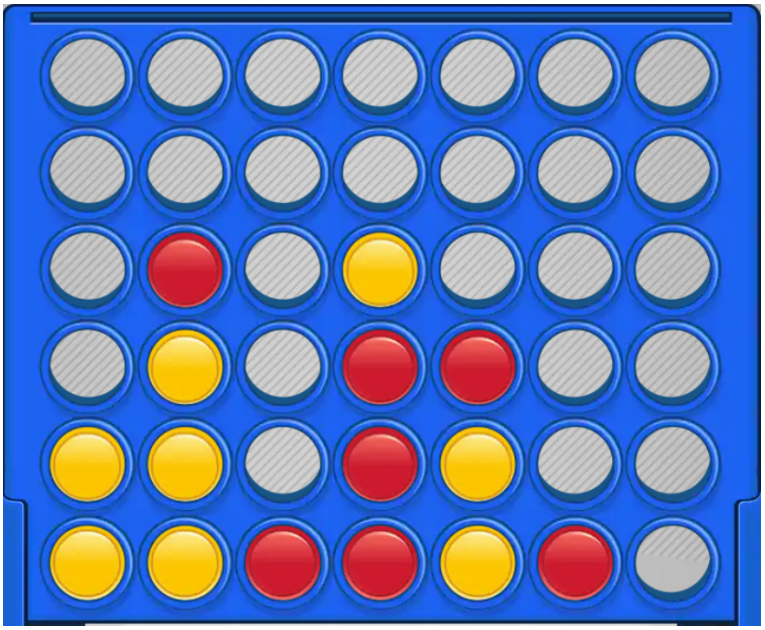
Assignment 2
Using Minimax Algorithm to Make Connect 4 AI Agent

1 Game Description

Connect 4 is a two-player game in which the players first choose a color and then take turns dropping their colored discs from the top into a grid. The pieces fall straight down, occupying the next available space within the column. The objective of the game is to connect-four of one's own discs of the same color next to each other vertically, horizontally, or diagonally. The two players keep playing **until the board is full**. The winner is the player having **greater number of connected-fours**.



2 Requirements

2.1 Game GUI

You are required to provide a full game with GUI with only one mode human vs. computer where we choose whether to use alpha-beta pruning or not in AI agent then play against the agent till the board is full. The game dimensions are as follows (width ≥ 7, length ≥ 6). You can fix any acceptable dimensions.

2.2 Algorithms

You are required to support the following 3 algorithms as options for the AI agent:

- Minimax without alpha-beta pruning
- Minimax with alpha-beta pruning
- Expected Minimax : probability that disc falls in chosen column is 0.6 and 0.4 that it will into the column on left or right

The agent will run the required algorithm in each turn and it is required to show the minimax tree in each turn by printing it to the console in a readable form that we can trace. Showing the tree using GUI will be grant a bonus.

2.3 Heuristic Pruning

You can notice that the full game tree is very huge $O(10^{35})$ so it is impossible to traverse it till the terminal states instead you are required to design a suitable heuristic function that evaluates the state of a game and returns a number indicating whether the computer is near to win or lose and truncate the game tree after K levels evaluating the states using the heuristic function. The game should take K as a parameter before beginning the game.
Better heuristic should return bigger number when computer is nearer to win and smaller number when human is nearer to win. You can notice that there is many factors that can be taken into account when designing the heuristic such as the current scores and the number of candidate points for each player. Try to think carefully in the design and take many factors in account. The best 3 heuristics will be grant a bonus. **show the values of board after applying the heuristic (you will lose marks if not) tree printing is a must**

3 Notes

3.1 Deliverables

- Your well commented code.
- A report showing your work,
  including:
  - sample runs and their corresponding minimax trees
  - comparison between the 2 algorithms in terms of time taken and nodes expanded at different K values

Also the report should contain the data structures used (if any) and algorithms, Assumptions and details you find them necessary to be clarified, Any extra work.

3.2 Further Notes

- You may use Java, Python or C++ for your implementation.
- Copied assignments will be severely penalized.
- You can work in groups of 2.

Good Luck