**Санкт-Петербургский государственный университет**

**Факультет прикладной математики и процессов управления**

**ОТЧЁТ**

**по дисциплине “Системное программирование в Linux”**

**о создании шахматной программы**

Автор работы: Горбунов К. Д. Группа: 22.Б16-пу

Преподаватель: Киямов Ж. У.

Санкт-Петербург

2023 г.

**Аннотация**

Данная программа представляет собой реализацию шахматной игры с оптимизацией ходов для белых игроков. Программа определяет наилучшие последовательности ходов, стремясь максимизировать суммарное количество очков на доске, основываясь на весе каждой фигуры. Результаты представлены в виде последовательностей ходов и общего количества очков.

**Содержание**

[**Аннотация** 2](#_Toc147273280)

[**1. Введение** 4](#_Toc147273281)

[**2. Цель работы** 4](#_Toc147273282)

[**3. Задачи** 4](#_Toc147273283)

[**4. Техническое описание** 5](#_Toc147273284)

[**4.1 Используемые инструменты и библиотеки** 5](#_Toc147273285)

[**5 Описание программы** 5](#_Toc147273286)

[**5.1 Описание ключевых компонентов программы** 5](#_Toc147273287)

[**5.2 Общий ход программы** 7](#_Toc147273287)

[**6. Рекомендации пользователю** 9](#_Toc147273284)

[**7. Рекомендации программисту** 10](#_Toc147273286)

[**8. Вывод** 10](#_Toc147273284)

[**9. Библиография** 11](#_Toc147273284)

**1. Введение**

Шахматы - это интеллектуальная настольная игра, требующая стратегического мышления и расчета ходов. Целью данной программы является разработка простой шахматной игры с оптимизацией ходов, чтобы выявить лучшие ходы для белых игроков на основе веса каждой фигуры на доске.

**2. Цель работы**

Целью данной работы является разработка программы, способной определить оптимальные последовательности ходов для белых игроков, чтобы максимизировать общее количество очков на доске. Это представляет интерес как для улучшения стратегического мышления в шахматах, так и для дальнейшего развития игровых алгоритмов.

**3. Задачи**

1. Инициализация шахматной доски и размещение фигур на начальных позициях для белых и черных игроков.
2. Определение возможных ходов для каждой фигуры, учитывая правила шахмат.
3. Оценка веса каждой фигуры на доске в соответствии с их значимостью: пешка - 1, слон и конь - 3, ладья - 5, король - 1.
4. Поиск оптимальных последовательностей ходов для обоих игроков с учетом максимизации суммарного веса фигур на доске.
5. Отображение результатов в виде последовательностей ходов и общего количества очков.

**4. Техническое описание**

**4.1 Используемые инструменты и библиотеки**

Для разработки программы были использованы следующие инструменты и библиотеки:

Язык программирования: C++;

Библиотеки: iostream, string, vector, random, algorithm.

**4.2 Структура программы**

Программа состоит из следующих основных компонентов:

1. Структура данных Piece: Описывает характеристики шахматной фигуры, включая имя, цвет и координаты на доске.
2. Класс ChessBoard: Представляет шахматную доску и фигуры на ней. Включает методы для инициализации доски, вывода её состояния и определения возможных ходов.
3. Основная функция main: Главная функция программы, где создается объект класса ChessBoard, инициализируется доска и фигуры, и происходит игра, опираясь на случайные ходы фигур.
4. Алгоритм оптимизации ходов: В конце программы анализируются оптимальные ходы для белых фигур, и выводятся наилучшие ходы для нескольких последовательных ходов.

**5** **Описание программы**

**5.1 Описание ключевых компонентов программы**

В таблице 1 представлено описание ключевых компонентов программы.

Таблица 1. Ключевые компоненты

|  |  |
| --- | --- |
| Компонент | Описание |
| Piece (Структура) | Представляет фигуру на шахматной доске. Содержит атрибуты: - name: Имя фигуры (например, "pawn", "rook"). color: Цвет фигуры (белый или черный). - y и x: Координаты фигуры на доске (строка и столбец). |
| ChessBoard (Класс) | Управляет состоянием шахматной доски и фигурами на ней. - Включает атрибуты: - board: Двумерный вектор, представляющий текущее состояние доски с расположенными фигурами. - white и black: Векторы, содержащие фигуры белых и черных игроков. - Основные методы: - make(): Инициализирует доску и фигуры. - printBoard(): Выводит текущее состояние доски. - move(Piece figure): Определяет возможные ходы для фигуры. |
| main (Главная функция) | Управляет выполнением программы. Создает экземпляр ChessBoard. Инициализирует игровую доску и фигуры. Выполняет ходы фигур с учетом случайного порядка. Производит анализ и оптимизацию последовательности ходов для белых фигур. |
| Алгоритм оптимизации | Оптимизирует последовательность ходов для белых фигур. Учитывает весовые характеристики фигур для выбора оптимальных ходов. Выводит оптимальную последовательность ходов и оценку их эффективности. |

**5.2 Общий ход программы**

Общий ход программы можно описать следующим образом:

1. **Инициализация шахматной доски и фигур:**
   * Создается экземпляр класса ChessBoard, который представляет шахматную доску и фигуры.
   * В методе make() инициализируется доска и размещаются фигуры на начальных позициях для белых и черных игроков.
2. **Вывод начального состояния доски:**
   * Вызывается метод printBoard(), который выводит на экран начальное состояние доски, отображая расположение фигур.
3. **Основной игровой цикл:**
   * Пользователь вводит количество ходов (steps), которые программа будет выполнять.
4. **Случайные ходы фигур:**
   * В цикле происходит выполнение ходов для белых и черных фигур поочередно.
   * Фигуры перемешиваются случайным образом (путем перемешивания векторов фигур).
   * Для каждой фигуры определяются её возможные ходы с помощью метода move().
   * Случайно выбирается один из доступных ходов для фигуры, и фигура перемещается на новую позицию.
   * Фигуры снимаются с доски, если они были взяты противником.
5. **Оптимизация ходов для белых фигур:**
   * После завершения случайных ходов, происходит анализ последовательности ходов белых фигур.
   * Выполняется оптимизация последовательности ходов на основе весовых характеристик фигур.
   * Оптимальная последовательность ходов и оценка их эффективности сохраняются.
6. **Анализ и завершение игры:**
   * Производится анализ текущего состояния доски и фигур.
   * Проверяется, остались ли на доске фигуры обоих цветов. Если нет, игра завершается.
   * Проверяется, находятся ли короли противников под боем. Если да, игра продолжается; если нет, игра завершается.
7. **Вывод оптимальной последовательности ходов:**
   * Выводится оптимальная последовательность ходов для белых фигур, а также оценка их эффективности.

На рис. 1 приведено окно пользователя.

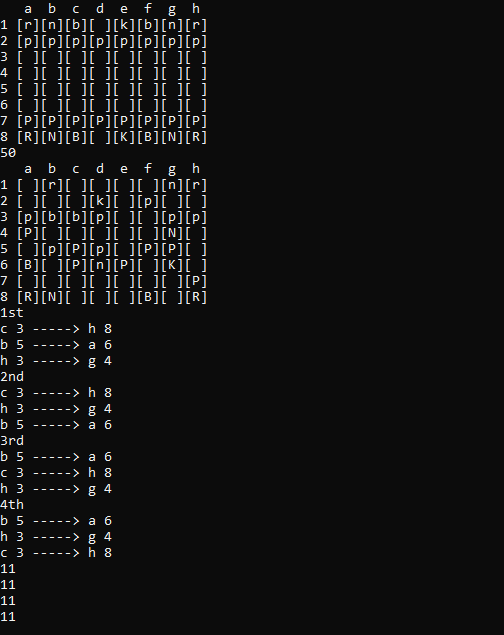


Рис. 1 Окно пользователя

**6. Рекомендации пользователю**

**Знакомство с шахматами:** Прежде чем использовать программу, рекомендуется ознакомиться с правилами шахмат, чтобы лучше понимать процесс и результаты выполнения программы.

**Изучение вывода:** Внимательно анализируйте вывод программы, особенно оптимальную последовательность ходов для белых фигур. Это может помочь вам лучше понимать стратегию игры.

**Настройка количества ходов:** Вы можете настроить количество ходов, которые программа выполняет. Экспериментируйте с этим параметром, чтобы получить разные результаты.

**7. Рекомендации программисту**

**Установите компилятор C++:** Если у вас его еще нет, установите компилятор C++, используя менеджер пакетов вашего дистрибутива Linux. Например, для Ubuntu это можно сделать следующей командой: sudo apt-get install g++.

**Склонируйте репозиторий:** Склонируйте репозиторий с программой на свой компьютер с помощью команды git clone: git clone https://github.com/Gorbunov-Konstantin/Chess.git

**Перейдите в каталог с программой:** Используя терминал, перейдите в каталог, в который была склонирована программа: cd chess\_program.

**Компилируйте программу:** Выполните компиляцию программы с помощью компилятора g++: g++ -o chess\_program chess.cpp. Это создаст исполняемый файл chess\_program.

**Запустите программу:** Запустите скомпилированную программу, введя следующую команду: ./chess\_program

**Ввод количества ходов:** Программа запросит вас ввести количество ходов (целое число), которое необходимо выполнить. Введите это число и нажмите "Enter".

**Анализ результата:** Программа выполнит указанное количество ходов и выдаст результат.

**8. Вывод**

В ходе разработки данной программы для анализа игры в шахматы была создана функциональная система, способная моделировать игру и анализировать ходы на основе стратегии. Программа позволяет провести анализ, определяя оптимальные ходы и стратегии игры.

Технически программа реализована на языке C++ и использует стандартную библиотеку STL. Она предоставляет пользователю возможность ввести количество ходов и выполняет их с учетом случайных факторов. В конечном итоге, программа выводит оптимальные ходы и стратегии, которые могут привести к выигрышу.

Эта программа может быть полезна для людей, интересующихся стратегическими настольными играми, такими как шахматы, и желающих изучить различные стратегии игры. Она также может быть использована как образовательный инструмент для анализа и обучения в шахматах.

**Библиография**

Бесплатная онлайн-платформа для шахмат с множеством ресурсов и API для разработчиков - [lichess.org](https://lichess.org/)

Вики-ресурс, посвященный программированию шахмат - https://www.chessprogramming.org/Main\_Page

Репозитории, связанные с программированием шахмат на GitHub - https://github.com/search?q=chess+programming

GitHub - https://github.com/Gorbunov-Konstantin/Chess.git