Lukas Heading

LPHZQD

Checkers Data Model

1. Checkers class/Object this acts as the overall container for the game. Also sets player names at the beginning before the match and assigns a color randomly.
    a. Methods
        i. Constructor method
    b. Variables
        i. The checker object has a board Object from the board class.
        ii. Two player objects from the player class.
        iii. An constant int variable for number of rows
        iv. An constant int variable for number of columns.
        v. An int variable called currentTurn, which is a counter starting on 1. It bumps at the end of a turn, dark pieces play on odd number and light pieces play on evens.
        vi. A variable called maxTurns: allows players to set max turn limit to declare draws or early winners.
2. Piece Class/Object: is piece is its own object that exists on the board.
    a. Methods
        i. Constructor(color, width, height)
        ii. getters/setters
    b. Variables
        i. Color color
        ii. Double width
        iii. Double height
        iv. Boolean is King
3. Board Class/object: This class/object also generates the pieces and controls objects movements on the board
    a. Methods

i. Constructor(numRows, numCols, boardHeight, boardWidth, color)
ii. validMove(): checks to see if it is an open space it is moving to, checks if it is trying to jump friendly piece also checks to see if it is going the correct direction, also checks for king status
iii. move(): moves the piece after checking validMove, also checks if there was an opponent's piece over where it moved and removes it from the array, also checks if it ends up in opponents final row and marks piece as a king.
iv. won(): after every round checks to see if any players are missing all pieces and declares a winner, also checks to see if max turn has been hit and can determine a winner if they have more pieces or a draw if they have the same.
v. getters/setters

b. Variables
i. An array to keep track of the pieces on the board
ii. Double boardHeight
iii. Double boardWidth
iv. Int numCols
v. Int numRows

4. Player Class/objects: Could create subclasses if we wanted to, one for human and one for computer. Then we could allow for computer to have difficulties and such
a. Methods
i. Constructor(name, color, isComputer)
ii. getters/setters

b. Variables
i. String name variable
ii. Color color
iii. Number of pieces
iv. Number of pieces captured from opponent

v. Boolean isComputer