**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра «Измерительно-вычислительные комплексы»

**Курсовая работа**

**По дисциплине «Алгоритмы и структуры данных»**

**Тема «Компьютерная логическая игра «Русские 80-клеточные шашки»**

**Пояснительная записка**

Р.02069337. 21/849-01 ПЗ-01

Листов 12

**Руководитель разработки**:

доцент каф. ИВК, к.т.н., доцент

*Шишкин Вадим Викторинович*

« » 2024 г.

**Исполнитель**:

студент гр. ИСТбд-32

*Абляев Глеб Сергеевич*

« » 2024 г.

**2024**

**Введение**

Реализована игра «Русские шахматы»

Шашки — это абстрактная стратегическая настольная игра для двух игроков, играющих на доске с клетками, состоящей из чередующихся светлых и темных квадратных ячеек. Целью игры является захват или блокировка всех шашек противника.

Игра шашки имеет древнее происхождение и широко распространена по всему миру. Она предоставляет игрокам возможность развивать стратегическое мышление, тактику и аналитические способности.

Шашки включают в себя не только базовые правила, но и сложные тактические методы и стратегии, такие как создание дамок, блокировка противника, проведение комбинаций и атака на ключевые позиции на игровом поле.

Программирование игры шашки не только демонстрирует понимание основных концепций игры и алгоритмов, но и представляет собой отличный способ практического применения знаний в области компьютерной графики, пользовательского интерфейса и разработки искусственного интеллекта.

**Проектная часть**

**1.1 Постановка задачи на разработку приложения**

Целью проекта является разработка приложения для игры в шашки. Основная задача заключается в создании программного продукта, который позволит пользователям играть в шашки против компьютера или друг против друга. Техническое задание на разработку приложения подробно описано в соответствующем документе.

**1.2 Математические методы**

В данном проекте математические методы не применяются, поскольку основной акцент делается на разработку игровой логики, пользовательского интерфейса и искусственного интеллекта для компьютерного противника.

**1.3 Архитектура и алгоритмы**

1.3.1 Архитектура

В архитектуре приложения выделяются следующие основные компоненты:

Piece (Шашка): Представляет из себя пешку на игровом поле. Содержит информацию о своих координатах как относительно игровой матрицы, так и относительно окна. Также включает функции для пересчета координат и отрисовки шашки, а также функцию для выполнения хода (изменение координат).

Board (Доска): Содержит игровую матрицу (двумерный список), представляющую игровое поле. Включает функции для создания доски, отрисовки её на экране, выполнения хода шашки (изменения её координат в матрице), поиска пешек, которыми обязательно нужно ходить, поиска обязательных ходов, поиска необязательных ходов и определения победителя.

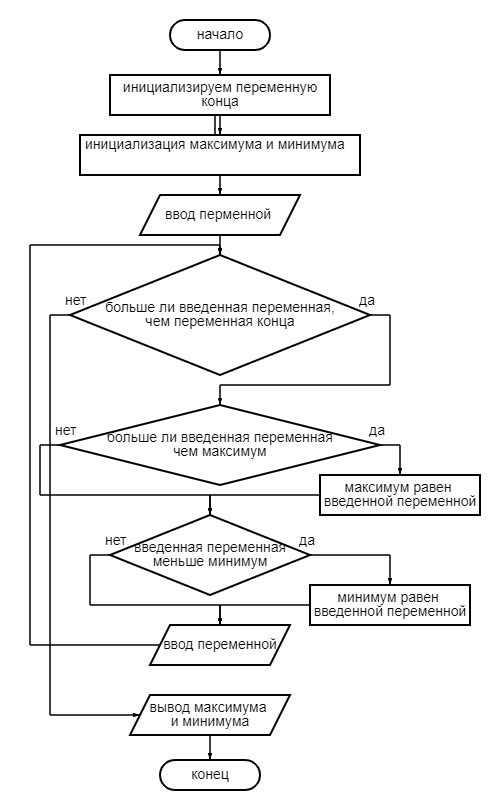
Game (Игра): Содержит информацию о текущем ходе, объект доски, доступные ходы и пользовательский интерфейс (GUI). Включает функции для определения победителя, выбора клетки для хода, смены очереди игрока и выполнения хода шашки.

Архитектура приложения предполагает четкое разделение ответственностей между компонентами, что позволяет обеспечить более гибкую и поддерживаемую структуру кода.

Алгоритм Минимакс

Алгоритм Минимакс - это классический алгоритм принятия решений в играх, основанный на принципе "минимум-максимум". Алгоритм широко применяется в задачах искусственного интеллекта для поиска оптимального хода в играх с нулевой суммой, таких как шашки, шахматы и другие.

Принцип работы алгоритма минимакс заключается в том, что он рекурсивно оценивает возможные ходы игроков на несколько ходов вперед, чтобы определить наилучший ход для текущего игрока и наихудший для его противника. Для этого алгоритм использует дерево возможных игровых состояний, где каждый узел представляет собой конкретное состояние игры, а рёбра - возможные ходы.



Алгоритм минимакс работает в двух режимах:

Максимизация: когда алгоритм выбирает ход для максимизации выигрыша текущего игрока.

Минимизация: когда алгоритм выбирает ход для минимизации потерь противника.

На каждом уровне рекурсии алгоритм переключается между этими двумя режимами, чтобы просчитать все возможные ходы и выбрать наилучший из них. По мере расширения дерева игровых состояний, алгоритм оценивает каждый возможный ход с помощью функции оценки (эвристики), которая оценивает выигрыш для текущего игрока.

1.3.4 Алгоритм создания доски

Этот алгоритм ответственен за создание игровой доски, её отрисовку и заполнение. Он создаёт двумерный массив (матрицу), представляющий собой игровое поле, и заполняет его начальным расположением шашек. Затем алгоритм отрисовывает доску в графическом интерфейсе пользователя (GUI), обеспечивая правильное отображение на экране.

1.3.5 Алгоритм поиска обязательных ходов

Этот алгоритм осуществляет поиск обязательных ходов для определенной шашки. Он анализирует текущее состояние игры и определяет, есть ли у какой-либо из шашек обязательные ходы, которые необходимо выполнить в текущем ходе. Для этого алгоритм проверяет возможные ходы каждой шашки и выявляет те из них, которые обязательны для выполнения.

1.3.6 Алгоритм поиска пешек с обязательными ходами

Этот алгоритм осуществляет поиск шашек, у которых есть обязательные ходы. Он проходит по всем пешкам на игровой доске и определяет, есть ли у них обязательные ходы. Если такие ходы обнаруживаются, шашки добавляются в список, который затем используется для выполнения обязательных ходов.

1.3.7 Алгоритм поиска всех пешек

Этот алгоритм осуществляет поиск всех шашек на игровой доске. Он проходит по всем клеткам игрового поля и определяет, находится ли на каждой из них шашка определенного игрока. Затем он составляет список всех найденных шашек для дальнейшего использования в других алгоритмах.

1.3.8 Алгоритм поиска ходов пешки

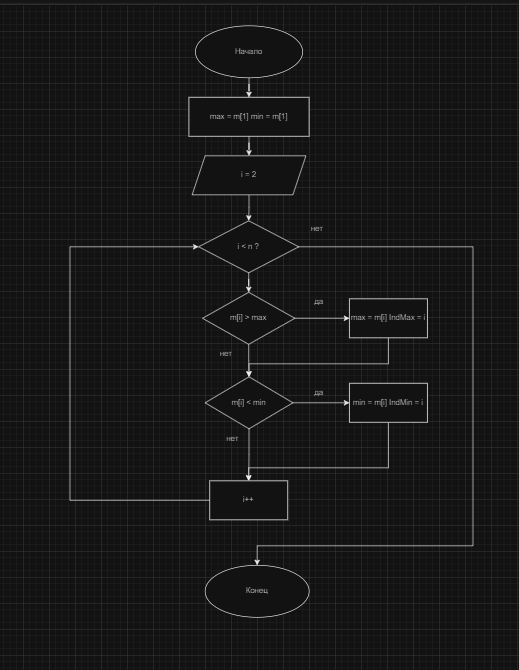
Этот алгоритм осуществляет поиск всех возможных ходов для определенной шашки, включая необязательные ходы, то есть те, которые не предполагают битья других шашек. Он анализирует текущее положение пешки на доске и определяет все возможные ходы в рамках правил игры.

1.3.9 Алгоритм выбора шашки/хода шашки

Этот алгоритм учитывает обязательные ходы и помогает игроку выбрать шашку, которой нужно сделать ход. Если у игрока есть возможность сделать обязательный ход, этот алгоритм помогает определить, какую шашку следует использовать для выполнения этого хода.

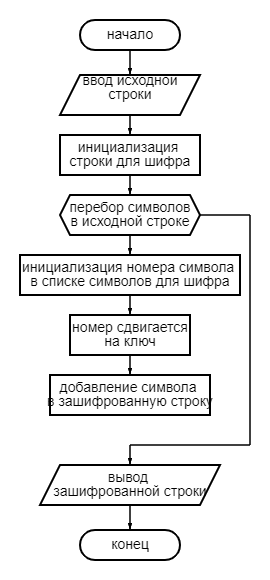
1.3.10 Алгоритм хода шашки

Этот алгоритм осуществляет перемещение пешки по игровой доске в соответствии с выбранным ходом. Он меняет положение шашки, удаляет битые шашки противника (если таковые есть) и переключает ход к следующему игроку.



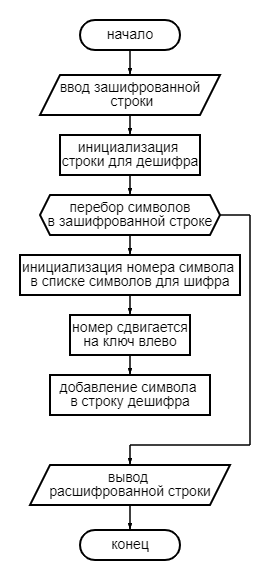
1.3.11 Алгоритм шифрования Цезаря

Алгоритм реализует шифрование пароля при регистрации.



1.3.11 Алгоритм дешифрования Цезаря

Алгоритм реализует дешифрование пароля при регистрации.



1.4 Тестирование

Для проверки реализованного приложения было проведено ручное тестирование. Исходный код был также проверен на наличие неиспользуемых переменных, ошибок, связанных с логикой игры, а также на ошибки форматирования введенных данных пользователем.

Для тестирования были выбраны два распространенных сценария, которые часто приводят к ошибкам:

Сценарий: Неверный ввод данных пользователем.

Действие пользователя: Ввод пустых данных.

Ожидаемый результат: Пользователь получит сообщение "INCORRECT LOGIN OR PASSWORD."

Фактический результат: Пользователь получает сообщение "INCORRECT LOGIN OR PASSWORD."

Сценарий: Попытка авторизации несуществующего пользователя.

Действие пользователя: Попытка авторизации с несуществующим аккаунтом.

Ожидаемый результат: Пользователь получит сообщение "INCORRECT LOGIN OR PASSWORD."

Фактический результат: Пользователь получает сообщение "INCORRECT LOGIN OR PASSWORD."

**2. Источники, использованные при разработке**

1. Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с. ( Дата обращения: 17.03.2024)
2. Златопольский Д.М. Основы программирования на языке Python. – М.: ДМК Пресс, 2017. – 284 с. ( Дата обращения: 13.03.2024)
3. Гэддис Т. Начинаем программировать на Python. – 4-е изд.: Пер. с англ. – СПб.: БХВ-Петербург, 2019. – 768 с. ( Дата обращения: 15.03.2024)
4. Лучано Рамальо Python. К вершинам мастерства. – М.: ДМК Пресс, 2016. – 768 с. ( Дата обращения: 17.03.2024)

**3. Приложения**

1) Приложение-1. Техническое задание.

2) Приложение-2. Руководство программиста.