Project Title: Checkers AI Game

Submitted By: [Muhammed Ahmed 22k-4725]
Group members: [Uzair Ali 22k-4695] [Huzkail waseem 22k-4678]

Course: AI
Instructor: [Miss Mehak mazhar]
Submission Date: [8-march-2025]

---

1. Project Overview

Project Topic:

This project focuses on developing an AI-enhanced Checkers game with three gameplay modes:

1. **Two-Player Mode** – Standard Checkers game between two human players.
2. **Easy AI Mode** – AI selects random moves using Python's random library.
3. **Hard AI Mode** – AI implements the Minimax algorithm at a depth of 3 to make strategic decisions.

Objective:

The primary objective is to develop an AI capable of playing Checkers efficiently using the Minimax algorithm. The game aims to provide different levels of difficulty, from simple random moves to a strategic AI that prioritizes king pieces. The AI's decision-making process is optimized using an evaluation function that weighs the number of regular and king pieces on the board.

---

2. Game Description

Original Game Background:

Checkers is a classic board game played on an 8x8 grid. Players move diagonally, capturing opponent pieces by jumping over them.

If a piece reaches the last row, it becomes a king and can move both forward and backward.

Innovations Introduced:

- **AI Gameplay Modes**: The inclusion of Minimax-based AI enhances the challenge level.
- **Visual Enhancements**: Pieces that are promoted to kings are marked with a yellow ring.
- **Evaluation Heuristic for AI**: The AI prioritizes moves leading to king pieces by weighting them more heavily in the evaluation function.

---

3. AI Approach and Methodology

AI Techniques to be Used:

- **Minimax Algorithm**: Used for the hard AI mode to make optimal moves based on a game tree search.
- **Heuristic Evaluation Function**: Evaluates board states by counting the difference in piece numbers and prioritizing kings.
- **Random Move Generator**: Used for the easy AI mode.

Heuristic Design:

- The evaluation function computes the difference between red and blue pieces.
- King pieces are weighted 1.5 times more than regular pieces to encourage strategic king-making moves.
- Reference: Analysis from http://studentnet.cs.manchester.ac.uk/resources/library/3rd-year-projects/2010/yiannis.charalambous-2.pdf.

Complexity Analysis:

- The Minimax algorithm with a depth of 3 results in a complexity of approximately **O(b^d)**, where **b** is the branching factor and **d** is the depth (3 in this case).

---

4. Game Rules and Mechanics

Modified Rules:

- Standard Checkers rules apply, with king pieces gaining bidirectional movement.
- AI prioritizes king-making moves for better strategic gameplay.

Winning Conditions:

- A player wins by capturing all opponent pieces or blocking all possible moves.

Turn Sequence:

- Players take alternating turns. The AI's turn is calculated using Minimax in hard mode and random moves in easy mode.

---

5. Implementation Plan

Programming Language:

- Python

Libraries and Tools:

- **Pygame** – For graphical interface and rendering
- **Random** – For easy AI move generation
- **Custom Minimax Implementation** – For hard AI mode

Milestones and Timeline:

- **Week 1-2:** Game design and rule finalization
- **Week 3-4:** AI strategy development (Minimax and heuristics)
- **Week 5-6:** Coding and testing the game mechanics
- **Week 7:** AI integration and testing
- **Week 8:** Final testing and report preparation

---

6. References

- Checkers Game Rules: [https://www.pinterest.com/pin/345792077637457088/]
- Minimax Algorithm Analysis: [http://studentnet.cs.manchester.ac.uk/resources/library/3rd-year-projects/2010/yiannis.charalambous-2.pdf]