**Project Title: Quantum Snakes & Ladders - AI Enhanced Board Game**

**Submitted By:** **Hamna Ahsan K213558**

**Neha Khan K214756**

**Shayan Qureshi K213603**   
**Course:** AI  
**Instructor:** Miss Mehak Mazhar

**Submission Date:** 8th march 2025

**1. Project Overview**

**Project Topic:**

Quantum Snakes & Ladders is a modified version of the classic Snakes & Ladders board game. The game incorporates **strategic AI, power-ups, and quantum elements** to increase complexity and engagement. The game will feature:

* **AI-controlled players using Minimax with Alpha-Beta Pruning** for decision-making.
* **Power-ups like Reroll, Superposition, and Entanglement** to influence gameplay.
* **Monte Carlo Tree Search (MCTS) for handling uncertainty in dice rolls.**

**Objective:**

* Develop an **AI strategy** that optimizes dice roll choices using **Minimax with Alpha-Beta Pruning**.
* Implement **a new set of rules and power-ups** to enhance game complexity.
* Use **MCTS** to simulate long-term game outcomes for decision-making.
* Develop a **graphical interface (GUI) using Pygame** for better gameplay experience.

**2. Game Description**

**Original Game Background:**

Snakes & Ladders is a **classic board game** where players roll a die to move forward on a numbered board. Landing on a ladder moves a player up, while landing on a snake moves them down. The goal is to reach the final square first.

**Innovations Introduced:**

* **AI-powered players:** The game includes an **AI opponent** that makes intelligent decisions instead of relying solely on random dice rolls.
* **Power-ups:** Players can collect power-ups that allow them to:
  + **Reroll**: Get another chance to roll the die.
  + **Superposition**: Choose between two possible dice rolls.
  + **Entanglement**: Swap positions with another player.
* **Strategic AI-based dice selection:** Instead of random dice rolls, the AI chooses the best roll using Minimax.
* **Monte Carlo simulations:** The AI predicts the best long-term moves using MCTS.

**3. AI Approach and Methodology**

**AI Techniques to be Used:**

* **Minimax Algorithm:** Determines the best move by assuming all players play optimally.
* **Alpha-Beta Pruning:** Optimizes Minimax to reduce unnecessary computations.
* **Monte Carlo Tree Search (MCTS):** Simulates multiple dice rolls and predicts the best move in uncertain scenarios.
* **Reinforcement Learning (Optional):** AI learns optimal strategies through self-play.

**Heuristic Design:**

* **Avoiding snakes** and **aiming for ladders** for positional advantage.
* **Evaluating power-ups** and deciding the best time to use them.
* **Predicting opponent moves** to minimize their advantage.

**Complexity Analysis:**

* **Minimax Time Complexity:** O(b^d) (where b = branching factor, d = depth of decision tree)
* **Alpha-Beta Pruning:** Reduces complexity to O(b^(d/2)).
* **Monte Carlo Tree Search:** Balances computation with randomized simulations for practical efficiency.

**4. Game Rules and Mechanics**

**Modified Rules:**

* Players can **collect power-ups** instead of rolling the die.
* AI-controlled players use **intelligent dice selection** instead of random rolls.
* Certain board positions trigger **special events** like teleportation or power-up loss.

**Winning Conditions:**

* A player wins by **reaching the final square first**.
* If multiple players reach the final square in the same turn, the player with **fewer total moves wins**.

**Turn Sequence:**

* Each player rolls a **six-sided die (1-6)** (AI chooses the best roll instead of random rolling).
* The player **moves accordingly** and activates any board effects (snakes, ladders, power-ups).
* If a power-up is available, the player can **use or save it for later**.
* The turn passes to the next player.

**5. Implementation Plan**

**Programming Language:**

Python

**Libraries and Tools:**

* **Pygame:** For board visualization and user interaction.
* **NumPy:** For handling data and AI calculations.
* **Scikit-learn:** For implementing reinforcement learning (if applied).
* **TensorFlow (Optional):** For training an advanced AI model.

**Milestones and Timeline:**

|  |  |
| --- | --- |
| **Week** | **Task** |
| 1-2 | Game design, board layout, and rule finalization |
| 3-4 | AI strategy development (Minimax, Alpha-Beta Pruning) |
| 5-6 | Coding core game mechanics and player movement |
| 7 | AI integration and testing with MCTS |
| 8 | Final testing, debugging, and report preparation |

**6. References**

* "Artificial Intelligence: A Modern Approach" by Stuart Russell & Peter Norvig.
* Research papers on **Minimax, MCTS, and AI for board games**.
* Pygame documentation: https://www.pygame.org/docs/
* Monte Carlo Tree Search in Board Games: A Survey.