AI PROJECT

*Project Report*



# "AI CHESS GAME"

**Group Members:**

* Abrar Saleem 20k-0129 (Sec-D)

**Course Instructor:**

Miss Yusra Kaleem  
Sir Abdullah Yaqoob

1. **Executive Summary**

**Project Overview**: This project aimed to enhance the traditional chess game by introducing a new hybrid piece called the Knight-Queen and developing an AI capable of playing this modified version using the Minimax algorithm with Alpha-Beta pruning. The primary objective was to build a strategic AI opponent that could handle the increased complexity brought by the new piece and make optimal decisions using well-designed heuristics.

1. **Introduction**

* **Background**: Chess is a classical two-player strategy board game played on an 8x8 grid. Traditionally, each player controls 16 pieces with unique movements. This project selected chess due to its structured rules and deep strategic possibilities. The innovation introduced was the Knight-Queen—a powerful piece that combines the movement abilities of both a knight and a queen—thus enriching tactical options and increasing game complexity.
* **Objectives of the Project:**
* Develop an AI model to play the modified chess game with the new Knight-Queen piece.
* Incorporate Minimax and Alpha-Beta pruning to ensure intelligent and efficient move selection.
* Evaluate AI performance against human players and optimize its decision-making capability.

1. **Game Description**

* **Original Game Rules**: In traditional chess, each player begins with 16 pieces: 1 king, 1 queen, 2 rooks, 2 bishops, 2 knights, and 8 pawns. The aim is to checkmate the opponent’s king by placing it under inescapable threat.
* **Innovations and Modifications:**
* Introduction of a new piece: Knight-Queen (QK), which can move like both a knight and a queen.
* The Knight-Queen increases the coverage and flexibility of attacks and defenses.
* The game logic and AI had to be adapted to incorporate this new piece's capabilities in move generation and evaluation.

1. **AI Approach and Methodology**

* **AI Techniques Used:**
* Minimax Algorithm: To simulate and evaluate possible future game states.
* Alpha-Beta Pruning: To optimize the Minimax tree by pruning non-promising branches, thereby reducing computation time.
* Heuristics-based Evaluation: Used to score board positions based on piece values and positional strength.
* **Algorithm and Heuristic Design:**
* Piece values: Pawn=1, Knight=3, Bishop=3, Rook=5, Queen=9, King=100, Knight-Queen=12.
* Evaluation considers material value, piece mobility, center control, and king safety.
* Tactical patterns like forks, pins, and checkmates are also assessed in the heuristic.
* **AI Performance Evaluation:**
* The AI was evaluated by simulating multiple games against itself and human players.
* Metrics included win rate, decision time per move, and heuristic accuracy in predicting strong positions.

1. **Game Mechanics and Rules**

* **Modified Game Rules:**
* The Knight-Queen can move in an L-shape (like a knight) and also along ranks, files, and diagonals (like a queen).
* All other standard chess rules are maintained, including castling, en passant, and promotion.
* **Turn-based Mechanics:**
* Players take alternate turns.
* Each player moves one piece per turn, adhering to legal move rules, including the new Knight-Queen movement.
* **Winning Conditions:**
* The game ends with a checkmate or resignation, as per traditional chess rules.

1. **Implementation and Development**

* **Development Process:**
* The game was implemented in Python using the Pygame library for GUI development.
* The core AI logic was written using Minimax and Alpha-Beta pruning principles.
* Heuristics were designed to evaluate the modified board positions effectively.
* **Programming Languages and Tools:**
* Programming Language: Python
* Libraries: Pygame, NumPy
* Tools: GitHub (version control), VS Code (IDE)
* **Challenges Encountered:**
* Designing legal moves for the Knight-Queen without disrupting existing logic.
* Managing increased branching factor in AI due to the hybrid piece.
* Ensuring the GUI updates seamlessly with real-time AI decision-making.

1. **Team Contributions**

* Team Member and Responsibilities:
* Abrar: Developed the AI algorithm (Minimax, Alpha-Beta Pruning)
* Abrar: Defined new chess rules and implemented Knight-Queen movement logic
* Abrar: Built the graphical interface using Pygame and managed user interaction
* Abrar: Conducted AI vs. human testing and fine-tuned heuristic functions

1. **Results and Discussion**

* **AI Performance:**
* The AI achieved a win rate of approximately 75% against novice human players.
* Average decision-making time per move: 1.5 to 2.2 seconds (with Alpha-Beta pruning).
* The heuristic evaluation effectively recognized threats and advantages, especially when the Knight-Queen was active.

1. **References**

* Russell, S., & Norvig, P. (2010). Artificial Intelligence: A Modern Approach
* Pygame Documentation: https://www.pygame.org/docs/
* Wikipedia: Chess and Minimax Algorithm
* GitHub Open Source Projects (for chess GUI inspiration)