**Project Title:** AI Solver for 2048  
 **Submitted By:** Yousuf Raza (22K5079)  
 Abdullah Arif (22K5071)  
 Saad Zafar (22K5039)  
 **Course:** AI  
 **Instructor:** Ms Alishba Subhani  
 **Submission Date:** 05/11/2025

## **1. Executive Summary**

### **Project Overview:**

This project aims to develop an AI agent capable of autonomously playing the popular game 2048. The AI solver evaluates the game state using heuristic-based decision-making techniques to determine the best possible move at each turn. Enhancements such as the snake pattern heuristic, adjacent tile clustering, and empty tile prioritization were introduced to enable intelligent play. The project may also extend to more advanced algorithms like Expect Imax or reinforcement learning to improve the agent’s strategic depth and decision-making under uncertainty.

## **2. Introduction**

### **Background:**

2048 is a single-player puzzle game where players slide numbered tiles on a 4×4 grid to combine them and create a tile with the number 2048. Due to its simple mechanics but complex strategy, it presents an ideal problem for AI experimentation. This project selected 2048 to explore search-based AI decision-making and evaluate how far heuristics can push performance before transitioning to probabilistic or learning-based models.

### **Objectives of the Project:**

* Develop an autonomous AI agent to play 2048.
* Design and implement heuristic evaluation functions for intelligent move selection.
* Evaluate the AI’s performance in terms of tile score, number of moves, and win rate.
* (Optional) Extend to Expect Imax or reinforcement learning-based agents in future work.

## **3. Game Description**

### **Original Game Rules:**

* The game is played on a 4×4 grid.
* Players swipe tiles in one of four directions (up, down, left, right).
* When two tiles of the same number collide, they merge into one with double the value.
* After each move, a new tile (usually a 2 or 4) appears in a random empty spot.
* The game ends when no valid moves remain.

### **Innovations and Modifications:**

* **AI-Driven Gameplay**: The human player is replaced by an autonomous AI agent.
* **Heuristic-Based Evaluation**: The AI chooses moves based on strategic scoring:  
  + Snake Pattern Heuristic: Rewards structured tile placement.
  + Adjacent Tile Heuristic: Prioritizes merges of similar tiles.
  + Empty Tiles Heuristic: Prefers moves that increase flexibility.
* **Future Addition**: Expect Imax algorithm to incorporate probabilistic modeling of tile spawns.
* **Custom Grid Class**: Built for efficient state management and computation.

## **4. AI Approach and Methodology**

### **AI Techniques Used:**

* **Heuristic-Based Search**: The agent evaluates and selects moves by computing a score for each possible board state.
* **Expectimax Algorithm (Planned)**: This approach considers both the AI’s deterministic moves and the randomness introduced by new tiles.
* **Reinforcement Learning (Future Work)**: Exploring the possibility of training the AI through self-play using reward optimization.

### **Algorithm and Heuristic Design:**

* **Snake Pattern Scoring**: Encourages a top-down or side-to-side winding layout to keep higher tiles grouped.
* **Adjacent Tile Clustering**: Gives preference to states where similar tiles are adjacent, making merges easier.
* **Empty Tile Count**: Favors game states with more free spaces to avoid premature game over.

### **AI Performance Evaluation:**

* Performance will be measured using:  
  + Highest tile achieved.
  + Average number of moves before the game ends.
  + Win rate across multiple simulations.

## **5. Game Mechanics and Rules**

### **Modified Game Rules:**

* The game is no longer user-controlled.
* The AI evaluates possible directions using heuristics and executes the best move.
* New tiles spawn randomly after each move, preserving the original game’s uncertainty.

### **Turn-based Mechanics:**

1. AI evaluates the board state.
2. It ranks all four possible moves based on heuristic scores.
3. The best move is executed.
4. A new tile (2 or 4) is randomly placed.
5. Repeat until no moves are possible.

### **Winning Conditions:**

* The AI’s success is judged based on its ability to reach tile 2048 or beyond.
* Performance metrics include highest tile achieved and longevity of play.

## **6. Implementation and Development**

### **Development Process:**

The game and AI were implemented in Python. NumPy was used for efficient matrix and grid operations, while the custom Grid class handled board logic and game state transitions. The AI evaluated the current board using heuristic functions and selected the optimal move. The design was modular to allow easy integration of more advanced techniques like Expectimax in future phases.

### **Programming Languages and Tools:**

* **Programming Language:** Python
* **Libraries:**
  + NumPy (grid computation and manipulation)
  + Matplotlib *(optional, for performance graphs)*
  + Pygame *(optional, for visualizing game state)*
* **Tools:**
  + GitHub (version control)

### **Challenges Encountered:**

* Balancing between computation speed and move accuracy.
* Designing heuristics that generalize well across all board states.
* Handling randomness from tile spawns, which complicates deterministic planning.
* Ensuring smooth integration of AI logic with the grid logic.

## **7. Team Contributions**

* **Yousuf Raza (22K5079):** Designed heuristic functions, worked on AI move evaluation logic.
* **Abdullah Arif (22K5071):** Developed the custom grid class, handled state transitions and tile merges.
* **Saad Zafar (22K5039):** Integrated AI with the game engine, conducted simulations and collected performance data.

## **8. Results and Discussion**

### **AI Performance:**

* The AI consistently achieved tile values in the range of 1024–2048.
* On average, games lasted between 400–600 moves.
* Heuristic-based decisions significantly reduced unnecessary moves and improved score consistency.
* The snake pattern heuristic showed clear benefits in tile clustering and space optimization.
* (If Expectimax is implemented) The AI is expected to make more robust decisions by considering future uncertainty.

## **9. References**

* Stuart Russell & Peter Norvig, *Artificial Intelligence: A Modern Approach (3rd Edition)*
* "Multistage Temporal Difference Learning for 2048-Like Game" –<https://ieeexplore.ieee.org/document/7518633>
* "Optimistic Temporal Difference Learning for 2048" –<https://arxiv.org/pdf/2111.11090>
* Various online resources and GitHub repositories on 2048 AI implementations and Expectimax algorithms