CPSC 2150 Project 1

Checkers

Nadia Alexander
Mether Oke
Laura Castro Rosales
Keerthi Surisetty

Requirements Analysis

Functional Requirements:

1. As a player, I need to be able to move the pieces in order to get them to the other side of the board.
2. As a player, I need to be able to jump the other player's pawns in order to get rid of the pieces.
3. As a player, I need to be able to move all 12 pieces diagonally so I can move my pieces around the board throughout the game.
4. As a player, I need to be able to identify an empty space so that I can jump my opponent's pieces when it is an option.
5. As a player, I need to have the ability to king my piece when it gets to the other side of the board in order to have advanced moving abilities with that piece.
6. As a player, I need access to the rules of checkers in order to play the game correctly and beat player 2.
7. As a player, I need to be able to identify which pieces are the opposing player's in order to avoid moving the opposing player's pieces.
8. As a player, I can move one piece per turn in order to eventually get rid of the opposing player's pieces and win the game.
9. As a player, when my piece is kinged, I need to be able to differentiate between the kinged piece and the regular pieces in order to avoid being confused.
10. As a player I need to be able to identify pieces that are already kinged so I don't try to king them again.
11. As a player I need to be able to select only one piece to move on my turn in order to avoid moving multiple pieces at once during my turn.

12. As a player, I need to be able to identify the black tiles that are unplayable to avoid moving a piece into that spot.
13. As a player, once someone wins the game, playing a new game must be an option in order to give me the option to play again.
14. As a player, I need access to the coordinate system of the board in order to locate the piece I want to move.

Non-Functional Requirements

1. This program must be able to run on the terminal without errors.
2. The game board must be a 8 by 8 grid of rows in order for each player to have 12 pieces.
3. The program must prompt the user to reenter an input when the player enters an invalid input.
4. This program needs to be able to run on all operating systems.
5. The program must be able to identify which pieces were jumped in order to remove them from the board.
6. This program must be able to identify the player's own token and if that token is chosen it needs to prompt the player to choose another token.
7. This program needs to be able to identify when a piece has been kinged and allow that piece to move in 4 directions.
8. After each player's turn, the current state of the board must be printed out to see the status of each player's pieces.
9. The program must re-prompt the player to choose a different piece to move if they try to move the opposing player piece.
10. The program must prompt the player to choose a different piece to move if the player tries to move a piece that has no possible space to move to.
11. The program needs to give the players a chance to play again by re-prompting them once the game is over.
12. The program must re-prompt the player to choose a valid direction if the player tries to move a piece in a direction other than NE, NW, SW, or SE.
13. The program must prompt the player to choose a different piece to move if the player chooses coordinates that are outside of the 8 by 8 board.
14. The program must give the player the option whether or not to jump the opposing player's piece when the option arises.

15. The program must allow the player to only jump a piece once per term if the ability to jump the piece is an option for the current player.
16. The program must prompt player x to start each game to keep consistency in the beginning of each game.

System Design – (UML diagrams)