Санкт-Петербургский политехнический университет Петра Великого

Кафедра компьютерных систем и программных технологий

**Отчёт по проекту**

**Дисциплина**: Алгоритмы и структуры данных

**Тема:** Русские шашки

Выполнил студент гр. 3530901/90003 Д.В. Бурьянов

(подпись)

Преподаватель М.Х. Ахин

(подпись)

“ ” 2020 г.

Санкт-Петербург

2020

Оглавление

[**1. Техническое задание** 3](#_Toc57231376)

[**2. Метод решения** 7](#_Toc57231377)

[**3. Описание классов и методов** 8](#_Toc57231378)

[**4. Работа программы** 11](#_Toc57231379)

# **1. Техническое задание**

**Описание приложения**

**Общие сведения**

Реализация игры «Русские шашки». Поле состоит из 64 клеток. На поле 12 белых и 12 черных шашек. Изначально шашки занимают по первые три ряда черных клеток, белые снизу, черные соответственно, сверху. Ходы производятся только по черным клеткам. Если шашка попала на крайнюю клетку со стороны доски, противоположной той, откуда она начинала, она становится дамкой, что позволяет ей делать ход на неограниченное число клеток в любом направлении, в отличие от обычных шашек, которые ходят только вперед на одну клетку. Отличительной особенностью русских шашек является обязательное взятие: если игрок имеет возможность побить вражескую шашку, он обязан это сделать. Игра заканчивается, когда у одного из игроков не остается ни одной шашки.

**Более подробная информация**

* Приложение будет реализовано на JavaFX
* На стартовом окне приложения есть следующие компоненты:

1. Изображение, которое является фоном окна
2. Текст «Welcome to Russian Checkers»
3. Текст «»
4. Кнопка «Игра с другом», которая сразу переводит нас в режим игры на двоих, где люди делают ход и за белых, и за черных
5. Кнопка «Игра с компьютером», которая переводит нас в окно выбора

* На окне выбора есть следующие компоненты:

1. Кнопка «Игра за белых», которая переводит нас в режим игры, где ИИ будет совершать ход черными шашками
2. Кнопка «Игра за черных», которая переводит нас в режим игры, где ИИ будет совершать ход белыми шашками
3. Изображение, которое является фоном окна

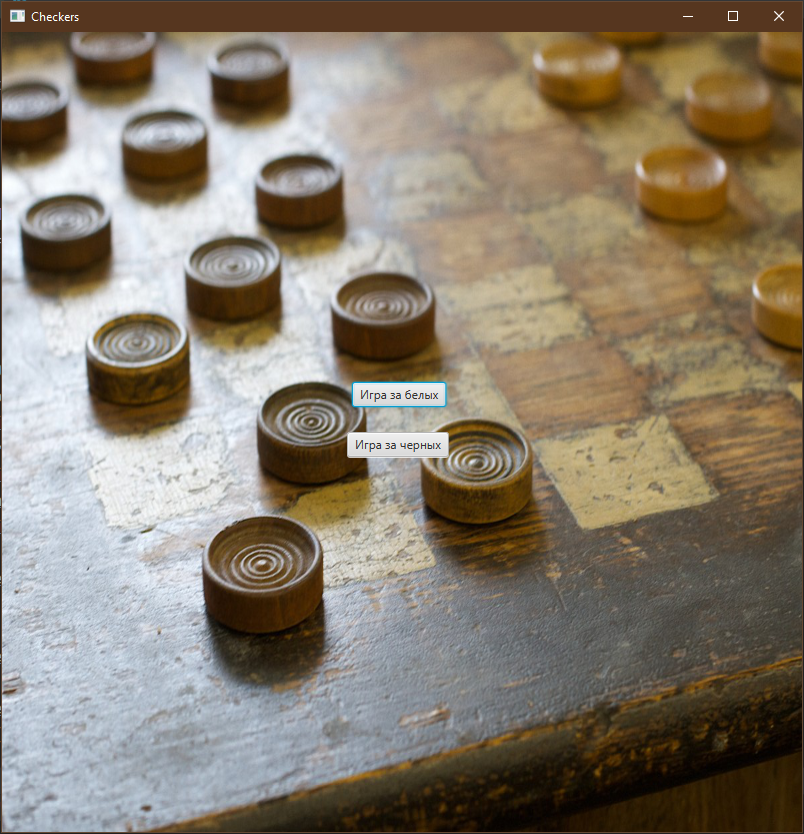
* В окне игры есть следующие компоненты:

1. 32 белых и 32 черных Rectangle, которые представляют собой клетки
2. 12 белых и 12 черных StackPane, которые представляют собой шашки

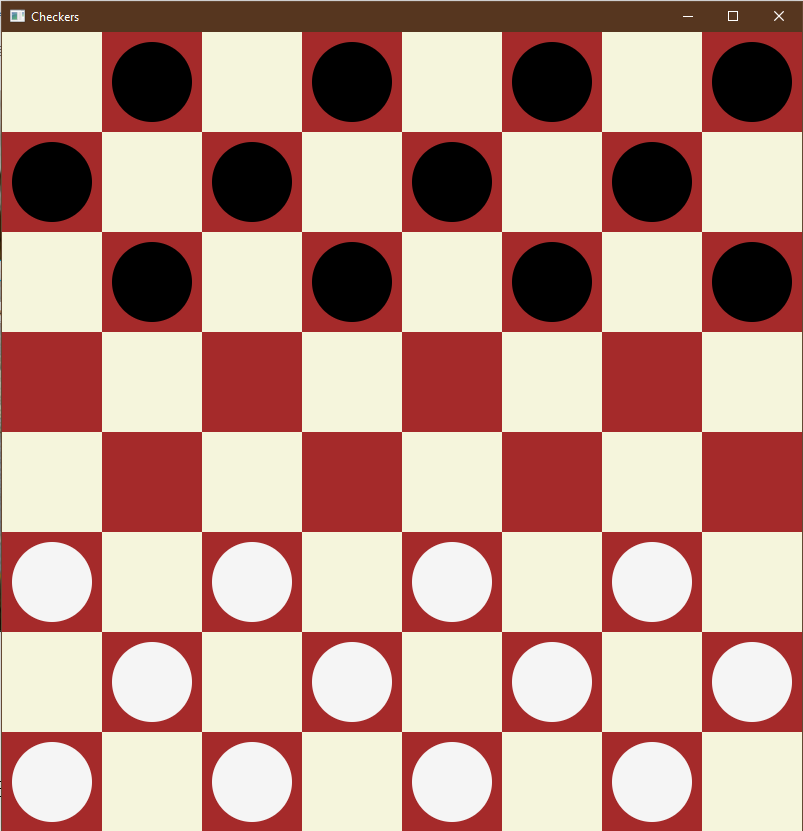
**Вид приложения**



Главный экран, встречающий при входе



Выбор цвета при игре с ИИ



Игровой процесс

**GitHub**

<https://github.com/DmitryBuryanov/RussianCheckers>

# **2. Метод решения**

В основе решения работы ИИ лежит идея алгоритма «Минимакс». Мы просчитываем наши действия на несколько ходов вперед, при этом после каждого хода оценивая позицию на поле. Так как мы делаем именно Русские шашки, добавление альфа-бета отсечений не является особо эффективным, потому что буквально через ход-два после начала партии игроки попадают в ситуацию, когда один из них обязан бить, поэтому ходы без взятия не будут просчитываться. После первого такого взятия, особенно при играх с сильным соперником, начинается большая цепочка ходов, в которых у игроков есть лишь один вариант хода, и обычно в таких играх побеждает игрок, который правильно начнет эту цепочку.

Алгоритм работает так:

Изначально заданная глубина просчета покажет, как глубоко мы зайдем при просчете ходов. Когда алгоритм запускается, он знает, за какой цвет ему надо играть, и поэтому он начинает проходить по клеткам поля и искать шашки заданного цвета. При нахождении такой он смотрит, обязан ли он сейчас бить. Если да, он проверяет, возможно ли взятие данной шашкой. Если нет, он начинает искать новую. Если да, он проверяет все возможные варианты взятия, которые он может совершить из данной клетки. В случае, если шашка – дамка, он проверяет и каждую позицию, на которую он может встать после взятия. Если же возможности взятия нет, он проверяет каждый ход, который он может сделать своими фигурами, причем после каждого хода алгоритм сохраняет состояние поля, которое было до хода, чтобы потом вернуть его, и запускает себя в рекурсии, но ход делая уже шашками другого цвета, просчет ходов которых аналогичен ходам черных. Алгоритм будет рекурсивно запускаться либо до тех пор, пока не будет достигнута максимальная глубина просчета, либо пока не будет достигнута победа одного из игроков. Как только прекращается рекурсивный вызов алгоритма, производится оценка состояния поля, и шашки становятся на свои изначальные места. Алгоритм возвращает список, состоящий из 5 элементов: начальные координаты шашки, координаты, куда надо сделать ход, и оценка, которая достигается через количество ходов, равное глубине просчета, если игроки играют на победу. Ход ИИ делается через метод поля, куда подставляются полученные ранее координаты.

# **3. Описание классов и методов**

**Класс MainApp:**

Отвечает за визуализацию состояния поля, получает от пользователя информацию о совершаемых ходах.

**Методы класса Main:**

makeScreen():

Отрисовка главного экрана приложения, в котором пользователь выбирает, как играть: с человеком или с ИИ

choise():

Если выбрана игра с ИИ, отрисовывает окно, в котором пользователь выбирает, за какой цвет шашек он будет играть

makeField():

Заполняет поле клетками

fillBoard():

На основе состояния доски заполняет поле шашками

start (Stage stage):

Запуск приложения

createCell(int i, int j):

Создает клетку по заданным координатам

createChecker(int i, int j, Checker checker, boolean isDamka):

Создает модель шашки, в которой описывается модель ее взаимодействия с пользователем. Шашка передвигается посредством удержания ЛКМ на ней и перетаскивания ее на другую клетку. В случае неправильных ходов(на белую клетку или ход не дамкой на несколько клеток вперед) шашка вернется на место. В случае возможного хода, но не в данной ситуации(ход чужой шашкой или ход без взятия, когда оно обязательно), появляются предупреждения, реализуемые через Alert, которые покажут, что вы сделали не так

**Класс CheckerModel:**

Визуально реализует шашку, наследуется от StackPane

У данного класса нет методов, у него есть только конструктор, главной задачей которого является создать Circle нужного цвета в нужной клетке. В случае, если шашка – дамка, на ней появляется изображение короны

**Класс Cell:**

Отражает состояние клетки поля

**Методы класса Cell:**

getChecker():

Возвращает шашку, содержащуюся в этой клетке

setChecker(Checker checker):

Помещает в данную клетку заданную шашку

hasChecker():

Проверяет, есть ли в данной клетке шашка

**Класс Checker:**

Отражает состояние шашки

go(int newX, int newY):

Изменяет координаты шашки

getOldX():

Возвращает координату по горизонтали

getOldY():

Возвращает координату по вертикали

**Enum Color:**

Содержит в себе 5 вариантов цветов для удобства обозначения:

**WHITE, BLACK, BEIGE, BROWN, NEITHRAL**

**Класс Gamestate:**

Содержит в себе состояние поля и изменяет его

**Методы класса Gamestate:**

getBoard():

Очищает доску и возвращает все шашки на начальные позиции

canMove(int newX, int newY, Checker checker):

Определяет, может ли заданная шашка походить на заданные координаты. Возвращает 0, если не может, 1, если может без взятия, и 2, если может со взятием

makeMove(int newX, int newY, Checker checker):

Совершает ход, убирая заданную шашку со старой позиции и переставляя ее на новую

gameover():

Проверяет, окончена ли игра. Возвращает “”, если не окончена, “Black won”, если выиграли черные, и “White won”, если выиграли белые

cellExist(int x, int y):

Проверяет, существует ли клетка с заданными координатами

canByte(Cell cell):

Проверяет, может ли шашка, находящаяся в заданной клетке побить вражескую шашку

needtobyteforWhite():

Проверяет, нужно ли белым бить в данный момент

needtobyteforBlack():

Проверяет, нужно ли черным бить в данный момент

makeIIMove(Color color):

Совершение хода ИИ

copyBoard():

Копирует и сохраняет текущее состояние доски

makeOldBoard(Cell[][] oldBoard, Color color):

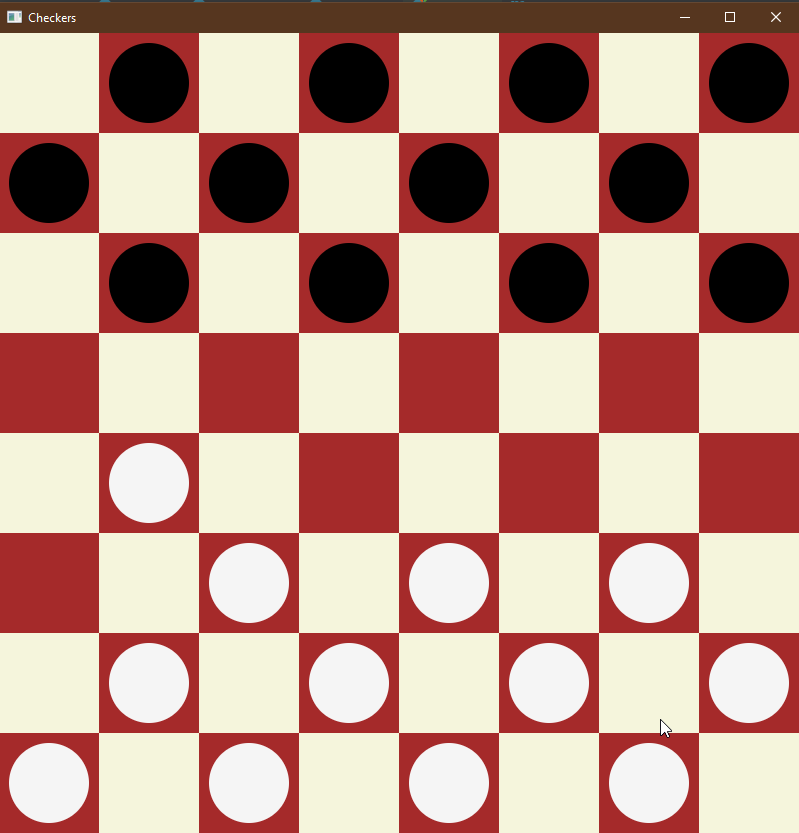
Отменяет предыдущий ход, заменяя состояние доски на ранее сохраненное

getEvaluation():

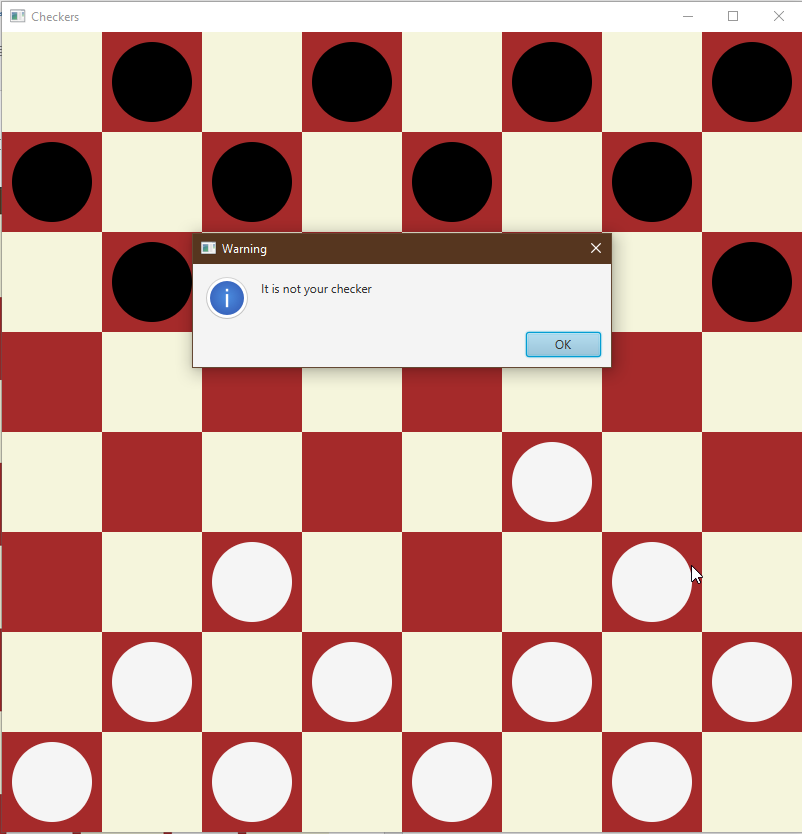
Оценочная функция. Положение примерной ничьей – 0. Чем оценка ниже – тем ближе к победе черные, чем оценка выше – тем ближе к победе белые

minimax(int depth, Color color) – алгоритм поиска наилучшего хода ИИ. Возвращает список из 5 элементов: старые координаты шашки, координаты, куда надо ходить и оценка при игре на победу

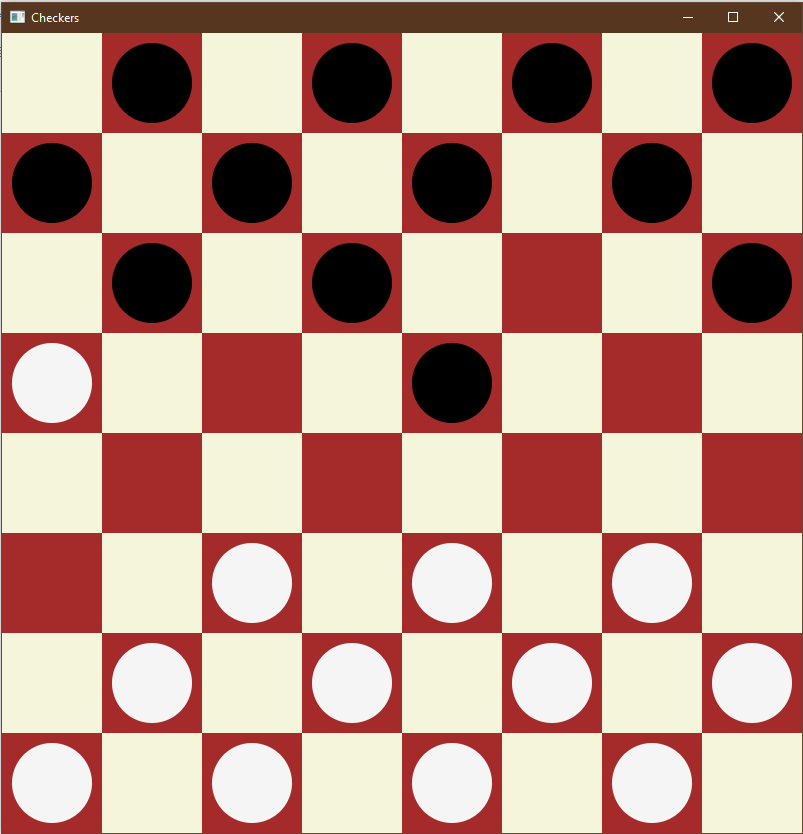
# **4. Работа программы**



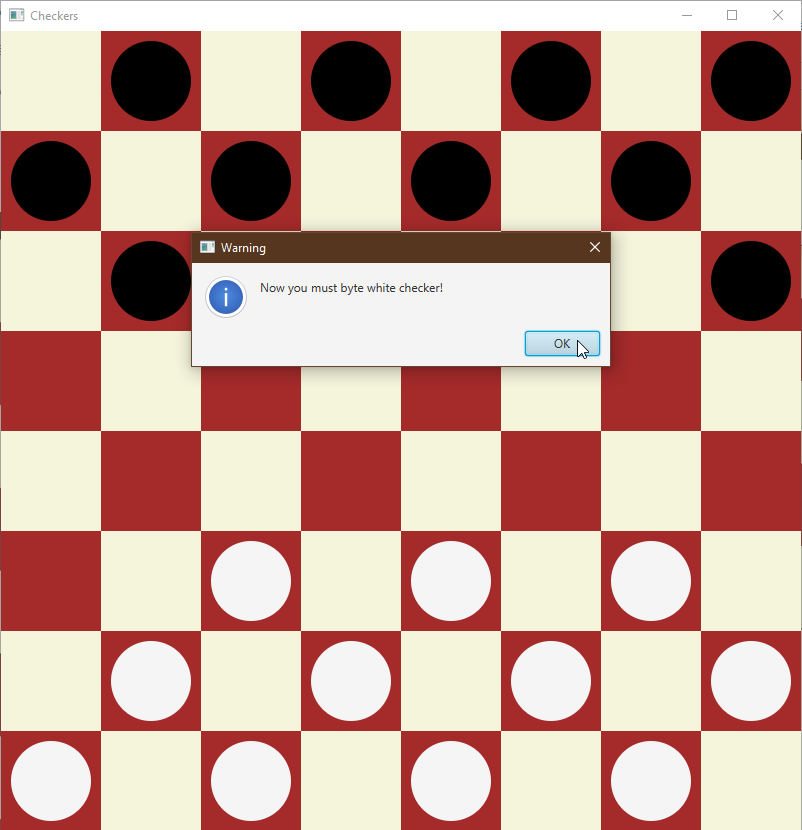
Выберем играть за черных: как видим ИИ сделал свой ход за белых



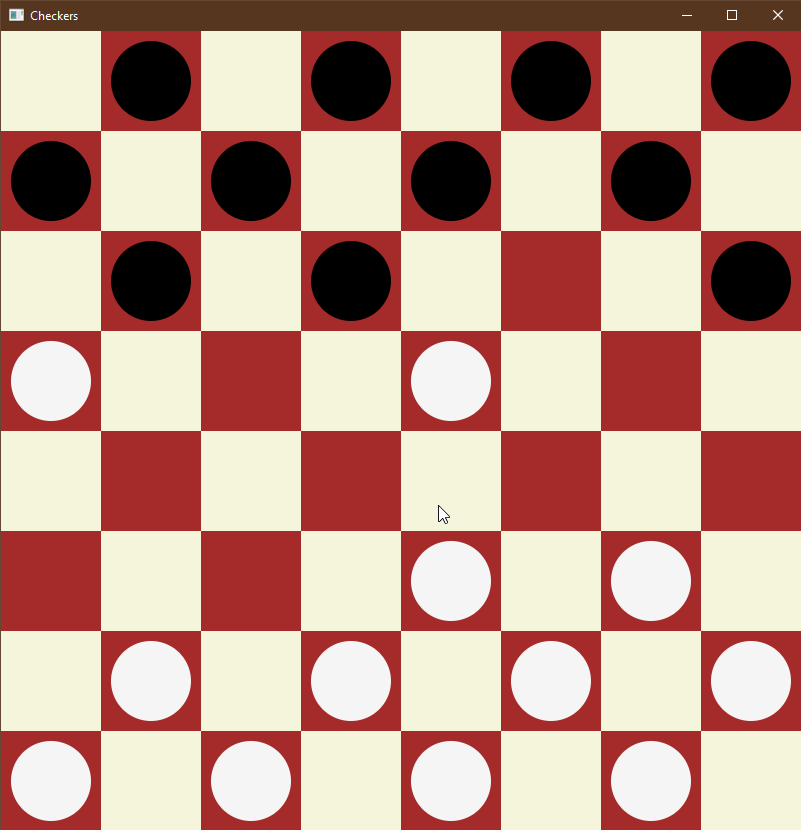
Попробуем походить за белых: вылезло предупреждение

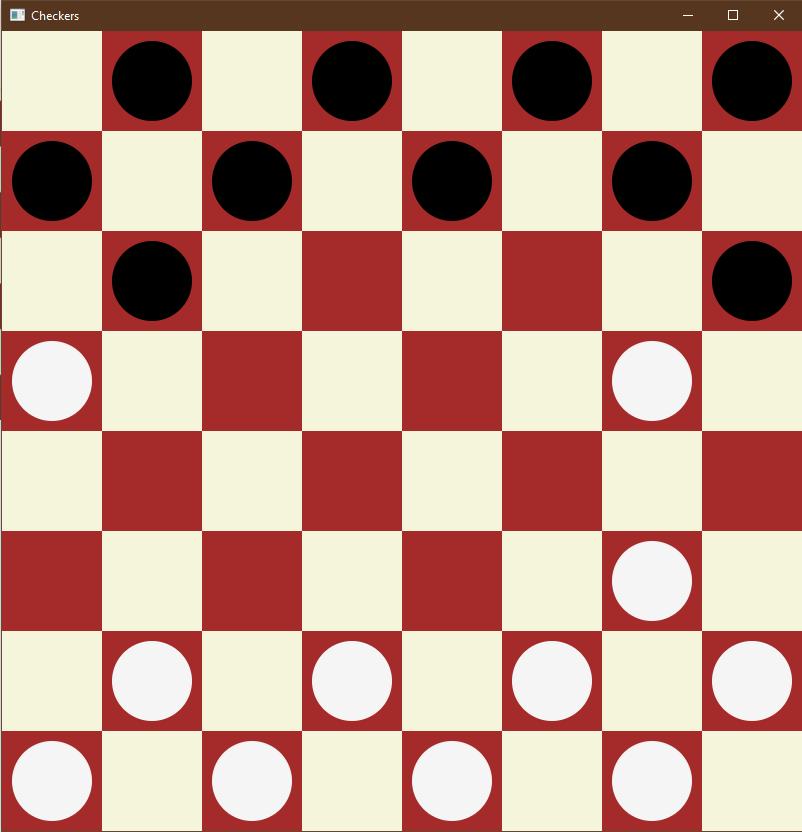


Совершим свой ход



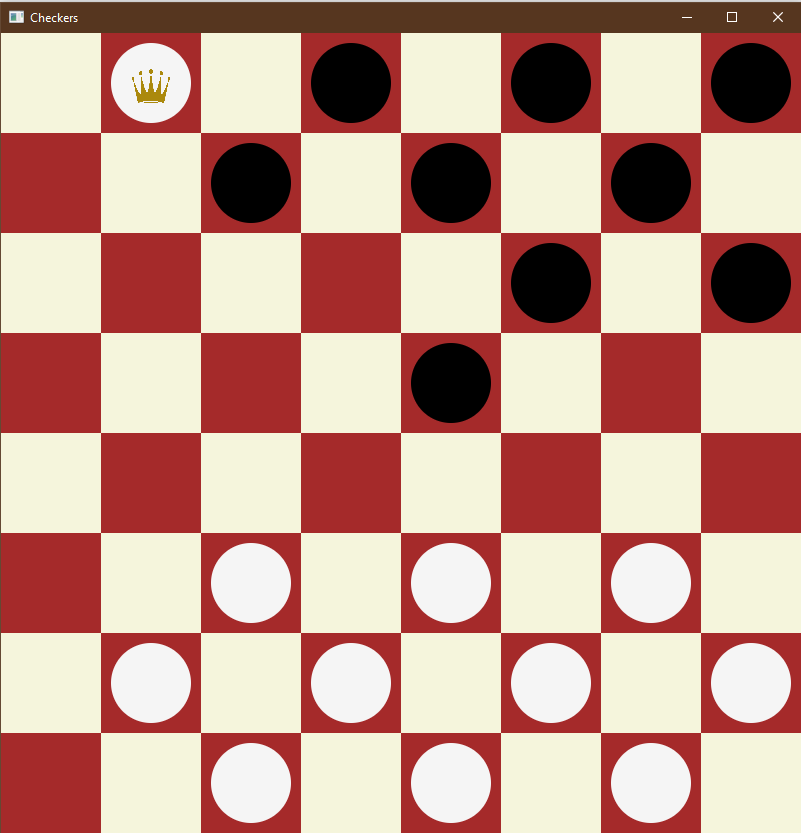
Не будем бить, когда это надо. Как видим, вылезло предупреждение.



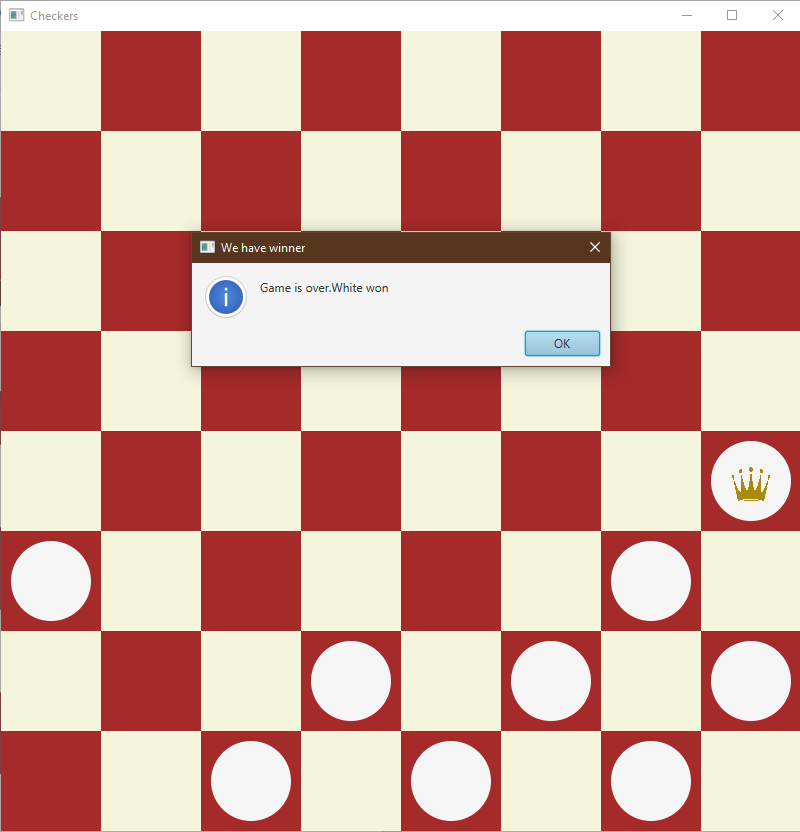


Совершим взятие. Как видно, после того как мы убили шашку соперника,

он убил нашу в ответ



Белая шашка добралась до противоположного края доски и стала дамкой.



Игра закончилась победой белых.

При игре за белых взаимодействие приложения и пользователя аналогично