CPSC 2150 Project 3 Report

Jason Russo

Joseph Becker

Evan Schwartz

**Requirements Analysis**

**Functional Requirements:**

1. As a player, I need to be able to choose a column to drop a chip so I can take a turn.
2. As a player, I need to know which columns are full, so I do not overload one.
3. As a player, I need to know whose turn it is, so I do not act out of turn.
4. As a player, I need to know the size of the board, so I do not select a column out of the bounds.
5. As a player, I need to know when the game is over, so I know when to stop making selections.
6. As a player, I need to know who won the game, so I can track wins over other players.
7. As a player, I need to have the option to play again so multiple games can be played.
8. As a player, I need to know if the game resulted in a tie so I can quit making moves and track my record.
9. As a player, I need to know what player token I am
10. As a player, I need to be informed if my choice of move is invalid
11. As a player, I need to see the game board after each turn so that I can stay informed on the progress of the game.
12. As a player, I should be prompted and able to start a new game after I complete my game
13. As a player, I need to be presented with a fresh board if I choose to play again, so I can start a game from scratch.
14. As a player, I need to know which number corresponds with each row
15. As a player, I need to know which number corresponds with each column
16. As a player, I need to be informed of the controls for the game
17. As a player, I need to know if my last placed token completed the required same tokens in a row horizontally to win the game.
18. As a player, I need to know if my last placed token completed the required same tokens in a row vertically to win the game.
19. As a player, I need to know if my last placed token completed the required same tokens in a row diagonally to win the game.
20. As a player, I need to be able to choose from a fast or memory efficient gameboard so that I can choose one that better suits my game
21. As a player, I can select a number of players so that I can play with more than 2 players
22. As a player, I can select my player token so that I know where my moved are made
23. As a player, I can select the number of rows, columns, and tokens in a row to win so that I can create different games

**Non-Functional Requirements**

1. The game must run on Unix
2. The game must run on the command line
3. The program must be written in Java
4. The program must be able to create a board for fast play
5. The program must be able to create a board for memory efficient play
6. The board size must be adjustable, with a min size 3x3 and max size 100x100
7. The board size must be able to be a square or a rectangle
8. The number of tokens to win must be between 3 and 25
9. The first player to choose their token must go first
10. The players must take their turns in the order their tokens were selected
11. (0,0) must be the bottom left position of the board

**System Design**

**GameBoard:**

**Class diagram**

|  |
| --- |
| **GameBoard** |
| * board: char[][] |
| + GameBoard(): void  + dropToken(char p, int c): void  + whatsAtPos(BoardPosition pos): char  + ToString(): string |

|  |
| --- |
| GameBoardMem |
| - Board: Map<Character, List<BoardPosition>> |
| + GameBoard(): void  + dropToken(char p, int c): void  + whatsAtPos(BoardPosition pos): char  + ToString(): string |

**GameScreen:**

**Class diagram**

|  |
| --- |
| **GameScreen** |
| - MIN\_PLAYERS: int  - MAX\_PLAYERS: int  - MIN\_ROWS: int  - MAX\_ROWS: int  - MIN\_COLS: int  - MAX\_COLS: int  - MIN\_NUM\_TO\_WIN: int  - MAX\_NUM\_TO\_WIN: int |
| + main() : void |

**BoardPosition:**

**Class diagram**

|  |
| --- |
| **BoardPosition** |
| - Row: int  - Column: int |
| + BoardPosition(aRow: int, aColumn: int)  + getRow(): int  + getColumn(): int  + equals(obj: Object): bool  + toString(): string |

|  |
| --- |
| **<<interface>>**  **IGameBoard** |
|  |
| + getNumRows(): int  + getNumColumns(): int  + getNumToWin(): int  + checkIfFree(int c): boolean  + dropToken(char p, int c): void  + checkForWin(int c): boolean  + checkTie(): boolean  + checkHorizWin(BoardPosition pos, char p): boolean  + checkVertWin(BoardPosition pos, char p): boolean  + checkDiagWin(BoardPosition pos, char p): boolean  + whatsAtPos(BoardPosition pos): char  + isPlayerAtPos(BoardPosition pos, char player): boolean |

|  |
| --- |
| **AbsGameBoard** |
|  |
| + toString(): String |