## Introduction:

- Our project implements AI agents for Connect 4
- For those unfamiliar, Connect 4 is a two-player game on a 7 by 6 grid.
- Players take turns dropping colored discs into columns, and gravity pulls them to the lowest open spot.
- The goal is to connect four of your pieces in a row, either horizontally, vertically, or diagonally.
- Despite simple rules, Connect 4 has over 4 trillion possible positions, so we can't just check every possibility.
- Instead, we built four AI strategies of increasing intelligence: Random, Greedy, Minimax, and Minimax with Alpha-Beta pruning.

## Heuristic Evaluation Function:

- Before explaining our AI strategies, we need to understand how we evaluate a board position.
- Our heuristic function scans every possible four-cell window on the board and assigns scores.
- For example, three of our pieces with one empty space scores +5 because we're one move from winning.
- Three opponent pieces with one empty space scores -4, signaling we need to block.
- We also reward controlling the center column with +3 per piece, since center positions connect to more winning lines.
- This scoring lets the AI judge how 'good' any position is without playing to the end."

## Greedy Algorithm:

- Our first real strategy is the Greedy AI
- It looks one move ahead by simulating each possible column, calculating the heuristic score of the resulting position, and picking the move with the highest score.
- This is fast, under 5 milliseconds, and beats random play easily. It blocks obvious threats and takes immediate wins.
- However, Greedy has a major weakness: it can't see beyond one move.

- If I set up a trap where I create two winning threats at once, Greedy blocks one but loses to the other. It can't anticipate my responses to its moves.

## Minimax:

- To fix this, we use the Minimax algorithm
- Minimax looks multiple moves ahead by building a game tree.
- It assumes both players play optimally: we try to maximize our score, and the opponent tries to minimize it.
- At each level, we alternate between picking the maximum score (our turn) and the minimum score (opponent's turn)
- By searching 4 or 5 moves deep, minimax can see traps coming and avoid them.
- The problem is computational cost
- With 7 possible moves per turn, depth 5 means checking 7 to the power of 5, which is about 17,000 positions.

## Alpha Beta Pruning:

- Alpha-Beta pruning solves this.
- It tracks two values: alpha, the best score we can guarantee, and beta, the best score the opponent can guarantee.
- When we find a branch where beta is less than or equal to alpha, we prune it, meaning we skip it entirely because it can't affect our final decision.
- With good move ordering, checking center columns first, Alpha-Beta cuts 80 to 95 percent of the search space.
- This lets us search deeper in less time.
- Our Alpha-Beta AI at depth 5 actually responds faster than regular Minimax at depth 4, while playing significantly stronger."