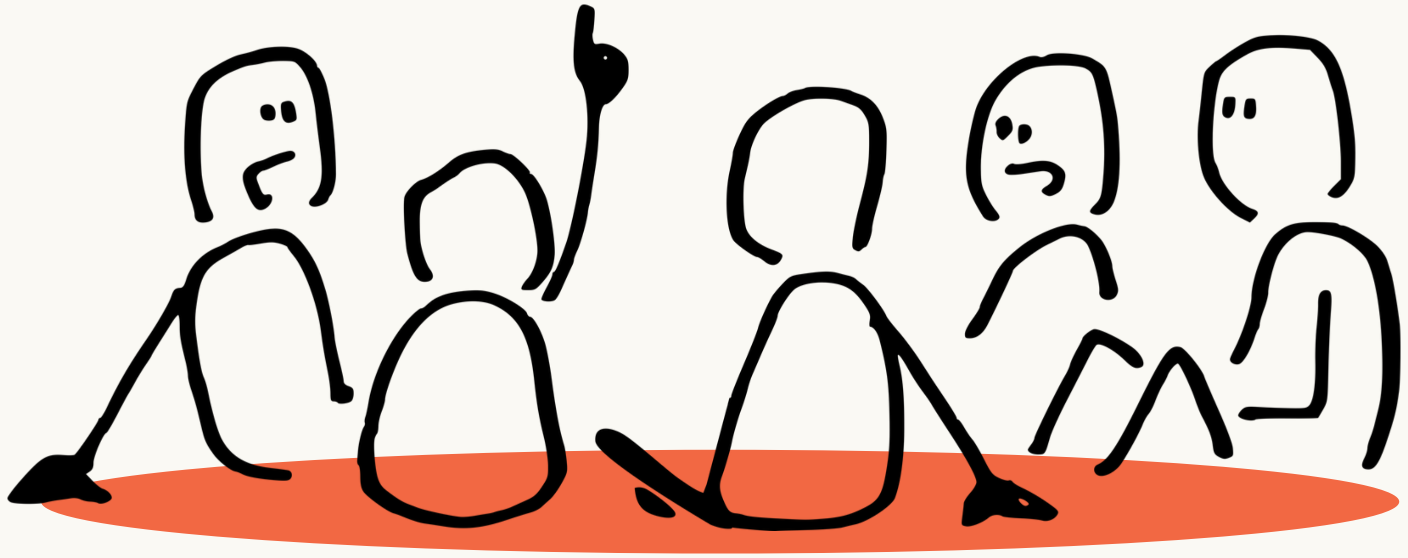
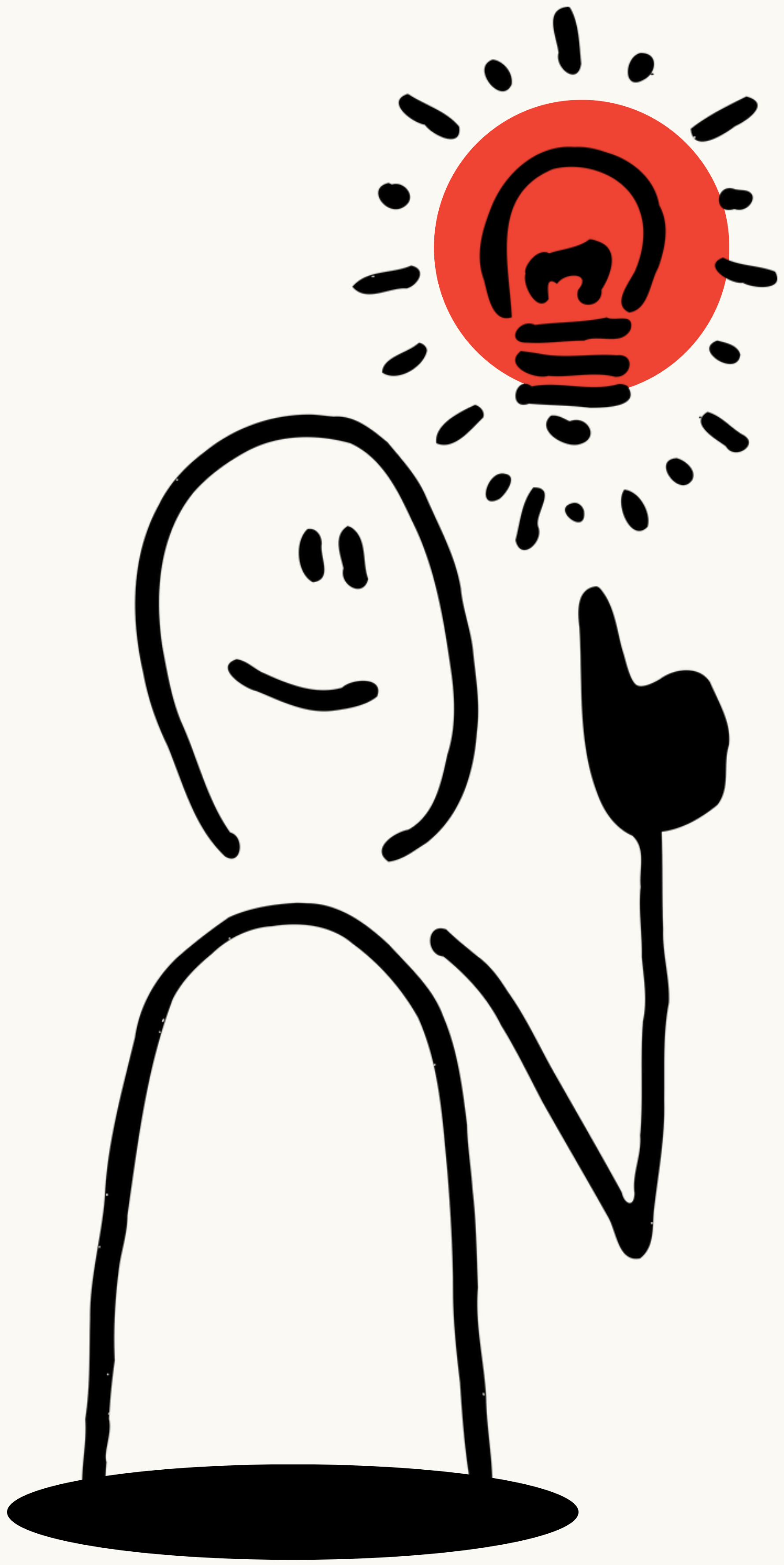
Algorithm and Data Structure

PROGRESS OF IMPLEMENTATION OF CLASSICAL AI ALGORITHMS IN A CONSOLE-*BASED CONNECT 4 GAME*



GROUP 4

Agha Aryo Utomo / 5026241003
Maria Stephanie F. K / 5026241052
Dyandra R-Noor B / 5026241051



SUMMARY

Focus: Developing an intelligence console-based Connect 4 game in Java
Objective: Create an AI opponent capable of evaluating the board, predicting future states, and making optimal moves in real time
Algorithms used:

- Linear Search
- BFS
- Merge Sort
- Minimax

At this stage, the game loop, board rendering, player input, and partial AI logic are functioning correctly and form the foundation for further algorithmic enhancements.

What We Have Done

Main.java: Create an object called game to run the Connect 4

Game.java:

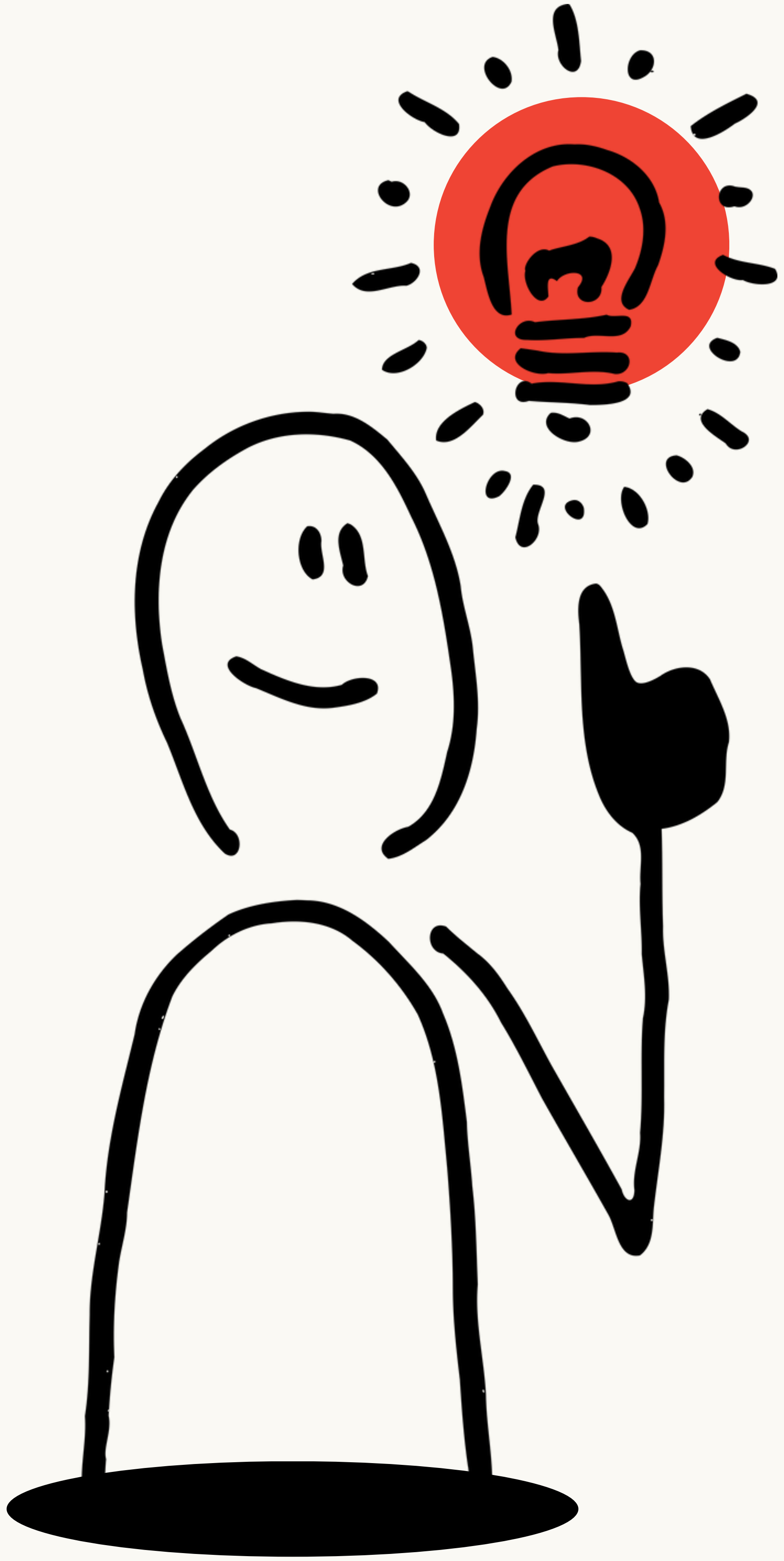
- Turn system (Human → AI → Human → AI)
- Input validation
- Calling the AI to choose moves
- Dropping discs
- Checking win/draw states
- Printing board
- Closing scanner

Board.java:

- the 2D grid
- win detection
- legal move calculation
- undo moves (critical for minimax)
- linear search gravity
- cloning board

AIPlayer.java:

1. Immediate win & block logic
2. Quick heuristic scoring
3. Merge sort for move ordering
4. Minimax & alpha-beta pruning



What We Will Do

1. **Revising the class structure:**

Current Class Structure:

1. Main.java
2. Game.java
3. Board.java
4. AIPlayer.java
    - AIPlayer
    - MergeSort
    - Minimax
    - MoveEvaluator
    - MoveScore

Next Class Structure:

1. Main.java
2. Game.java
3. Board.java
4. AIPlayer.java
5. MergeSort
6. Minimax
7. MoveEvaluator
8. MoveScore

2. Modify the checkWin() method to be BFS based to detect winning 4 in a row paths