Санкт-Петербургский государственный университет

Факультет прикладной математики – процессов управления

**Лабораторная работа №3**

**Отчет**

**по дисциплине «Системное программирование в Linux»**

Автор работы: Дацык Р.В.

Группа: 22.Б15-пу

Преподаватель: Дик А.Г.

Санкт-Петербург, 2023

**Оглавление**

[**1.** **Цель работы** 3](#_Toc151824592)

[**2.** **Задачи** 3](#_Toc151824593)

[**3.** **Введение** 3](#_Toc151824594)

[**4.** **Теоретическая часть** 3](#_Toc151824595)

[**5.** **Алгоритм метода** 5](#_Toc151824596)

[**6.** **Описание программы** 5](#_Toc151824597)

[**7.** **Рекомендации пользователю** 7](#_Toc151824598)

[**8.** **Рекомендации программисту** 7](#_Toc151824599)

[**9.** **Контрольный пример** 8](#_Toc151824600)

[**10.** **Вывод** 9](#_Toc151824601)

[**11.** **Список использованной литературы** 9](#_Toc151824602)

# **Цель работы**

Разработка программного решения, которое генерирует случайное начальное положение на шахматной доске и вычисляет наилучшие ходы в данной позиции, с целью обеспечить анализ и оптимизацию шахматных стратегий.

# **Задачи**

1. Разработка алгоритма для генерации случайного начального положения на шахматной доске.
2. Разработка алгоритма для вычисления наилучших ходов в заданной шахматной позиции.
3. Интеграция разработанных алгоритмов в программное решение.
4. Проведение анализа и оптимизация шахматных стратегий на основе результатов программы.

# **Введение**

Шахматы – это стратегическая настольная игра, которая требует от игроков планирования и тактического мышления. Эта программа предоставляет инструмент для анализа шахматных стратегий, оценки возможных вариантов развития игры и выявления наилучших решений для игрока. Подход, используемый в программе, обеспечивает анализ различных комбинаций ходов и выявление оптимальных ходов для достижения желаемого результата.

# **Теоретическая часть**

* **Шахматная доска:**
  + Доска представляется двумерным вектором (двумерным массивом), где каждая клетка может содержать информацию о фигуре, стоящей на ней.
* **Генерация возможных ходов:**
  + Для каждой фигуры реализованы функции, генерирующие возможные ходы в соответствии с правилами шахмат.
  + Реализованы функции для пешек, коней, слонов, ладей и королей, каждая из которых возвращает вектор ходов.
* **Оценка ходов:**
  + Каждый ход оценивается в баллах в зависимости от типа хода и его влияния на игровую ситуацию.
  + Оценка включает в себя бонусы за шах, захват фигур оппонента, а также специфичные баллы за каждый тип фигуры.
  + При оценке учитывается возможность атаки короля.
* **Выбор лучших ходов:**
  + Для каждого хода расчитывается оценка, и топ-3 хода сохраняются для последующего применения.
  + При равенстве баллов предпочтение отдается первому лучшему ходу.
* **Применение лучших ходов:**
  + Для каждого лучшего хода производится обновление доски.
  + Выводится информация о ходе, его типе, координатах начальной и конечной клеток.
* **Цикл игры:**
  + Игра выполняется в цикле, в котором генерируются и применяются лучшие ходы.
  + Выводится информация о текущем состоянии доски после каждого хода.

# **Алгоритм метода**

1. **Генерация возможных ходов:**
   1. Перебор всех клеток на доске.
   2. Для каждой непустой клетки:
      1. Определение цвета фигуры (белая или чёрная).
      2. В зависимости от типа фигуры (пешка, конь и так далее) вызывается соответствующая функция для генерации возможных ходов этой фигуры.
      3. Полученные ходы добавляются в общий список всех ходов.
2. **Оценка ходов:**
   1. Создание вектора для хранения топ-3 лучших ходов, начально инициализированных нулевыми баллами.
   2. Вызов функции для генерации всех возможных ходов.
   3. Перебор всех ходов и сравнение их оценок.
   4. Обновление топ-3 ходов, если текущий ход имеет большую оценку.

# **Описание программы**

Алгоритм реализован на языке С++ с использованием следующих пакетов:<iostream>, <vector>, <cstdlib>, <ctime>, <string>

Ниже представлена таблица описывающая функции, которые были использованы в программе

Таблица 6.1. Описание функций

|  |  |
| --- | --- |
| Название функции | Назначение функции |
| ChessBoard | (Конструктор): Создает объект доски для шахмат, инициализирует пустую доску размером 8x8. |
| printBoard | Выводит текущее состояние шахматной доски в текстовом виде, отображая положение каждой фигуры. |
| isPositionValid | Проверяет, является ли указанная позиция (строка и столбец) допустимой для размещения фигуры. |
| isKingUnderAttack | Проверяет, находится ли король под атакой на доске. |
| canPieceAttackKing | Проверяет, может ли указанная фигура атаковать короля на доске. |
| generateRandomPosition | Генерирует случайное начальное распределение фигур на доске. |
| applyMove | Применяет ход к доске, обновляя ее состояние. |
| generatePawnMoves | Генерирует возможные ходы для пешки. |
| generateKnightMoves | Генерирует возможные ходы для коня. |
| generateBishopMoves | Генерирует возможные ходы для слона. |
| generateRookMoves | Генерирует возможные ходы для ладьи. |
| generateKingMoves | Генерирует возможные ходы для короля. |
| generatePossibleMoves | Генерирует все возможные ходы для всех фигур на доске. |
| findBestMoves | Находит топ-3 лучших хода на основе их оценки в баллах. |
| findAndMakesMoves | Находит и выполняет указанное количество лучших ходов, основываясь на оценке хода. |
| getPieceSymbol | Возвращает строковое представление символа фигуры для вывода в консоль. |
| convertToChessNotation | Преобразует координаты строки и столбца в шахматную нотацию (например, "a1", "e5"). |
| my\_rand | Генерирует псевдослучайное число на основе линейного конгруэнтного метода. |

# **Рекомендации пользователю**

Для успешного запуска программы необходимо устройство с операционной системой Linux, macOS или Windows, а также среда разработки, поддерживающей запуск компилятора С++

При запуске программы у пользователя запрашивается количество фигур на доске. После ввода пользователю предоставляется изначальная доска с указанием сгенерированных фигур и 3 атаки по 3 последовательных хода, в каждом из которых указана атакующая фигура, а также ее стартовая и конечная позиции.

# **Рекомендации программисту**

Для запуска программы необходима 64-битная операционная система Windows, Linux или macOS. Для работы с кодом необходима среда разработки, совместимая с компилятором С++ и библиотеки <iostream>, <ctime>, <cstdlib> и другие.

Исходный код программы и доступен по ссылке:

<https://github.com/CapTopGrade/Linux/tree/main/Chess>

# **Контрольный пример**

В данном разделе представлен контрольный пример, демонстрирующий работу программы.

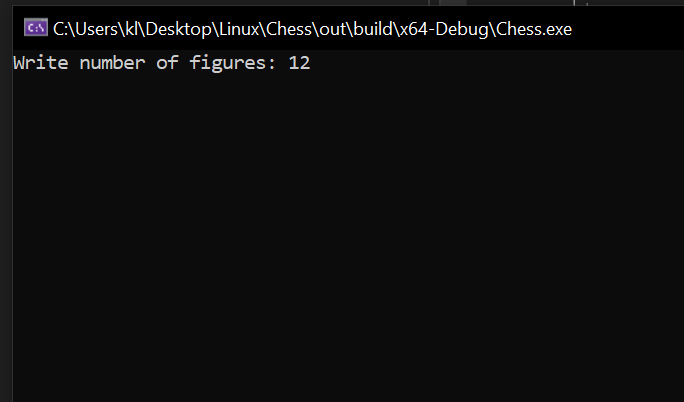


Рис. 9.1 Пример ввода данных

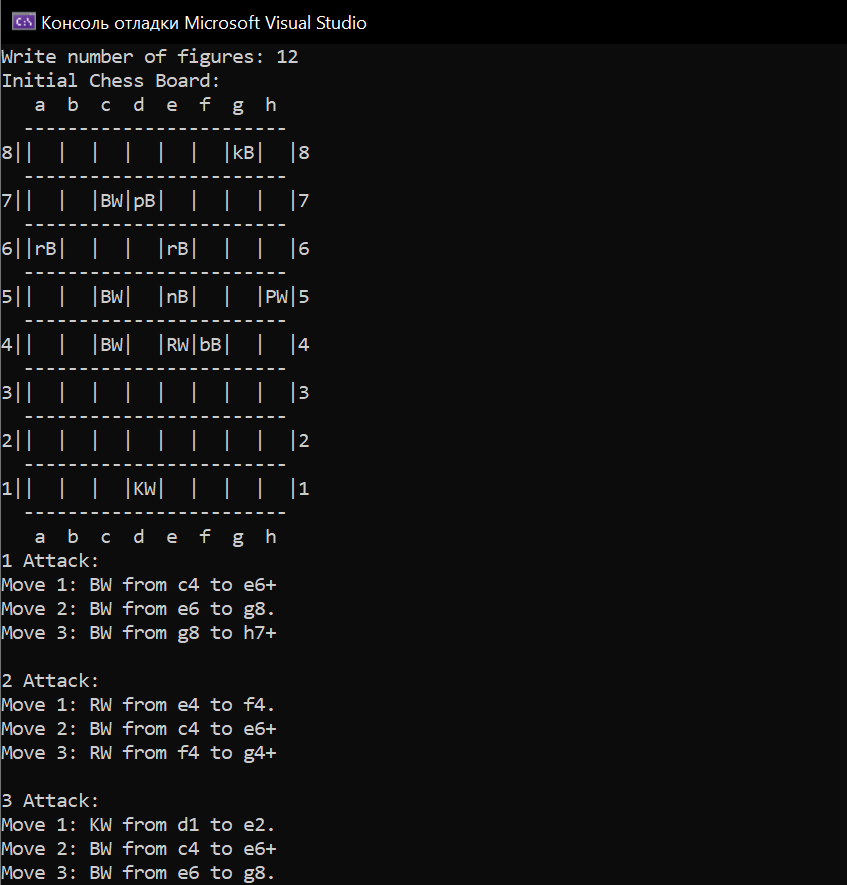


Рис. 9.2 Пример вывода программы

# **Вывод**

В результате выполнения данной работы была разработана программа, способная автоматически создавать начальное положение шахматных фигур на доске и определять три наилучших хода в данной игровой ситуации.

# **Список использованной литературы**

[1] Документация по работе с С++: <https://ru.cppreference.com/w/>

[2] Рекомендации по улучшению кода: <https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md>