1. Game Class Refinement

In the draft, the `Game` class had attributes like `gameID` and `gameType`, along with methods like `isTie()` and `playAgain()`. These were removed in the final UML. Instead, the class now focuses on the actual game mechanics, holding essential objects:

- A `Grid` for the board
- Two `Player` objects (`player1` and `player2`)
- A `Turn` object to track whose turn it is
- A `Result` object to determine the winner
- A `Scanner` for user input

Additionally, `startGame()` was replaced with `setupPlayers()` and `play()`, which now handle **player selection and game execution** in a more structured way.

2. Turn Class Simplification

The draft version of the `Turn` class managed turns using a `turn` counter and a `static final` value of `9` (representing the max number of moves). It also included methods like `incrementTurn()` and `whoseTurn()`.

The final UML simplifies this by replacing the counter with a **boolean `playerOneTurn`**, which directly tracks whose turn it is. Instead of incrementing a number, the game now simply switches between `true` and `false`. The `playOneTurn()` method was introduced to manage a single turn, making the class cleaner and more efficient.

3. Improved Player Structure

In the draft, `Player` had a method for choosing a move, but it wasn't well-defined. The final UML improves this by introducing `doTurn(Grid grid)`, which both `Human` and `Computer` classes override. The `Computer` class now includes AI-related methods like `findBestMove()` and `minimax()` to handle decision-making.

4. Grid Class Enhancement

The `Grid` class remains similar, but it now includes `final POSITIONS: String`, likely to map moves to positions like "A1" and "B2" instead of plain row/column numbers. This makes the game **more user-friendly and intuitive**.