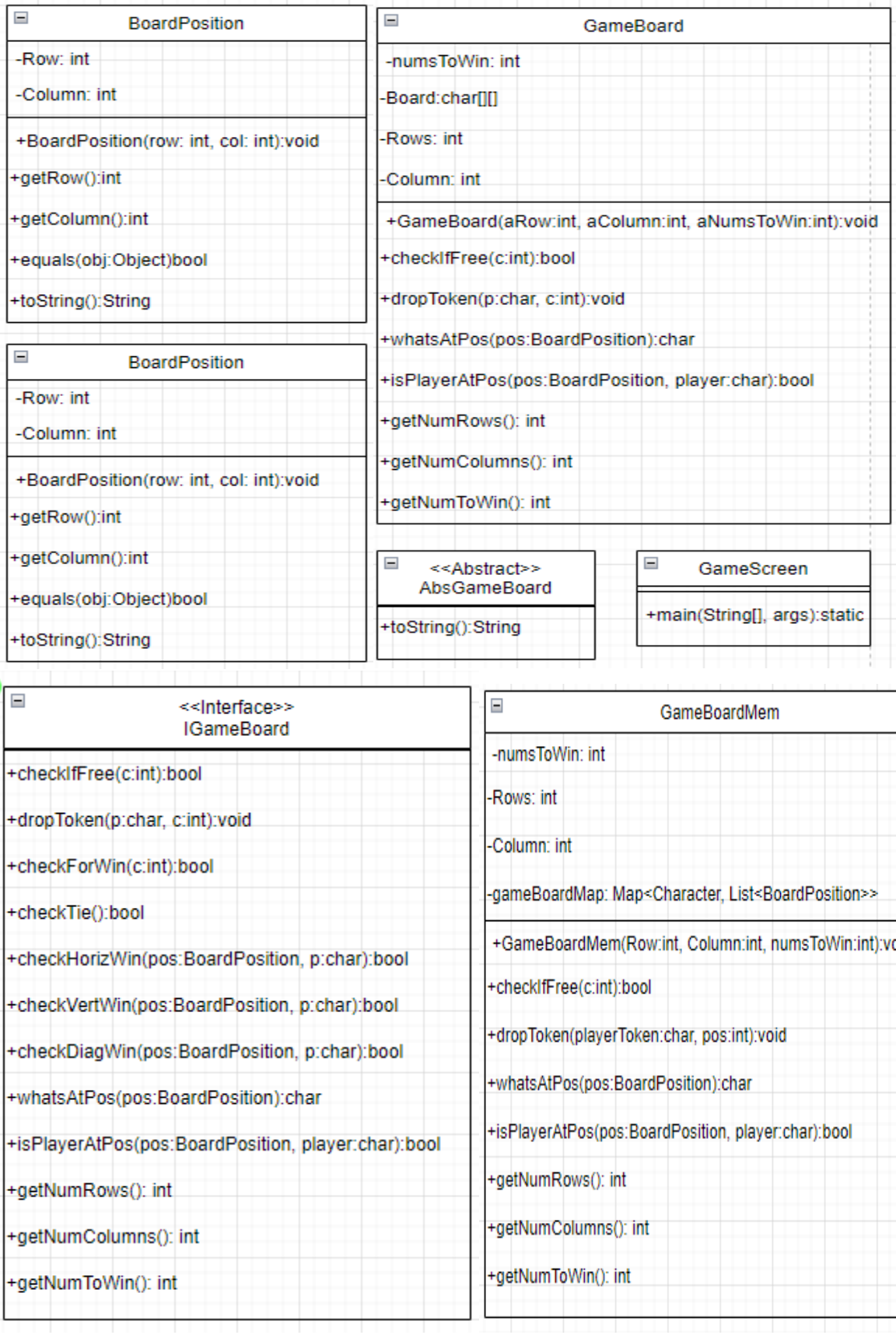
BoardPosition
-Row: int
-Column: int
+BoardPosition(row: int, col: int):void
+getRow():int
+getColumn():int
+equals(obj:Object)bool
+toString():String
BoardPosition
-Row: int
-Column: int
+BoardPosition(row: int, col: int):void
+getRow():int
+getColumn():int
+equals(obj:Object)bool
+toString():String
GameBoard
-numsToWin: int
-Board:char[][]
-Rows: int
-Column: int
+GameBoard(aRow:int, aColumn:int, aNumsToWin:int):void
+checkIfFree(c:int):bool
+dropToken(p:char, c:int):void
+whatsAtPos(pos:BoardPosition):char
+isPlayerAtPos(pos:BoardPosition, player:char):bool
+getNumRows(): int
+getNumColumns(): int
+getNumToWin(): int
<<Abstract>>
AbsGameBoard
+toString():String
GameScreen
+main(String[], args):static
<<Interface>>
IGameBoard
+checkIfFree(c:int):bool
+dropToken(p:char, c:int):void
+checkForWin(c:int):bool
+checkTie():bool
+checkHorizWin(pos:BoardPosition, p:char):bool
+checkVertWin(pos:BoardPosition, p:char):bool
+checkDiagWin(pos:BoardPosition, p:char):bool
+whatsAtPos(pos:BoardPosition):char
+isPlayerAtPos(pos:BoardPosition, player:char):bool
+getNumRows(): int
+getNumColumns(): int
+getNumToWin(): int
GameBoardMem
-numsToWin: int
-Rows: int
-Column: int
-gameBoardMap: Map<Character, List<BoardPosition>>
+GameBoardMem(Row:int, Column:int, numsToWin:int):void
+checkIfFree(c:int):bool
+dropToken(playerToken:char, pos:int):void
+whatsAtPos(pos:BoardPosition):char
+isPlayerAtPos(pos:BoardPosition, player:char):bool
+getNumRows(): int
+getNumColumns(): int
+getNumToWin(): int

CPSC 2150 Project 3 Report
Jared Alvarado, Jakob Padar, Arthur Garner
Requirements Analysis
Functional Requirements:
1. As a user, I need an interface so I can choose where to place my token.
2. As a user, I need a menu so I can choose to exit the game
3. As a user, I need a menu so I can choose to play the game.
4. As a user, I need an updated game board so I can see where the game stands.
5. As a user, I need the game to decide when there is a winner or □e so that I know who won or if we tied.
6. As a user, I need the game to show who's turn it is and show the board so I can see and make my next move.
7. As a user, I need the game to show who is the winner and show a congratulations message so I can end the game a start a new one.
8. As a user, I need the game to show me that the game is a □e so I can figure out how to win next time I play a game.
9. As a user, I need the game to ask if I want to play again after a game has ended so I can start another game right away.
10. As a user, I need the game to ask me to choose a new position if my token position is invalid so that I can keep playing and not be stopped by errors.
11. As a user, I need the game to tell me if I picked a full column so I can pick a new position for my token.
12. As a user, I need the game to tell me I won if I have X number of tokens horizontally, vertically, or diagonally on the gameboard so I can tell how I won and start a new game.
13. As a user, I need to be able to choose the fast or memory efficient version, so I can run the game best suited for my computer.
14. As a user, I need to be able to choose my token, so that I can represent myself on the board.

Non-Functional Requirements
1. The game must run using a command-line interface.
2. The game must be written in Java code.
3. The game must be expandable to a higher number of columns.
4. The game must be resizable each game.
5. The game must verify whether a token position is valid.
6. The players tokens need to be unique.
7. The bottom left board is always (0,0)