Project Final Report

Artificial Intelligence

Submitted by:

Areej Iman (SP23-BSE-007)

Warda Irfan (SP23-BSE-054)

Submitted to:

Ma'am Farah Saeed

Project Title: Checkers Game with AI using Minimax Algorithm

1. Abstract/Summary:

The objective of this project is to develop a fully functional Checkers game application featuring a graphical user interface (GUI) and an AI opponent. The game is implemented in Python using Tkinter for GUI, and the AI employs the Minimax algorithm with alpha-beta pruning for decision making. The project aims to provide a smooth interactive experience where the human player competes against an intelligent computer opponent, showcasing fundamental AI concepts applied to classical board games.

2. Introduction:

Checkers, also known as Draughts, is a popular two-player strategy board game with simple rules but complex strategy. The goal is to capture all opponent pieces or block their moves. This project implements a standard 8x8 checkers board, allowing a human player to compete against a computer-controlled AI. The AI uses the Minimax algorithm optimized by alpha-beta pruning to select the best moves based on heuristic evaluation of board states.

3. Implementation:

Board Representation:

- The board is represented as a 2D list (8x8) with constants defining empty cells, player pieces, and kings.
- Each tile alternates color for visual clarity.

Game Logic:

- Initial setup places player and AI pieces in their starting rows.
- Movement rules follow standard checkers: simple diagonal moves and mandatory captures.
- Multiple jumps (multi-capture moves) are supported recursively.
- Pieces promote to kings when reaching the opposite end.

AI Implementation:

- The AI uses the Minimax algorithm with alpha-beta pruning for efficient decision-making.
- Evaluation function scores the board based on piece counts and king values, favoring positions beneficial to AI.
- The AI searches up to a depth of 4 moves ahead for reasonable performance.

GUI Features:

- Interactive board rendering with color-coded tiles and pieces.
- Highlighting selected pieces and their valid moves.
- Indication of mandatory captures by highlighting.

- Smooth player interaction via mouse clicks.
- Game over detection with a message box announcing the winner.

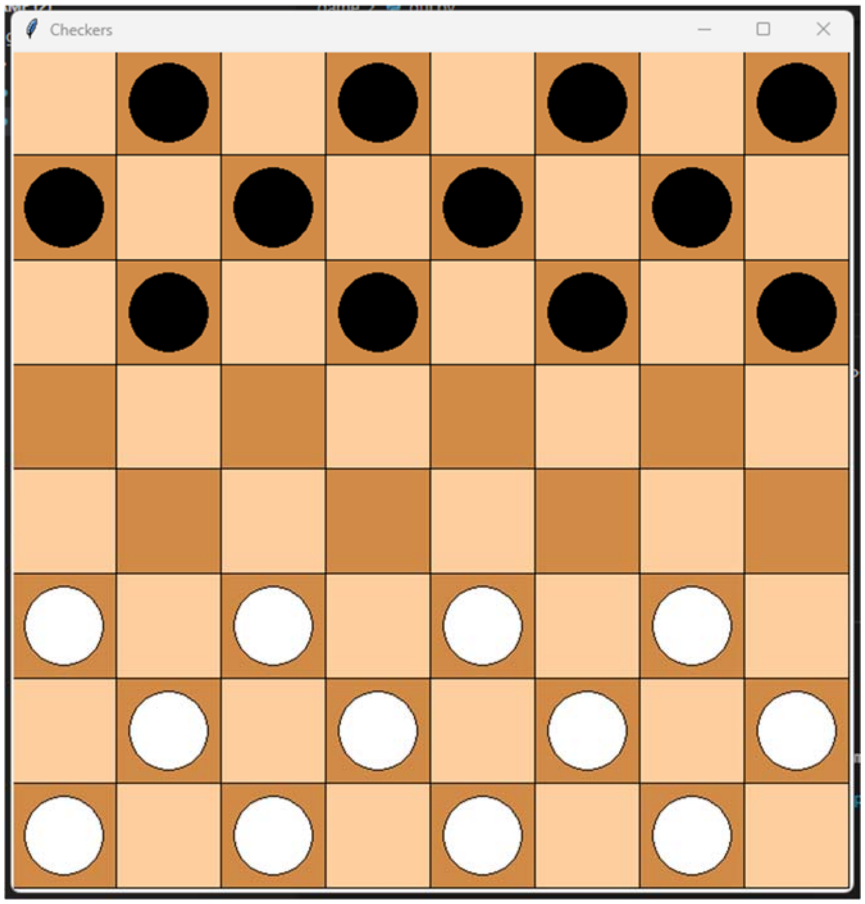
CODE: Zipped with this document

5. Results:

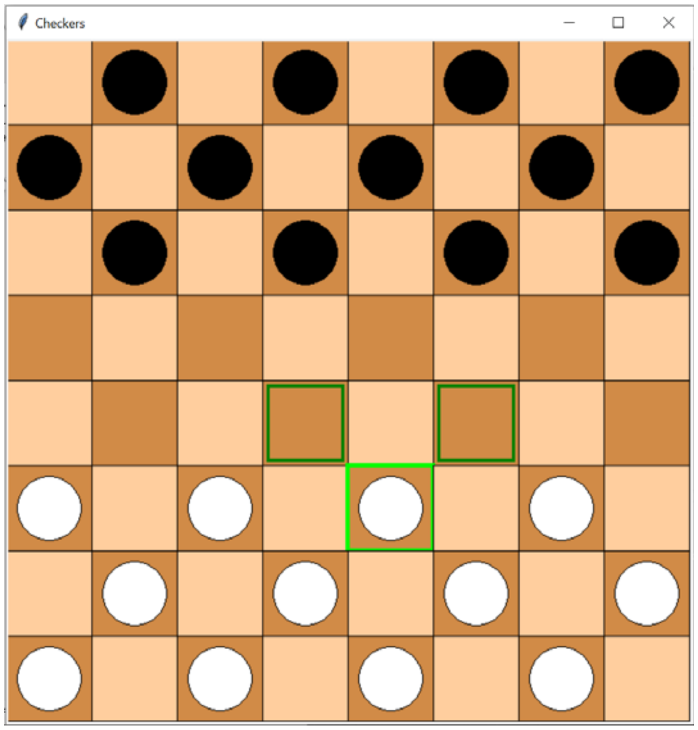
Functional Gameplay:

- The game board displays correctly with alternating tile colors and pieces properly positioned at the start.
- Players can select and move pieces according to the rules of Checkers, including normal moves and forced captures.
- Pieces get promoted to kings when reaching the opponent's baseline, and king moves (both forward and backward) are supported.

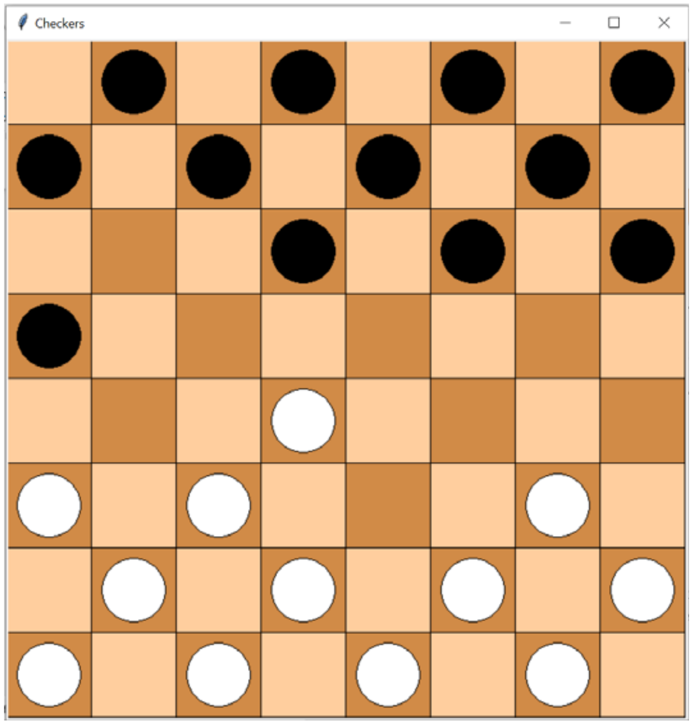
When you first start the game:



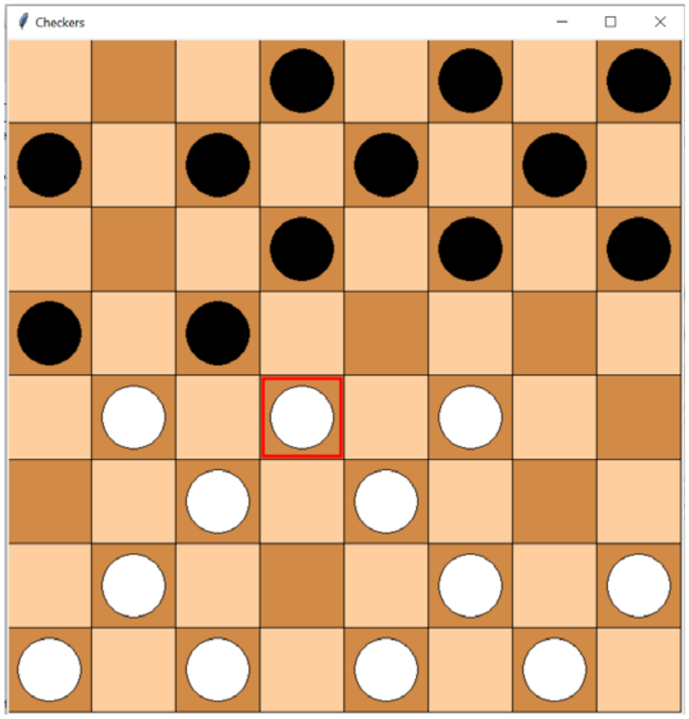
Possible Movements for the User:



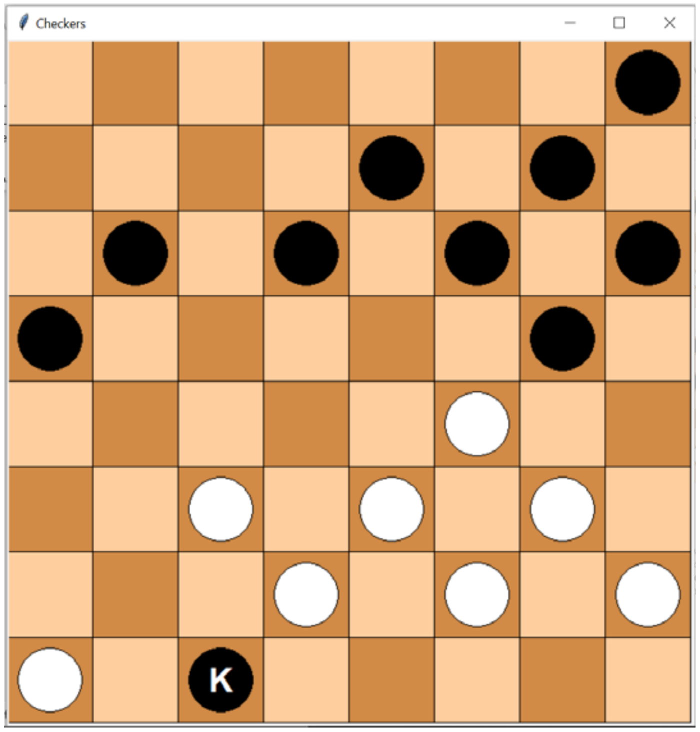
AI moved based on minmax Algo:



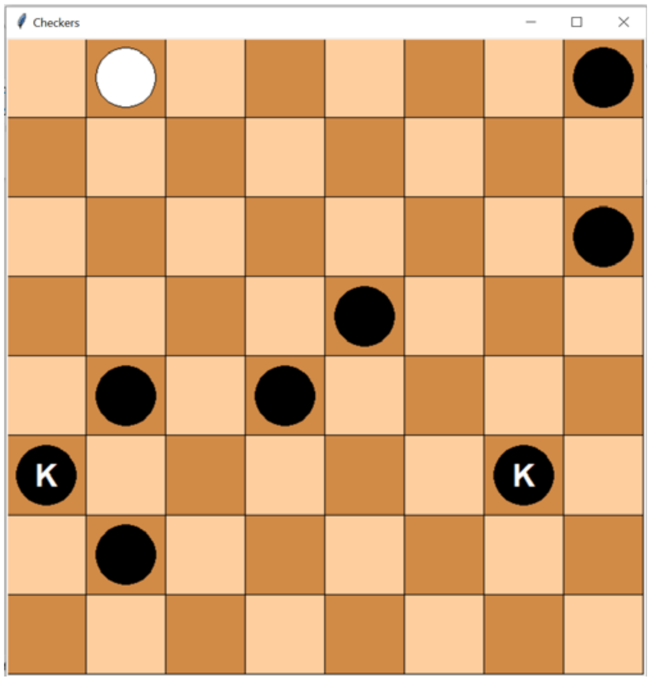
User can now capture AI's piece:



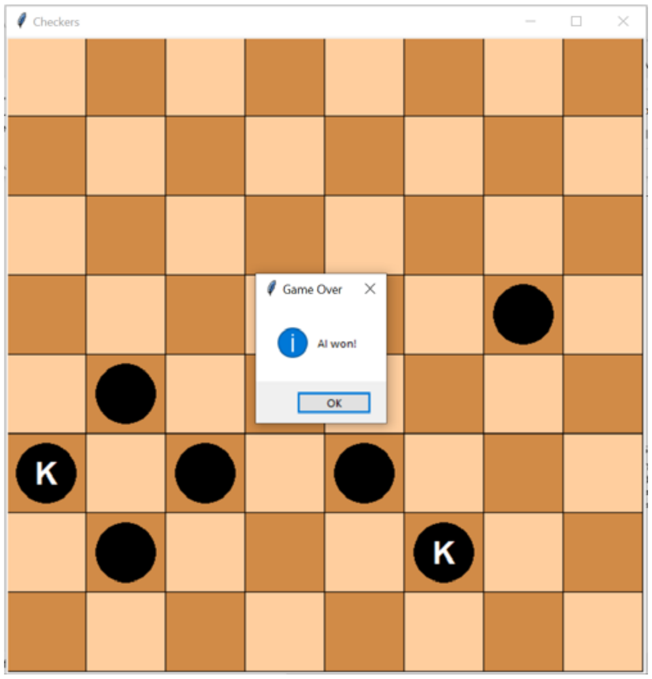
AI piece becomes the king indicating it can move diagonally both forward and backward::



User becomes the king:



AI wins the game:



AI won!

6. Conclusion:

This project successfully demonstrates the implementation of a classic Checkers game with a playable AI opponent. It combines GUI programming with fundamental AI search algorithms, providing an educational and enjoyable experience. The project meets its objectives and serves as a practical example of applying algorithms in game development.