Project Report: Exploding Checkers AI Game

Group Members

- **Muneeb Maqsood Khan (22k4481)**
- **Hashir Ahmed Khan (22k4419)**

---

1. Introduction

This project presents *Exploding Checkers*, a dynamic twist on the traditional checkers game developed in Python using the Pygame library. It introduces explosive pieces capable of initiating chain reactions to eliminate multiple opponent pieces, adding strategic complexity. The game includes a visually engaging interface, smooth animations, and a heuristic-driven AI opponent.

---

2. Objectives

- Develop a playable checkers game with an innovative explosive mechanic.
- Design and implement an AI opponent with strategic decision-making abilities.
- Enhance the player experience with intuitive UI and polished animations.
- Ensure correct handling of chain explosions and dynamic game state updates.

---

3. Technologies Used

- **Python 3**: Core programming language used for all logic.
- **Pygame**: For graphics rendering, animations, and user interaction.
- **Math & Random Libraries**: Used for easing animations and randomly assigning explosive pieces.

---

4. Game Design and Features

4.1 Board and Pieces

- 8×8 grid with alternating light and dark squares.
- Each player starts with 12 pieces positioned on the first three rows of dark squares.
- Red pieces represent the human player; Blue pieces represent the AI.
- Pieces become **Kings** upon reaching the last row, allowing backward moves.
- Some pieces are randomly assigned as **explosive**, marked by a pulsing green border and an asterisk.

4.2 Explosive Mechanics

- Explosive pieces remove all pieces in a 3×3 grid centered on them.
- Chain explosions occur if neighboring pieces are also explosive.
- Explosions are limited and must be used strategically.
- Implemented via a **queue-based system** to manage cascading explosions efficiently.

4.3 Movement and Captures

- Diagonal movement for regular pieces; kings can move backward.
- Captures are **mandatory** and support multi-jump sequences.

- Highlighting of valid moves helps guide players.
- AI prioritizes moves involving captures and explosions.

4.4 AI Implementation

- The AI evaluates moves using a custom **heuristic function** that considers:
    - Number of regular pieces and kings.
    - Number of explosive pieces.
    - Potential outcome of explosions.
- AI evaluates multiple move sequences and chooses the most advantageous.

4.5 User Interface and Animation

- Game window includes a **header bar** showing the current turn.
- Visual polish includes:
    - Gradient-filled pieces with shadows.
    - Crown icons for kings.
    - Asterisk and green glow for explosive pieces.
    - Smooth animations using **ease-in-out quadratic functions**.
- Frame-by-frame rendering ensures seamless visuals and interaction.

---

5. Code Structure Overview

- `Piece` **Class**: Manages individual piece data (position, type, status) and drawing logic.
- `Board` **Class**: Manages board state, piece placement, valid move detection, explosions, and win conditions.

- **Main Game Loop**: Handles event input, AI turns, game state updates, and rendering.

---

6. Challenges and Solutions

Explosion Chain Reactions

- **Challenge**: Ensuring that chain reactions were correctly detected and processed.
- **Solution**: Implemented a **queue-based BFS algorithm** to simulate and resolve multiple chained explosions.

AI Move Evaluation

- **Challenge**: Making the AI behave strategically and not randomly.
- **Solution**: Designed a custom **heuristic evaluation** function that weighs several factors, including piece safety and potential explosive impact.

Smooth Animations

- **Challenge**: Creating a visually appealing user experience.
- **Solution**: Used **interpolation and easing functions** for animations, along with dynamic redrawing of pieces.

User Interaction

- **Challenge**: Providing clear feedback for valid moves and turn transitions.
- **Solution**: Integrated UI indicators for move highlights, current player turn, and visual feedback on selected pieces.

---

7. Conclusion

The *Exploding Checkers* game offers a refreshing and challenging extension of classic checkers. With visually refined graphics, explosive gameplay mechanics, and a competent AI, the project successfully integrates innovation and user experience. It showcases advanced game logic, thoughtful design, and clean implementation using Pygame.