Description of Back 2 Back Data Structure

Jordan Turley

The Back 2 Back game has a board, which consists of holes, or cells, in which you place pieces on. A piece has pegs that can be either 1 deep, where it does not protrude to the other side, or 2 deep, where it does protrude. The board can be turned around, to place a piece on the other side, and each piece can be rotated clockwise to place it on the board in a different orientation. In my implementation, I created a BoardCell class, which stores 2 booleans for if the front or back of the cell is covered, and 2 chars, to represent the color of the piece covering that side. I also created a Board class, which has a 2D array of BoardCells to represent the actual board. I also created a Piece class to represent the color and layout of a Piece. The Piece class has a 2D array of integers, which are 0, 1 or 2, representing the layout of the piece and the depth of each peg. All 11 of the pieces are created statically and are stored in a static array. The Piece class has a rotate method which allows you to rotate that piece, and also keeps track if you have placed the piece or not, so you cannot place the same piece on the board twice. The Board class lets you place Piece objects on the board using an add method. This add method checks to make sure the piece is not hanging off the edge of the board, and that the piece does not illegally overlap with another piece. There is also a switchSide method which lets you flip the board from the front to the back to see the other side, so you can place pieces on either side. I decided to use only one array to keep track of the board, and then flip the array horizontally when you flip the board. There is also a Game class, which lets you enter text into the command line to actually play the game.