**Checkers In Java : Final Report**

**The Objective:**

The objective of our project was to construct a working checkers program in Java with graphics that are interactive in order for the user to be able to play. To be more specific, the requirements of the project dictated that the pieces should only be able to move and rest in valid positions on the board. The game should have a turn tracker which correctly determines the side which should move based upon a piece’s ability to move. The game should give players the ability to “king” their pieces when they reach the other side of the board. The game should also have the ability to determine who won after any given move on the board. Lastly the game must come with a feature which enables the user to see a persistent history of the previous games played. This record would include a record of who the players were, who was victorious, and how many turns it took to complete the game. All of these requirements were met in the project. The team decided to add a few extra features beyond the basic requirements. This included a live view of all of the captured pieces, a text box indicating whose turn it is, the ability for a player to resign, and a new game button to reset the game in the middle of any given game.

**Code Structure:**

In order to utilize graphics this project used the Abstract Window Toolkit and its associated graphical libraries as well as the Swing graphical interface libraries. On the highest level, the code was divided into two major classes. One of these for obvious reasons was the main and the other was the Game class. The main class consisted entirely of the basic framework and declaration of all of the graphical components. This included the window itself, the buttons, and the widgets to display to the user. The other class would define the Game and all of the components within it. As such the main class was used to declare an object of type Game as well as provide the functionality necessary to call for another object of the same class, with certain variables being reset to their initial states when beginning a new game. The main class also initiates two pop-up windows which prompt the users for their names in order to save them in the games record. The Game class begins with a constructor which initializes the interactive nature of the program, sets the game, and repaints the board for the player to begin. Inside the board class there is a very important function which dictates the majority of the graphical components of the program. This controls the graphical representation of the board, the pieces on the board, the king feature, the indicators of valid moves, and the collection of the pieces on the right hand side. The setUp function prompts the user for the names of the players and arranges the pieces on the board for the beginning of the game. The game is set up on a 2D array called board which stores the piece objects of the game. The legal moves and jumps are stored in their own respective lists. Those classes are composed of checkerMove objects which define a row and column a piece can move to. Then every time the piece is clicked on it accesses and displays the options to the user with all of the valid jumps and moves. All of the other functions dictate the behavior of the game and these piece objects. The function pieceJumps adds to the count of whatever pieces have been captured in order to tell the repaint function how many pieces to draw. The movePiece, jumpPiece, and isKing functions all perform the purposes which are indicated by their names. The endGame function checks for all of the pieces to determine if there is a winner for the game. The saveStats function writes all of the relevant data from previous games to a .txt file.

**Instructions To Run The Code:**

The code is not complicated to run. All that is required is the CheckersRemade.Java and the games.txt file which the program will write all of its results to. All of this can be pulled from the Git Repository and run so long as the user has Java installed in their system. The user should use the most recent version of Java if possible. This can be run in any Java friendly IDE or through the console fairly easily. If using a command line the user can simply type in the command line;

javac CheckersRemade.java

java CheckersRemade