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 «Шахматы»

БГУИР КП 1–40 02 01 205 ПЗ

Студент: группы 250502,   
Бригадир А. С.

Руководитель: ассистент каф. ЭВМ Богдан Е. В.

Минск 2023

Учреждение образования

«Белорусский Учреждение образования

«Белорусский государственный университет информатики   
и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой

*––––––––––––––––––––––––*

(подпись)

––––––––––––––––––2023 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту     *Бригадир Анне Сергеевне –––––––––––––*

Тема проекта *Игра «Шахматы»* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Срок сдачи студентом законченного проекта––––*15 декабря 2023 г.–––*

3. Исходные данные к проекту*-пЯзык программирования – С++, среда разработки – Qt-Creator ------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------*

*––––––––––––––––––––––––––––––––––––––––––––––––––––––           —–––*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)

1. Лист задания.

2. Введение.

3. Обзор литературы.

3.1. Обзор методов и алгоритмов решения поставленной задачи.

4. Функциональное проектирование.

4.1. Структура входных и выходных данных.

4.2. Разработка диаграммы классов.

4.3. Описание классов.

5. Разработка программных модулей.

5.1. Разработка схем алгоритмов(два наиболее важных метода).

5.2. Разработка алгоритмов (описание алгоритмов по шагам, для двух методов).

6. Результаты работы.

7. Заключение

8. Литература

9. Приложения

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

*1. Диаграмма классов. ––––––––––––––––––––––––––––––*

*2. Схема структурная*

6. Консультант по проекту (с обозначением разделов проекта) Е. В. Богдан

7. Дата выдачи задания –––––*15.09.2023г.––––––––––––––––––––––   –*

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

*1. Выбор задания. Разработка содержания пояснительной записки. Перечень графического материала – 15 %;––––––––––––––––––––––––––––*

*разделы 2, 3 – 10 %;–––––––––––––––––––––––––––––––––––––*

*разделы 4 к –20 %;–––––––––––––––––––––––––––––––––––––––*

*разделы 5 к – 35 %;––––––––––––––––––––––––––––––––––––––*

*раздел 6,7,8 – 5 %;–––––––––––––––––––––––––––––––––––––––*

*раздел 9 к – 5%;–––––––––––––––––––––––––––––––––––––––––*

*оформление пояснительной записки и графического материала к 15.12.22 – 10 %*

*Защита курсового проекта с 21.12 по 28.12.23г.–––––––––––––––––––––––––*

РУКОВОДИТЕЛЬ Е.В. Богдан

(подпись)

Задание принял к исполнению *–––––––\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_––*

(дата и подпись студента)

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 6

1 ПОСТАНОВКА ЗАДАЧИ 7

2 ОБЗОР ЛИТЕРАТУРЫ 8

* 1. Анализ существующих аналогов 8
     1. Chess.com 8
     2. Chess for windows 9
  2. Требования к работе программы 10

3 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ 11

3.1 Приветственное меню приложения 11

3.2 Главное игровое окно 11

* 1. Конечное окно 11

1. ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ 13
   1. Входные и выходные данные 13
   2. Разработка диаграммы классов 14
   3. Описание классов.. 14
      1. Класс MainWindow 14
      2. Класс Game 15
      3. Класс Chesspiece 15
      4. Класс ChessBox 15
      5. Класс ChessBoard 15
      6. Класс Bishop 15
      7. Класс King 15
      8. Класс Knight 15
      9. Класс Pawn 15
      10. Класс Queen 15
      11. Класс Rook 15
2. РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ 23
   1. 23

ЗАКЛЮЧЕНИЕ 31

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 32

ПРИЛОЖЕНИЕ А Структурная схема 33

ПРИЛОЖЕНИЕ Б Диаграмма классов 34

ПРИЛОЖЕНИЕ В Листинг кода 37

ПРИЛОЖЕНИЕ Г Ведомость документов 68

**ВВЕДЕНИЕ**

.

C++ является одним из наиболее популярных языков программирования, который широко используется для разработки приложений, включая системное программное обеспечение, игры, веб-приложения и многое другое. Он был разработан в 1979 году Бьярном Страуструпом как расширение языка программирования C и предоставляет разработчикам мощные инструменты для создания различных типов приложений. Он поддерживает процедурное, объектно-ориентированное и обобщенное программирование, что делает его очень гибким и универсальным языком.

ООП - это парадигма программирования, которая позволяет организовать код в виде объектов, которые взаимодействуют друг с другом. Основными принципами ООП являются наследование, полиморфизм и инкапсуляция.

Наследование позволяет создавать иерархию классов, где дочерние классы наследуют свойства и методы от родительских классов. Это позволяет повторно использовать код и создавать более абстрактные и специализированные классы.

Полиморфизм позволяет использовать один и тот же интерфейс для работы с разными типами объектов. Это достигается с помощью виртуальных функций и переопределения методов в дочерних классах.

Инкапсуляция позволяет скрыть детали реализации класса и предоставить только необходимый интерфейс для работы с ним. Это достигается с помощью модификаторов доступа, таких как public, private и protected.

Qt — это библиотека классов C++ и набор инструментального программного обеспечения для создания кросс-платформенных приложений с графическим интерфейсом (GUI).

Qt позволяет запускать написанное с его помощью программное обеспечение в большинстве современных операционных систем путём простой компиляции программы для каждой системы без изменения исходного кода. Включает в себя все основные классы, которые могут потребоваться при разработке прикладного программного обеспечения, начиная от элементов графического интерфейса и заканчивая классами для работы с сетью, базами данных и XML. Является полностью объектно-ориентированным, расширяемым и поддерживающим технику компонентного программирования.

# 1 ПОСТАНОВКА ЗАДАЧИ

Тема курсового проекта была выбрана в первую очередь для получения знаний в области разработки десктопных приложений с применением ООП, для изучения возможностей Qt — библиотеки классов C++ и набора инструментального программного обеспечения для создания кросс-платформенных приложений с графическим интерфейсом.

Правила игры «Шахматы» включают в себя различные алгоритмы совершения хода, зависящие от вида фигуры, ее положения на доске, возможности создать ситуацию угрозы фигуре противника, конечной целью является взятие короля. Всего в шахматах есть 6 различных видов фигур.

Король - самая важная фигура, но при этом и одна из самых слабых. Король может ходить только на одно поле в любом направлении: вверх, вниз, в стороны и по диагонали. Король никогда не может вставать на битое поле (где его может взять фигура соперника). Когда король атакован другой фигурой, это называется "шах". На рисунках 1.1(а) и 1.1(б) представлены иконки фигур белого и черного короля.



Рисунок 1.1(а) – Иконка белого короля



Рисунок 1.1(б) – Иконка черного короля

Ферзь - самая сильная фигура. Он может ходить по прямой в любом направлении - вперёд, назад, в стороны или по диагонали на любое число полей, но при этом он не может перепрыгивать через другие фигуры. Если ферзь или любая другая фигура берёт фигуру соперника, ход заканчивается. На рисунках 1.2(а) и 1.2(б) представлены иконки фигур белого и черного ферзя.



Рисунок 1.2(а) – Иконка белого ферзя



Рисунок 1.2(б) – Иконка черного ферзя

Ладья может ходить на любое число полей, но только вперёд, назад и в стороны (не по диагонали). На рисунках 1.3(а) и 1.3(б) представлены иконки фигур белой и черной ладьи.



Рисунок 1.3(а) – Иконка белой ладьи



Рисунок 1.3(б) – Иконка черной ладьи

Слон может ходить на любое число полей, соединенных углами по диагонали. Каждому слону доступна только половина полей доски (одного цвета, белого или черного). На рисунках 1.4(а) и 1.4(б) представлены иконки фигур белого и черного слона.



Рисунок 1.4(а) – Иконка белого слона



Рисунок 1.4(б) – Иконка черного слона

Кони ходят иначе, чем другие фигуры - на два поля в одном направлении и далее на одно поле под углом 90 градусов, буквой "Г". Конь - единственная фигура, способная перепрыгивать через другие фигуры. На рисунках 1.5(а) и 1.5(б) представлены иконки фигур белого и черного коня.



Рисунок 1.5(а) – Иконка белого коня



Рисунок 1.5(б) – Иконка черного коня

Пешка - необычная фигура, она ходит и берёт по-разному: ходить пешка может лишь вперёд, а брать - лишь по диагонали. Пешка может передвигаться только на одно поле за ход, но еще не ходившая пешка может пойти вперёд на одно или два поля. Пешка может брать только по диагонали на одно поле перед собой. Пешка не может ходить или брать назад. Если другая фигура находится прямо перед пешкой, та не может сделать ход вперёд и не может взять эту фигуру. На рисунках 1.6(а) и 1.6(б) представлены иконки фигур белой и черной ладьи.



Рисунок 1.6(а) – Иконка белой пешки



Рисунок 1.6(б) – Иконка черной пешки

Цель игры - поставить мат королю соперника. На доске мат, когда один из игроков не может защититься от шаха. Защититься можно следующими способами:

* отступить на другое поле (но нельзя рокировать)
* закрыться от шаха другой фигурой
* или взять фигуру, напавшую на короля

Если король не может избежать мата - партия завершена.

**2 ОБЗОР ЛИТЕРАТУРЫ**

**2.1 Анализ существующих аналогов**

Для создания собственного работающего приложения, необходимо иметь представление об уже существующих аналогах.

* + 1. **Chess.com**

Chess.com – сайт для игры в шахматы онлайн, форум, шахматная социальная сеть, которая так же имеет свое десктопное приложение, которое позволяет играть в реальном времени с другими игроками, обучаться стратегиям и тактике, просматривать профессиональные партии и многое другое. На рисунке 2.1 показана часть интерфейса сайта. При нажатии на фигуру отображаются возможные для нее ходы. Есть возможность сыграть с искусственным интеллектом с разными уровнями сложности.



Рисунок 2.1 – Скриншот с сайта Chess.com

* + 1. **Chess for windows**

Игра «Шахматы» из официального приложения Microsoft Store. В данном приложении так же есть возможность сыграть с искусственным интеллектом разного уровня, выбрать цвет фигур пользователя в начале игры. Есть такая опция, как подсказка, которая делает следующий ход вместо игрока.

****

Рисунок 2.2 – Скриншот приложения «Chess for windows»

* 1. **Требования к работе программы**

После рассмотрения уже существующих аналогов игры «Шахматы» можно выделить основные задачи, которые необходимо выполнить для реализации собственного приложения. Для разработки десктопной игры «Шахматы» необходимо написать игровую логику шахмат и создать пользовательский интерфейс, который включает в себя саму доску с фигурами, отображение возможных ходов для каждый фигуры, очередь хода.

Для реализации игровой логики шахмат необходимо создать классы для шахматной доски и фигур, определить методы для проверки возможных ходов, наличия шаха, передвижения фигур.

Для написания приложения был выбран язык С++, так как необходимо использовать объектно–ориентированную парадигму, что делает удобным создание и взаимодействие нескольких окон. Полная информация о языке приведена в источниках [2] и [3].

Для выполнения графического интерфейса была выбрана Qt — библиотека классов C++ и набор инструментального программного обеспечения для создания кросс-платформенных приложений с графическим интерфейсом. Полная информация о QT приведена в источниках [1] и [4].

**3 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ**

Весь функционал приложения можно разбить на блоки для упрощения разработки и конечной реализации.

* 1. **Приветственное меню приложения**

Данное окно включает визуальное отображение приветствия пользователя, возможность начать игру или выйти из приложения.

Для создания и отображения графики главного меню можно использовать Qt Designer – графический инструмент для создания пользовательских интерфейсов, который предоставляет широкий выбор готовых виджетов и макетов, которые легко настраиваются и располагаются для создания нужного пользовательского интерфейса. Базовым классом для этого окна является класс Qt QMainWindow, который представляет собой виджет для создания основного окна приложения. Он обеспечивает основную структуру и функциональность для главного окна приложения, включая меню, панели инструментов, статусную строку и т. д.

* 1. **Главное игровое окно**

Главное игровое окно является основным в данном приложении. Оно включает в себя:

* Саму доску с фигурами;
* Отображение активного хода;
* Отображение фигур, которые выбыли из игры;
* Предупреждение о возможном шахе.

Визуальную составляющую функционала взаимодействия с пользователем (меню с кнопками и другие элементы управления) можно реализовать с помощью Qt Designer, а именно Widget Editing Mode, так как в режиме редактирования виджетов объекты можно перетаскивать из окна виджетов главного окна в форму, редактировать, изменять их размер, перетаскивать по форме. Таким же образом реализуется отображение времени и следующий ход. Так же с помощью данных классов и класса QGraphicsItems реализуется отображение графики фигур на доске.

* 1. **Конечное окно**

Конечное окно предназначено для уведомления пользователя о

победившей стороне. После окончания игры отображается начальное окно, где вместо кнопки «play» отображается надпись с победившей стороной.

Необходимая надпись устанавливается при поступлении сигналов игрового окна, после чего выполняются соответствующие слоты начального окна.

**4 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

В данном разделе описываются входные и выходные данные программы, диаграмма классов, а также приводится описание используемых классов и их методов.

**4.1 Входные и выходные данные**

Пользователь начинает игру нажатием на кнопку, выбирает ходы фигур путем нажатия клавиши мыши, выходными данными являются передвижения фигур, текстовые уведомления пользователя о выигрыше и проигрыше.

**4.2 Разработка диаграммы классов**

Диаграмма классов представлена в Приложении А, а также на Рисунке 4.1.

Рисунок 4.1 – Диаграмма классов игры.

* 1. **Описание классов**
     1. **Класс MainWindow**

Собственный класс MainWindow наследует класс QMainWindow – это класс в библиотеке Qt, который содержит основные элементы для создания и управления главным окном приложения, в данном случае пользовательское меню.

Поля:

MainWindow \*ui; - поле дизайн-макета главного окна

Методы:

MainWindow(QWidget \*parent = nullptr); - конструктор класса, который устанавливает в окне дизайн-макет

~MainWindow(); - деструктор класса, который очищает в окне установленный дизайн-макет

Слоты:

void on\_buttonEXIT\_clicked(); - функция закрытия окна при поступлении соответствующего сигнала

void on\_buttonPVP\_clicked(); - функция открытия игрового окна при поступлении соответствующего сигнала

void whiteWON(); - функция вызова финального окна при поступлении соответствующего сигнала

void blackWON(); - функция вызова финального окна при поступлении соответствующего сигнала

* + 1. **Класс Game**

Класс **Game –** класс основного цикла игры, в котором происходит начальная отрисовка сцены, которая включает шахматную доску,, фигуры, поля для выбывших фигур, отображение очереди хода и наличия шаха.

Поля:

QGraphicsScene \*gameScene; – графическая сцена игрового окна

QList <ChessPiece \*> whiteDead – список объектов класса

ChessPiece, хранящий все выбывшие белые фигуры

QList <ChessPiece \*> blackDead – список объектов класса

ChessPiece, хранящий все выбывшие черные фигуры

QGraphicsRectItem \* deadHolder; – графический объект боковых

полей для отображения выбывших фигур и основного поля шахматной доски

QGraphicsRectItem \* pawnMenu;

QString turn; – очередь хода

ChessBoard \*chess; – графический объект класса ChessBoard

QList <QGraphicsItem \*> listG; – список графических объектов

QGraphicsTextItem \* turnDisplay; – графический текстовый объект для отображения очереди хода

ChessBox \*collection[8][8]; –массив объектов класса ChessBox,

хранящий все ячейки доски

QList <ChessPiece \*> alivePiece; – список объектов класса

ChessPiece оставшихся фигур

ChessPiece \*pieceToMove; – объект класса ChessPiece, активная фигура

QGraphicsTextItem \* choosePawnText; – графический текстовый объект для отображения меню пешки

Методы:

Game(QWidget \*parent = 0) – конструктор класса, который устанавливает в окне такие графические объекты, как сама графическая сцена, текстовые объекты очереди хода и наличия шаха

void drawChessBoard(); – метод добавления шахматной доски,

боковых панелей для выбывших фигур

void displayDeadWhite(); – отображение списка объектов класса

ChessPiece выбывших белых фигур

void displayDeadBlack(); – отображение списка объектов класса

ChessPiece выбывших черных фигур

void addToScene(QGraphicsItem \*item); – метод для добавления в

графическую сцену объекта класса QGraphicsItem

void removeFromScene(QGraphicsItem \*item); – метод для

удаления из графической сцены объекта класса QGraphicsItem

QString getTurn()– метод, возвращающий очередь хода

void setTurn( QString value); – метод, устанавливающий очередь

хода

void changeTurn(); – смена хода на противоположную

void drawDeadHolder(int x, int y,QColor color)– метод для

отрисовки зон для выбывших фигур и шахматной доски

void drawPawnMenu()– метод отрисовки меню пешки, дошедшей до

конца доски

void gameOver(); – окончание игры

void removeAll(); – прекращение отображения всех объектов

void start(); – начало игры, добавление в сцену всех необходимых

объектов, начало их отображения, добавление всех фигур

* + 1. **Класс Chesspiece**

Класс **Chesspiece –** базовыйкласс шахматной фигуры

* + 1. **Класс ChessBox**

Класс **ChessBox –** класс графического отображения ячеек.

Описание класса:

public:

ChessBox(QGraphicsItem\* parent = 0); – отображение прямоугольника

~ChessBox(); – деструктор

void mousePressEvent(QGraphicsSceneMouseEvent\* event); – обработка нажатий мыши

void setColor(QColor color); – установка цвета

void placePiece(ChessPiece\* piece); – положение ячейки

void resetOriginalColor(); – сброс цвета ячейки

void setOriginalColor(QColor value); – восстановление цвета ячейки

QString getChessPieceColor(); – получение цвета ячейки

void setChessPieceColor(QString value); – установка цвета фигуры в ячейке

private:

QColor originalColor; – цвет ячейки

bool hasChessPiece; –смена цвет ячейки

QString chessPieceColor; – цвет фигуры в ячейке

* + 1. **Класс ChessBoard**

Класс **ChessBoard –** класс графического отображения ячеек.

Описание класса:

public:

ChessBoard();– конструктор, установка всех фигур

void drawBoxes(int x, int y); – отображение ячейки

void setUpWhite(); – установка всех белых фигур

void setUpBlack(); – установка всех черных фигур

void addChessPiece(); – изменение полей количества фигур

private:

QList <ChessPiece\*> white; – список белых ячеек

QList <ChessPiece\*> black; – список белых ячеек

* + 1. **Класс**

Класс **ChessPiece** **–** базовыйкласс фигуры.

Описание класса:

public:

void mousePressEvent(QGraphicsSceneMouseEvent\* event); – обработка нажатий мыши

virtual void setImage() = 0; – установка изображения фигуры

bool getIsPlaced(); – проверка на установку фигуры

void setIsPlaced(bool value); – проверка на установку фигуры

QList <ChessBox\*> moveLocation(); – список перемещений

virtual void moves() = 0; – перемещение

bool firstMove; – первый ход

Класс **Button** **–** базовыйкласс фигуры.

Описание класса:

public:

Button(QString name, QGraphicsItem\* parent = NULL); – конструктор

void mousePressEvent(QGraphicsSceneMouseEvent\* event); – обработка нажатий мыши

void hoverEnterEvent(QGraphicsSceneHoverEvent\* event); – обработка нажатий клавиши Enter

void hoverLeaveEvent(QGraphicsSceneHoverEvent\* event); – обработка прекращения нажатий

private:

QGraphicsTextItem\* text; – надпись

Класс **Bishop** **–**класс фигуры слона, наследует базовый класс фигуры.

Описание класса:

public:

Bishop(QString team, QGraphicsItem\* parent = 0); – установка фигуры

void setImage(); – установка изображения фигуры

void moves(); – ход фигуры

Класс **King** **–**класс фигуры короля, наследует базовый класс фигуры.

Описание класса:

public:

King(QString team, QGraphicsItem\* parent = 0); – установка фигуры

void setImage(); – установка изображения фигуры

void findUnSafeLocation(); – поиск позиции шаха

void moves(); – ход фигуры

Класс **Knight** **–**класс фигуры коня, наследует базовый класс фигуры.

Описание класса:

public:

Knight(QString team, QGraphicsItem\* parent = 0); – установка фигуры

void setImage(); – установка изображения фигуры

void moves(); – ход фигуры

Класс **Pawn** **–**класс фигуры пешки, наследует базовый класс фигуры.

Описание класса:

public:

Pawn(QString team, QGraphicsItem\* parent = 0); – установка фигуры

void setImage(); – установка изображения фигуры

void moves(); – ход фигуры

Класс **Queen** **–**класс фигуры королевы, наследует базовый класс фигуры.

Описание класса:

public:

Queen(QString team, QGraphicsItem\* parent = 0); – установка фигуры

void setImage(); – установка изображения фигуры

void moves(); – ход фигуры

Класс **Rook** **–**класс фигуры королевы, наследует базовый класс фигуры.

Описание класса:

public:

Rook(QString team, QGraphicsItem\* parent = 0); – установка фигуры

void setImage(); – установка изображения фигуры

void moves(); – ход фигуры

**5 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

В данном разделе рассмотрены описания алгоритмов, используемых в программе.

**ЗАКЛЮЧЕНИЕ**

Данный курсовой проект

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Qt 5.10 Профессиональное программирование на С++, – Макс Шлее, 2018
2. Объектно-ориентированное программирование в С++, – Роберт Лафоре, 2019
3. Язык программирования С++, – Бьерн Страуструп, 1985
4. Qt Documentation [Электронный ресурс]. -Электронные данные.  
   -Режим доступа: <https://doc.qt.io/> - Дата доступа: 20.10.2023
5. Chess.com [Электронный ресурс]. -Электронные данные.– Режим доступа: <https://www.chess.com/> - Дата доступа: 20.09.2023

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Схема структурная**

**ПРИЛОЖЕНИЕ Б**

(обязательное)

**Диаграмма классов**

**ПРИЛОЖЕНИЕ В**

(обязательное)

**Ведомость документов**