

# CPSC 2150 Project 3 Report

Jacob Colson, Warren Wasden, Steven Cabezas

## Requirements Analysis

### Functional Requirements:

1. As a player I need to be able to drop my token on a particular column so that I can progress the game.
2. As a player I want to be able to select a number of Rows and Columns that allow up to a 100x100 board, so I can customize my game.
3. As a player, I want the game to allow for a minimum of 3 Rows or Columns so the game is playable.
4. As a player, I want the game to allow for a fast GameBoard implementation, so the game performance is very quick.
5. As a player, I want the game to allow for a memory efficient GameBoard implementation, so the game does not use a large amount of memory.
6. As a player, I want to be able to choose my own character to represent my token, so that I can personalize the game and know when it is my turn.
7. As a player I want to drop my token in a specific place so that I can block the other player from getting the required number of tokens in a row to win.
8. As a player, I want to choose how many tokens it takes to win, so I can make the game shorter or longer.
9. As a player, I need the game to allow for a minimum of 3 tokens to win, so that the game is playable.
10. As a player I want to be able to drop the required number of tokens in a diagonal line in order to win the game.
11. As a player I want to be able to drop the required number of tokens in a horizontal line in order to win the game.
12. As a player I want to be able to drop the required number of tokens in a vertical line so that I can win the game.
13. As a player I need to be able to select another column if a previously chosen column does not exist so that I can continue my turn.

14. As a player I need to be able to select another column if the column I chose first is full so that I can continue my turn.
15. As a player I want to be able to input a column number to place a token in so that I can use my turn.
16. As a player I want to be able to choose whether or not I would like to play again so that I can continue playing.
17. As a player I want to be able to know whose turn it is so that we don't lose track of whose turn it is.
18. As a player, I want the game to allow for a minimum of 2 players so I have someone to play against.
19. As a player I want the game to allow for a maximum of 10 players so we can play against each other.
20. As a player I want to be able to interact and be informed by a command line interface generated by the game.
21. As a player I want the game to alternate between player X and player O so each player gets a fair turn.
22. As a player I want the game to display the board after each turn so that I can determine where to place my token.
23. As a player I want the game board to update each time I place a token so I don't lose track of where my tokens are.
24. As a player I want the game to display the winner at the end so that we know who won.
25. As a player I want the columns to be numbered so I can easily tell which column is which.

### **Non-Functional Requirements**

1. The game must be written in Java.
2. The game will allow for a maximum of 10 players and alternate between the players.
3. The game must allow for a minimum of 2 players.
4. Players will be able to play on a customizable board of a maximum of 100x100.
5. The minimum number of rows or columns must be 3.
6. The game must determine if a specific space is available or already occupied.
7. Game must check if the player has won after each token is placed.
8. The game must check for a tie if neither player has won the game, then ask if they want to play again.

9. If a specific space is unavailable, the user must be asked if they would like to select another position.
10. The fast GameBoard implementation must use a 2-D character array.
11. The memory efficient GameBoard implementation must use a HashMap.
12. Players must be informed of which player won the game, then ask if they want to play again.
13. The game must run with a command line interface for user interaction.
14. Players need to be informed of when a selected row is full.
15. The game must inform the player if a selected column does not exist.
16. The game must allow each player to select their desired token at the beginning of the game.

## System Design

### GameScreen:

GameScreen
- playerBoard: IGameBoard - winningChar: char
+ maxPlayers: int + minPlayers: int + maxRowsCol: int + minRowsCol: int + printBoard(): void + printWinner(): void + askPlayerForColumn(): int + main(String[ ] args): void

### BoardPosition:

BoardPosition
- Row: int - Column: int
+ BoardPosition(int aRow, int aColumn) + getRow(): int + getColumn(): int + equals(Object obj): boolean + toString(): String

## IGameBoard, GameBoard, GameBoardMem, and AbsGameBoard:

