Chase Allen

Cjan8c

Checkers Game Data Model Concept

**Collect and display the player names**

A way to collect and display player names is to start the app in an empty state with a menu at the top of the screen. The menu will have a dropdown with the option to start a game with two computers, one human, or two humans. When the game is started, load the board and pop up alerts that allows the user to add names. The names will be stored in Strings and can be displayed at the top of the app to show who currently has the board to play. The Strings can be swapped for turns when the state switches after a player moves a checker piece.

Private String getName(){

TextInputDialog playerOne = new TextInputDialog();

Dialog.setTitle(“Enter Name”);

String player = dialog.getResult();

Return player;

}

**Determine who moves first and gets assigned the dark pieces**

A way to determine who moves first would be to generate a random number that is either 0 or 1. If it is a 0, then Player 1 will go first and be assigned the dark pieces. If the number is a 1, Player 2 will be assigned the dark pieces and first move.

Private void assign(){

Random rand = new Random();

Int n = rand.nextInt(1);

If(n == 0){

//order of strings in setPlayers will generate player one and two in that order

setPlayers (playerOne, playerTwo);

}else if(n == 1){

setPlayers(playerTwo, playerOne);

}

}

**Display a board with pieces the user can interact with if one or both are humans**

The board is loaded when players are chosen. They will be able to interact with the pieces assigned to them only when it is their turn which will be controlled by state monitoring and a bool value on the checker pieces that disables their movement when it is not their turn. After a player moves their piece, the state will change, it will display that it is the new users turn, and they will then be able to move their pieces.

Private void startGame(){

loadBoard();

playerOne = getName();

playerTwo = getName();

}

**Identify when there is a winner, loser, or if a draw has occurred**

When all the pieces of a color are removed from the board, that player loses and the other wins. This will be tracked by keeping an int of each number of pieces on the board for each player. When the int reaches zero then the player will lose.

int playerOnePieces = 12

int playerTwoPieces = 12

if(playerOnePieceLost()){

playerOnePieces--;

}

if(playerTwoPiecesLost(){

playerTwoPieces--;

}

**Preventing illegal moves**

Moves that are considered allowable will be forward, diagonal moves and jumps. A jump can only be done if the player is in a diagonally adjacent spot without a checker piece behind it. Illegal moves would include backwards moves, moving to a position with a piece already in it, and non-diagonal moves. The first idea I have to prevent illegal moves is to have each square of the board as an object that can be tracked. This is necessary so that you can keep track of what each square is and it’s adjacent pieces. Knowing where each piece is will make tracking illegal moves easier and could potentially be used to suggest moves for players when they click on a checker piece.