**Project Title:  
UNO Spinner Showdown: An AI-Powered Twist on Classic UNO**

**Submitted By:** Zain Abbas

**Course:** AI

**Instructor:**Miss Alishba Subhani

**Submission Date:** 11/5/25

1. Project Overview

**Project Topic:** This project implements a strategic AI-driven version of the classic UNO card game, enhanced with spinner mechanics for added randomness and complexity. The innovation introduces a "Spinner Wheel" mechanic that can dramatically alter gameplay by skipping turns, forcing card swaps, reversing play, or making players draw extra cards.

**Objective:** To develop a competitive AI for UNO that strategically plays the game using heuristic-based decision-making, while incorporating spinner dynamics to increase unpredictability. The project aims to balance strategic planning and adaptive responses to random spinner outcomes.

2. Game Description

**Original Game Background:** UNO is a card game typically played with 2–10 players. Each player is dealt 7 cards, and the goal is to discard all cards first by matching the color or value of the top card on the discard pile. Special cards like Skip, Reverse, Draw Two, and Wild add strategic depth.

**Innovations Introduced:**

* **Spinner Mechanic**: Every few turns, a spinner is activated. It randomly lands on effects like “Draw 4,” “Skip Turn,” “Reverse Order,” or “Swap Hands,” adding unpredictable events.
* **AI Opponent**: A bot that plays strategically based on hand evaluation, color prevalence, and special card prioritization.
* **Dynamic Wild Card Logic**: AI selects wild card color based on most frequent color in hand, enhancing decision-making.

**Impact on Gameplay:**

* Increases unpredictability and dynamic strategy requirements.
* Encourages adaptive AI behavior under uncertain outcomes.
* Adds replay value with more varied game sessions.

3. AI Approach and Methodology

**AI Techniques to be Used:**

* **Heuristic-Based Strategy**: Since UNO is a game with incomplete information, instead of Minimax, the AI uses a heuristic decision-making strategy that evaluates playable options based on:  
  + Card utility
  + Color distribution in hand
  + Priority of special cards
* **Reinforcement Learning (Optional)**: Could be introduced in future versions to train AI via self-play under spinner rules.

**Heuristic Design:**

* Prioritize playable cards that reduce hand size fastest.
* Prefer cards that limit opponent responses (e.g., Skip, Draw Two).
* Wild cards use most frequent hand color for flexibility.
* Spinner outcomes handled with contingency planning.

**Complexity Analysis:**

* **Time Complexity**: O(n) where n is number of playable cards, per turn decision.
* **Challenges**:  
  + Handling hidden opponent cards.
  + Modeling spinner randomness in AI planning.
  + Ensuring real-time responsiveness in multi-turn simulations.

4. Game Rules and Mechanics

**Modified Rules:**

* Spinner activates every 3 turns (configurable).
* Spinner outcomes include:  
  + Draw 4 cards
  + Skip current player’s turn
  + Reverse turn order
  + Swap hands with a random player
* AI opponent plays based on current game state and hand evaluation.

**Winning Conditions:**

* First player to discard all cards wins.
* If spinner forces players to draw past a max card count, game ends in AI/human loss.

**Turn Sequence:**

* Players alternate turns.
* Spinner is triggered every few turns.
* AI makes real-time decisions using heuristics.
* Turn continues unless Skip/Draw/Reverse modifies flow.

5. Implementation Plan

**Programming Language:** Python

**Libraries and Tools:**

* random (for card shuffling and spinner)
* colorama (for colored terminal output)
* (Optional) pygame – for future graphical interface
* NumPy – for data structuring (if extended to multiple AIs)
* AI Libraries (if extended): TensorFlow, Keras for RL experiments

**Milestones and Timeline:**

* **Week 1-2:** Finalize game rules and design spinner mechanics
* **Week 3-4:** Develop AI strategy and heuristics
* **Week 5-6:** Implement core game logic and spinner integration
* **Week 7:** Integrate AI into game loop and test
* **Week 8:** Final testing, optimization, and documentation

6. References

* Official UNO Rules (Mattel Games)
* AI for Games – Ian Millington
* Reinforcement Learning: An Introduction – Sutton & Barto
* https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory/
* https://realpython.com/tutorials/python-projects/
* <https://www.pygame.org/>