# Java Lab Project – Progress Report, Team 3

**Implementation:** The application’s main function will be an infinite while-loop that queries the user what he/she wants to do. The options will be singleplayer, multiplayer, ranking and quit. A switch statement will use the user input to determine which functions to use. After game/ranking viewing is complete, the loop will start a new iteration and will present the choice to the user again. We will have classes for Multiplayer, Singleplayer, Ranking, Board and Move.

The Move class will inherit from the Board class and will use the information to consider whether a move is legal and whether the move will result in a win.

The Multiplayer class will ask the users for their names and the players take turns inputing the moves they want to make, in an infinite while-loop. The Board and Move classes will be used to print and update the board and check if the moves are legal and result in a victory.

The Singleplayer class will provide the user with a computer-controlled opponent. We’re not entirely sure how to implement this yet, but we’re considering scanning the board and appraising the empty slots with numbers from 1-6, indicating the potential of putting a stone there. We will take care of rules that we cant put stones in 3,3 or we can put it but same stones 6 or more in a row is not a winning case. We will first look at the above rules and implement the AI

The Ranking class will make use of an external .txt-file to store records outside the current game execution. A ranking board will show who beat who, in how many moves.

**Development:** We established a GitHub repository (<https://github.com/SimenSverdrup/Gomoku>), to handle version control and manage collaboration. According to our original development plan, we have implemented 2 classes out of the planned 6. These are Main.class, Ranking.class, …

We still have not started to implement the Singleplayer-class, which due to the requirement of a playing AI probably will be the most difficult one. So far, our plan is to implement it using a max-min-tree (as the nature of the Gomoku game isn’t too complicated) or scanning the board and appraising the empty slots with numbers from 1-6, indicating the potential of putting a stone there. We may have to split the Singleplayer class up into several classes, to simplify the implementation.