преобразовании переменной к другому типу), определён в стандарте. Строго определены места программы, в которых он выполняется. Это даёт возможность замерять или рассчитывать время реакции программы на внешнее событие.

- автоматический вызов деструкторов объектов при их уничтожении, в порядке, обратном вызову конструкторов, что позволяет избежать утечек памяти.

- язык поддерживает понятие (const) константности, что позволяет компилятору, например, диагностировать ошибочные попытки изменения значения константной переменной. Объявление константности даёт программисту, читающему текст программы дополнительное представление о правильном использовании классов и функций, а также может являться подсказкой для оптимизации.

- язык имеет крупную стандартную библиотеку STL, которая включает в себя распространённые контейнеры и алгоритмы.

- С++ сочетает в себе как возможности низкоуровневых языков программирования, так и возможности высокоуровневых. Имеется возможность работы на низком уровне с памятью, адресами.

- кроссплатформенность: стандарт языка накладывает минимальные требования на ЭВМ для запуска скомпилированных программ. Доступны компиляторы для большого количества платформ, на языке C++ разрабатывают программы для самых различных платформ и систем.

- эффективность. Язык спроектирован так, чтобы дать программисту максимальный контроль над всеми аспектами структуры и порядка исполнения программы.

**1 ОБЗОР НА ЛИТЕРАТУРУ**

Pac-Man — одна из самых знаковых видеоигр всех времен, и большинство людей (даже не геймеры) хотя бы поверхностно с ней знакомы. Цель игры очень проста — игрок помещается в лабиринт, наполненный едой (изображенной в виде шариков или точек), и ему нужно съесть все это, чтобы перейти на следующий уровень. Эту задачу усложняют четыре призрака, которые преследуют Pac-Man(желтый кружек, за которого играет пользователь программы) по лабиринту. Если Pac-Man вступает в контакт с любым из призраков, игрок теряет жизнь, а позиции Pac-Man и призраков сбрасываются обратно в исходное положение. Помимо того, что Pac-Man просто избегает их, единственной защитой от призраков являются четыре большие гранулы «energizer», расположенные по углам лабиринта. Поедание одного заставляет призраков пугаться и отступать на короткое время, а   
Pac-Man может даже съесть их в течение этого периода. Съеденный призрак не уничтожается полностью, а возвращается в исходное положение, прежде чем возобновить преследование.

Призраки всегда находятся в одном из трех возможных режимов: “преследование”,“побег” или “испуг”. «Нормальный» режим, в котором призраки преследуют Pac-Man, — это “преследование”, и это тот, в котором они проводят большую часть своего времени. В режиме “преследование” все призраки используют позицию Pac-Man как фактор при выборе своей цели.

В режиме “побег”у каждого призрака есть фиксированная плитка лабиринта , каждая из которых расположена за углом лабиринта. Это приводит к тому, что четыре призрака расходятся по углам всякий раз, когда они находятся в этом режиме. Режим “испуг” уникален, потому что в этом режиме у призраков нет определенной целевой плитки. Вместо этого они псевдослучайно решают, какие повороты делать на каждом перекрестке. Призрак в режиме “испуг” становится темно-синим, движется намного медленнее и может быть съеден Pac-Man. Однако продолжительность испуганного режима сокращается по мере прохождения игроком уровней.

**2 СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ**

Одним из важнейших пунктов в ООП является разработка логической модели системы в виде диаграммы классов. Диаграмма классов для представления статической структуры модели в терминологии классов ООП. Она отражает связи между классами и структурами.

Диаграмма классов программы представлена в приложении А.

Необходимые фрагменты кода можно найти в приложении Д.

# **2.1 Структура программы**

В программе можно выделить несколько основных элементов: блок взаимодействия пользователя с программой (меню), блок анализа поведения враждебных существ(призраков), блок анализа движений персонажа(Пакман)  
блок записи и чтения рекордов, блок обработки ошибок, блок часто используемых переменных, блок главного игрового цикла, который является ядром программы, блок отрисовки, реализованный с помощью графической библиотеки SFML, блок анализа действий пользователя при работе с окнами.

Блок взаимодействия пользователя с программой отвечает за выбор пользователем определённых действий, например начать игру( кнопка Play), посмотреть рекорды(кнопка Records), выйти из игры(кнопка Exit), управление персонажем(Пакман) с помощью стрелочек, выход из окон клавишей ESC, удаление рекордов клавишей DELETE.

При запуске программы, создаются основные объекты, используемые в программе для работы, например массив символов-объектов карты для отрисовки, объект класса Pacman, объект класса GhostManager для более удобной работы с призраками.

В блоке обработки ошибок в основном происходит анализ успешного открытия файлов.

В блоке часто используемых переменных содержатся объекты перечисляемого типа(enum), константные переменные, все это доступно в любой точке программы, например время отрисовки кадра(в миллисекундах), численная абстракция скорости движения объектов(в миллисекундах), размер минимальной единицы отрисовки с помощью SFML.(constexpr unsigned char CELL\_SIZE = 16).

В блоке отрисовки, реализованном с помощью графической библиотеки SFML, используются такие методы как a) загрузка текстур из папки Resources

б) работа со спрайтами

в) Работа с окнами, в частности создание их и установление видимости игрового поля

г) функции и методы для обработки окон  
  
В блоке анализа действий пользователя при работе с окнами происходит обработка нажатий клавиш(стрелочки вверх и вниз) и последующая отрисовка.  
  
Соответствующие фрагменты кода можно найти в приложении Г

# **2.2 Структура входных данных**

Информация в программе хранится в текстовом файле (наиболее удобный для редактирования и понятный рядовому пользователю формат).

Структура данных, считываемых из файла, хранящего информацию о рекордах игроков, представлена на рисунке 2.2.1

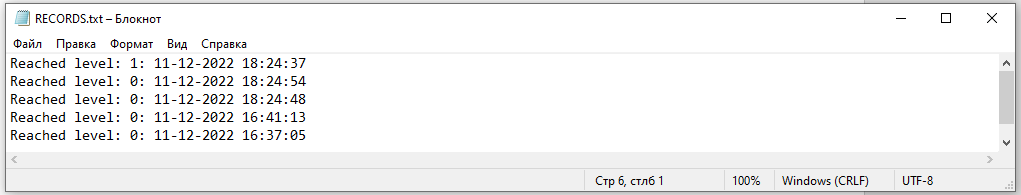


Рисунок 2.2.1 – Структура данных файла «RECORDS.txt»

**2.3 Класс Position**

Содержит координаты объектов

**Атрибуты класса**:

short x; позиция по оси абсцисс

short y; позиция по оси ординат

***Методы:***

bool operator==; оператор присваивания

**2.4 Класс SimpleObject**

Содержит основные атрибуты объектов

**Атрибуты класса**:

*Position position;* содержит координаты объектов*;*

***Методы:***

get\_position(); вернуть позицию объекта  
set\_position(); установить позицию объекта

**2.5 Класс Draw**

Виртуальный класс, содержащий метод отрисовки

**Атрибуты класса**:

*нету*

***Методы:***

Virtual void draw(bool,sf::RenderWindow&)=0;

отрисовка спрайтов игровых объектов

**2.6 Класс MovingObject**

Содержит поля и методы, необходимы для описания интеллекта призраков и анализа движений PacMan.

**Атрибуты класса**:

unsigned char direction; содержит направление движения

unsigned short animation\_timer;

необходим для отрисовки соответствующих спрайтов

***Методы:***

unsigned char get\_direction(); получить направление  
void set\_direction(unsigned char direction);  
установить направление  
unsigned short get\_animation\_timer();  
получить текущее время анимации

void set\_animation\_timer(unsigned short i\_animation\_timer); установить время анимации

**2.7 Класс GhostManager**

Ведет вспомогательную работу с призраками. В частности переключает их режимы.

**Атрибуты класса**:

unsigned char current\_wave; текущая волна

unsigned short wave\_timer; время, отведенное текущей волне

std::array<Ghost, 4> ghosts; контейнер призраков

***Методы:***

void update(); обновление логики призраков  
void draw(bool i\_flash,sf::RenderWindow&i\_window); переопределенный метод отрисовки призраков.  
void reset(); сбросить текущие данные призраков

**2.8 Класс Pacman**

Содержит методы и поля для обработки главного персонажа

**Атрибуты класса**:

bool dead; статус смерти

unsigned short energizer\_timer;

таймер действия гранулы усиления

bool animation\_over;

нужно для отрисовки надписей после смерти или выигрыша

***Методы:***

bool get\_dead(); узнать жив ли персонаж

void set\_dead(bool i\_dead);

установить жив персонаж или мертв

unsigned short get\_energizer\_timer();

узнать время действия гранулы усиления

void set\_energizer\_timer(unsigned short energ\_tim); установить время действия гранулы усиления

bool get\_animation\_over();

нужно для анализа отрисовки надписей после смерти или выигрыша

void set\_animation\_over(bool an\_tim); установить время конца анимации

void reset(); сбросить характеристики PacMan к первоначальным

void draw(bool i\_victory, sf::RenderWindow& i\_window); отрисовка спрайта PacMan

void update(); обновление позиции PacMan;

**2.9 Класс Ghost**

Содержит методы и поля для обработки призраков

**Атрибуты класса**:

bool dead; статус смерти

unsigned short energizer\_timer;

таймер действия гранулы усиления

bool animation\_over;

нужно для отрисовки надписей после смерти или выигрыша

***Методы:***

bool movement\_mode; установка режима погони или разбегания

unsigned char frightened\_mode; установка режимов “преследование”, “побег”,”испуг”

unsigned char id; уникальный номер призрака

bool use\_door; может ли призрак использовать дверь

unsigned char frightened\_speed\_timer; установка соответствующего режима

Position home;позиция “дома”

Position home\_exit; позиция “выход из дома”

Position target; позиция “цель”

**2.10 Класс Myqueue**

Собственный шаблонный контейнер для содержания данных любого типа

**Атрибуты класса**:

struct Node

{

Node(T var); конструктор для присваивания значения

Node\* next; указатель на следующий элемент

Node\* previous; указатель на предыдущий элемент

T data; шаблонный тип данных

};

struct head\_and\_tail

{

head\_and\_tail();

Node\* head; указатель на голову очереди

Node\* tail; указатель на хвост очереди

};

head\_and\_tail h\_t;

объект, содержащий указатели на голову и хвост.

***Методы:***

*Myqueue();* конструктор

void push(T variable); поместить объект в конец очереди

void pop(); извлечь объект из головы очереди

T& front(); получить ссылку на объект в голове очереди

T& back(); получить ссылку на объект в конце очереди

T& peek(int id); получить ссылку на объект по индексу

void print(); вывести содержимое контейнера с последующим

удалением объектов

bool isEmpty(); пуст ли контейнер  
void clean(); очистить контейнер  
int amount(); получить количество элементов  
void pop\_back(); извлечь с хвоста  
~Myqueue(); деструктор

**2.11 Класс MainMenu**

Класс для анализа действий пользователя в меню и отрисовки поясняющего текста

**Атрибуты класса**:

int MainMenuSelected; индекс нажатой кнопки  
sf::Font font; шрифт выводимого текста  
sf::Text mainMenu[Max\_main\_menu]; массив текста

***Методы:***

MainMenu(float width, float height); конструктор

void draw(sf::RenderWindow& window); отрисовка текста меню

void MoveUp(); анализ нажатия стрелочки вверх

void MoveDown(); анализ нажатия стрелочки вниз

int MainMenuPressed(); возвращает индекс нужной кнопки

~MainMenu(); деструктор

**2.12 Класс MenuRecords**

Класс для анализа действий пользователя в меню рекордов и отрисовки поясняющего текста

**Атрибуты класса**:

int MenuRecordsSelected; индекс нажатой кнопки

sf::Font font; шрифт

std::string infoRecords[Max\_menu\_records];  
массив рекордов

sf:: Text textRecords[Max\_menu\_records];  
соответствующие тексты

static int position; для заполнения массива текстов при входе в рекорды

***Методы:***

MenuRecords(); конструктор

void draw(sf::RenderWindow& window); отрисовка текста

void MoveUp(); анализ стрелочки вверх

void MoveDown(); анализ стрелочки вниз

void analyze(std::string); при входе в рекорды заполняет массив теста

void deleteRecord(); обработка удаления рекордов

std::string\* getRecords(); получить рекорд

int MenuRecordsPressed(); возвращает индекс нажатой кнопки

# **3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

Необходимые фрагменты кода можно найти в приложении Г.

**3.1 Алгоритм по шагам метода *convert\_sketch***

Алгоритм по шагам convert\_sketch заполнение контейнера STL array output\_map соответствующими значениями-константами по анализу символов, заданными заранее в контейнере STL array map\_sketch

1. Начало.

2. Входные данные: ссылка на STL контейнер array i\_map\_sketch, содержащий заранее заполненную карту символами для последующего анализа,   
ссылка на STL контейнер array i\_ghost\_positions, содержащий позиции призраков,   
ссылка на объект Pacman(для установки его позиции).

3. создание STL контейнера array output map, в котором будут храниться соответствующие константы(после анализа символов из i\_map\_sketch)

4. Выбор одного из элементов поля(проход будет с левого верхнего угла до правого нижнего, реализовано это с помощью двойного цикла)

5. Заполнение текущего места в output\_map константой Cell::Empty

6. В зависимости от символа в i\_map\_sketch, заносится соответствующая  
константа в output\_map(Cell::Wall, Cell::Door, Cell::Pellet), либо устанавливаются позиции призраков или пакмана.

7. Возвращаем output\_map.

8. Конец.

**3.2 Алгоритм по шагам метода *IsInFileNewRecord***

Алгоритм по шагам метода IsInFileNewRecord– загрузка(или не загрузка) предполагаемого рекорда в файл RECORDS.txt.

1. Начало.

2.Входные данные- текущий уровень

3. Создание объекта string с возможным новым рекордом

4.Открытие файла, если такого файла не было, то он создается и в него заносится новый рекорд, переходим к пункту алгоритма

5. Создание собственного контейнера Myqueue<string> queueOfRecords

6. Считывание из файла RECORDS.txt рекордов и занесение их в контейнер Myqueue<std::string> queueOfRecords.

7. Открытие этого же файла на запись

8. Занесение в файл рекордов из моего контейнера до тех пор пока он не станет пустым или же не занесем новый рекорд

9.Если был занесен новый рекорд, то запись в файл оставшихся рекордов из контейнера до тех пор, пока их количество не превысит AMOUNTRECORDS

10. Если новый рекорд не был занесен и при этом ничего в файле до этого не было или что-то было, но при этом количество рекордов больше 0 и меньше AMOUNTRECORDS, то в файл дописывается новый рекорд

11. Закрытие файла

12. Конец

# **3.3 Схемы алгоритмов**

Схема алгоритма FillFileAfterRemove() представлена в приложении Б.

Схема алгоритма get\_target\_distance() представлена в приложении В.

**4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

Далее приведены инструкции использования программы . Для передвижения по меню нужно использовать стрелочки “вверх” и “вниз”.  
Главное меню изображено на (рис. 5.1). Меню рекордов изображено на   
(рис. 5.2)

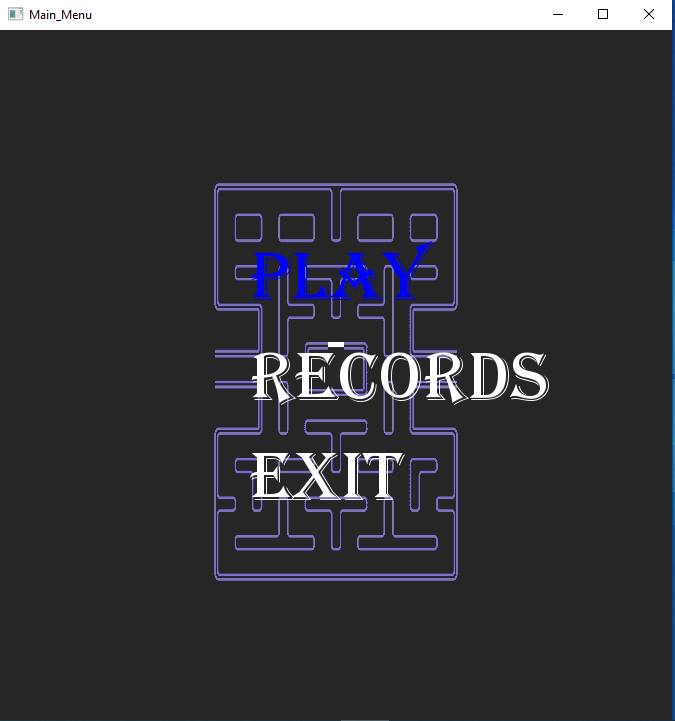
****

Рисунок 5.1

****

Рисунок 5.2

Передвижение по игровому полю осуществляется с помощью стрелочек “влево” “вправо” “вверх” “вниз”. Выход из игры осуществляется нажатием клавиши ESC или нажатием левой кнопки мыши по значку “крестик” игрового окна. Игровое поле изображено на (рис. 5.3).

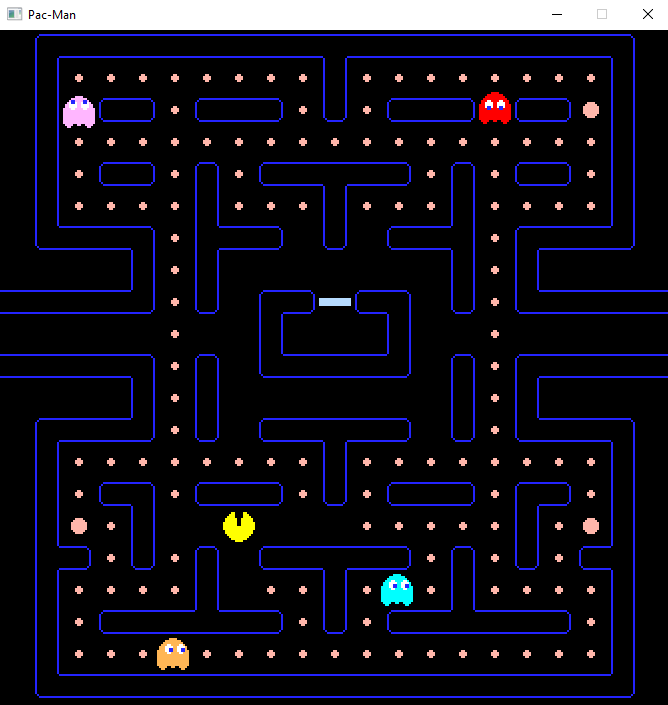


Рисунок 5.3

При нажатии соответствующих клавишь выхода выскочет окно предупреждения, где пользователю предложат выйти без сохранения рекорда(выход осуществляется клавишей SPACE) или продолжить игру при этом нажав на любую клавишу, кроме SPACE.  
Предупреждение перед выходом из игры, что рекорд не сохранится изображено на (рис. 5.4).

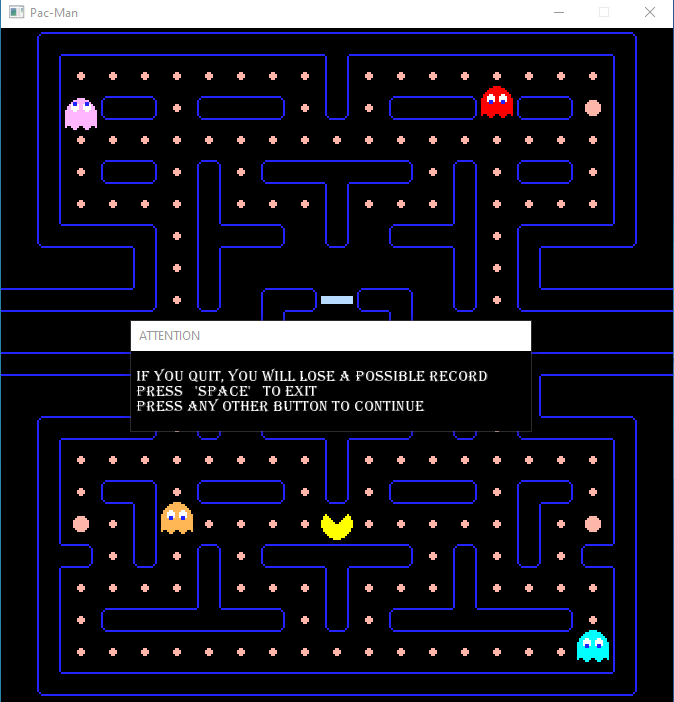


Рисунок 5.4

При поражении или выигрыше, появится соответствующее окно. Для продолжения игры необходимо нажать клавишу ENTER

Проигрыш изображен на (рис. 5.5).  
Выигрыш изображен на (рис. 5.6).



Рисунок 5.5

****

Рисунок 5.6

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения данного курсового проекта были применены на практике знания, полученные в ходе изучения курса ПНАЯВУ, а также в ходе самостоятельного обучения. Благодаря этому была реализована программа игра “Pacman”, проведено тестирование данного программного продукта и устранение ошибок.

Были реализованы методы взаимодействия пользователя с программой, реализована логика призраков, удаление рекордов.

Данный проект может быть усовершенствован добавлением отдельного потока для анализа нажатых клавиш игроком, так как в моей реализации, когда программа отрисовала кадр, то я отключаю поток на промежуток времени, чтобы установить фиксированное количество кадров в секунду, если же игрок нажмет на клавишу в этот момент(это конечно очень мало вероятно, так как промежуток времени слишком мал, но все же такое возможно), то нажатие не обработается, а с другим потоком эта проблема была бы решена.

# **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

[1] Дейтел, Х.М. Как программировать на С++ / Х.М. Дейтел, П.Д. Дейтел; пер. с англ. – М. : Бином, 2007. – 1152 с..

[2] Страуструп, Б. Язык программирования С++ / Б. Страуструп; специальное издание. Пер. с англ. – СПб. : BHV, 2008. – 1098 с.

[3] Элджер, Дж. C++: библиотека программиста / Дж. Элджер. – СПб. : Питер, 2001 с.

[4] Объектно-ориентированное программирование в С++/ Роберт Лафоре г. пер. с англ. – Санкт-Петербург, 2019. – 1152 с.

[5] Поведение призраков/ https://gameinternals.com/understanding-pac-man-ghost-behavior . Посещение 15.11.2022

**Приложение А**

*(обязательное)*

Диаграмма классов

**Приложение Б**

*(обязательное)*

Схема алгоритма метода FillFileAfterRemove()

25

**Приложение В**

*(обязательное)*

Схема алгоритма метода Ghost::get\_target\_distance()

27

**Приложение Г**

*(обязательное)*

Листинг программы с комментариями

29

**ПРИЛОЖЕНИЕ Д**

(*обязательное*)

Ведомость документов

76