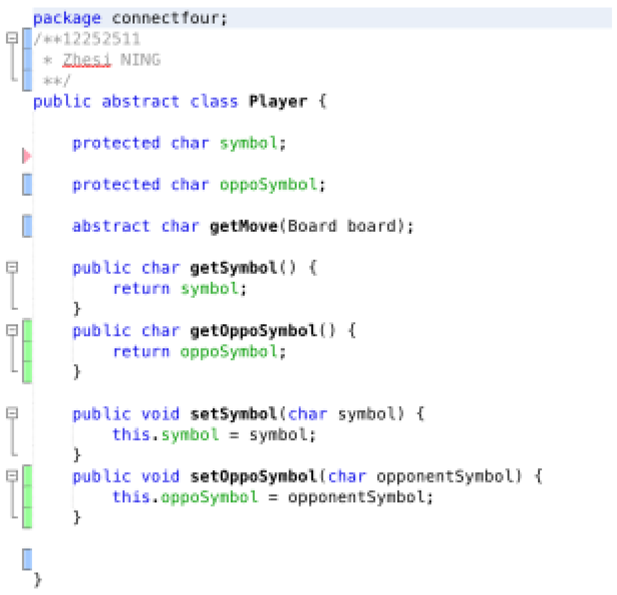
12252511
Zhesi Ning
AI_2_ Report

I. Game setup and ASCII-art display:
In this java version of mortified connect-four we have two types of player :
Computer player and human player

Computer player takes an argument which set the computer player's effort level.

Both computer and human player all extend an player interface.



```
package connectfour;
/**12252511
 * Zhesi NING
 **/
public abstract class Player {

    protected char symbol;

    protected char oppoSymbol;

    abstract char getMove(Board board);

    public char getSymbol() {
        return symbol;
    }
    public char getOppoSymbol() {
        return oppoSymbol;
    }

    public void setSymbol(char symbol) {
        this.symbol = symbol;
    }
    public void setOppoSymbol(char opponentSymbol) {
        this.oppoSymbol = opponentSymbol;
    }

}
```

II. end-of-game detection
Function getWinner() in GameBoard fins the winner if there is one.
And the public boolean isGameOver() checks if there is a winner and number of moves left is 0 then ends the game.

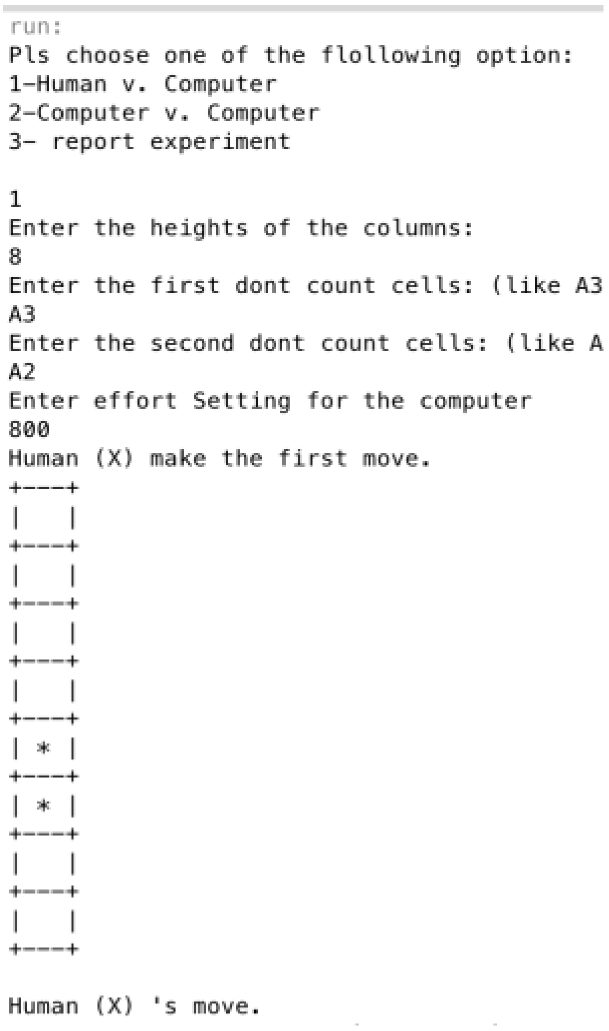
III:human-vs-computer play

Running the program mainGame class will take input from the user asking for which mode they want to play :
1 human vs computer
2 computer vs computer
3 experiment for the report

and we can set the heights for the columns up to 12 and which cell to ignore as well

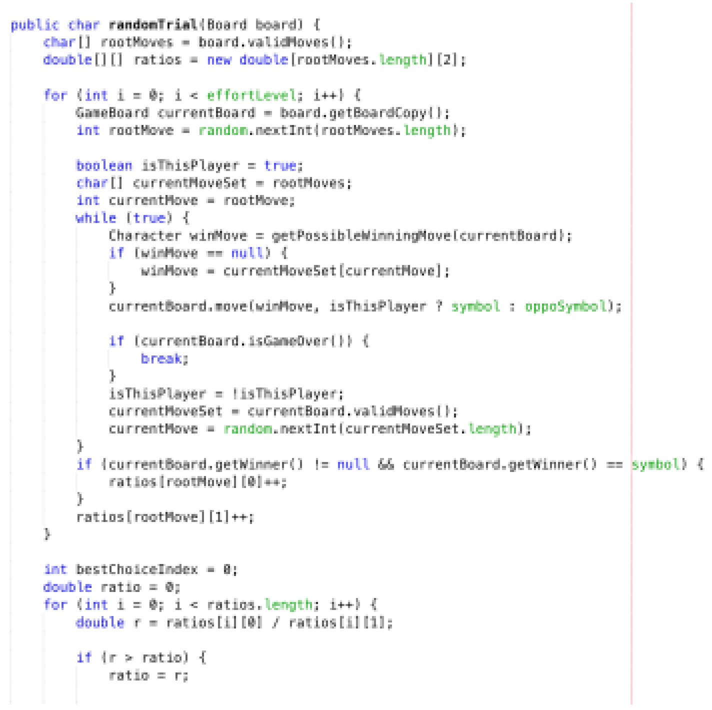
IV: computer-vs-computer play

Choosing option 2 will give you enter effort level for both of the ai player. Which than return wich player wins the game.
Ps. Computer player make a choice of move based on Monte Carlo technique.



```
run:
Pls choose one of the flollowing option:
1-Human v. Computer
2-Computer v. Computer
3- report experiment

1
Enter the heights of the columns:
8
Enter the first dont count cells: (like A3
A3
Enter the second dont count cells: (like A
A2
Enter effort Setting for the computer
800
Human (X) make the first move.
+----+
|    |
+----+
|    |
+----+
|    |
+----+
|    |
+----+
| *  |
+----+
| *  |
+----+
|    |
+----+
|    |
+----+


Human (X) 's move.
```

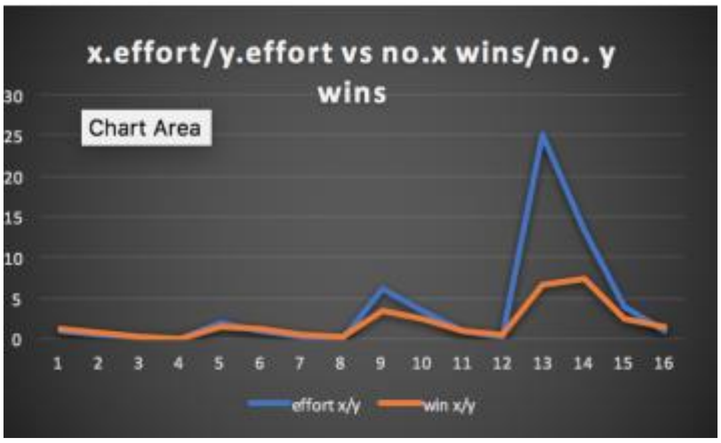
12252511
Zhesi Ning
AI_2_ Report

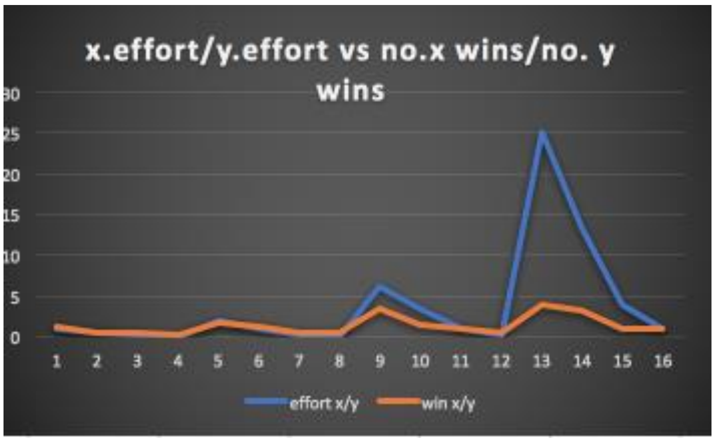


```java
public char randomTrial(Board board) {
    char[] rootMoves = board.validMoves();
    double[][] ratios = new double[rootMoves.length][2];

    for (int i = 0; i < effortLevel; i++) {
        GameBoard currentBoard = board.getBoardCopy();
        int rootMove = random.nextInt(rootMoves.length);

        boolean isThisPlayer = true;
        char[] currentMoveSet = rootMoves;
        int currentMove = rootMove;
        while (true) {
            Character winMove = getPossibleWinningMove(currentBoard);
            if (winMove == null) {
                winMove = currentMoveSet[currentMove];
            }
            currentBoard.move(winMove, isThisPlayer ? symbol : oppoSymbol);

            if (currentBoard.isGameOver()) {
                break;
            }
            isThisPlayer = !isThisPlayer;
            currentMoveSet = currentBoard.validMoves();
            currentMove = random.nextInt(currentMoveSet.length);
        }
        if (currentBoard.getWinner() != null && currentBoard.getWinner() == symbol) {
            ratios[rootMove][0]++;
        }
        ratios[rootMove][1]++;
    }

    int bestChoiceIndex = 0;
    double ratio = 0;
    for (int i = 0; i < ratios.length; i++) {
        double r = ratios[i][0] / ratios[i][1];

        if (r > ratio) {
            ratio = r;
```

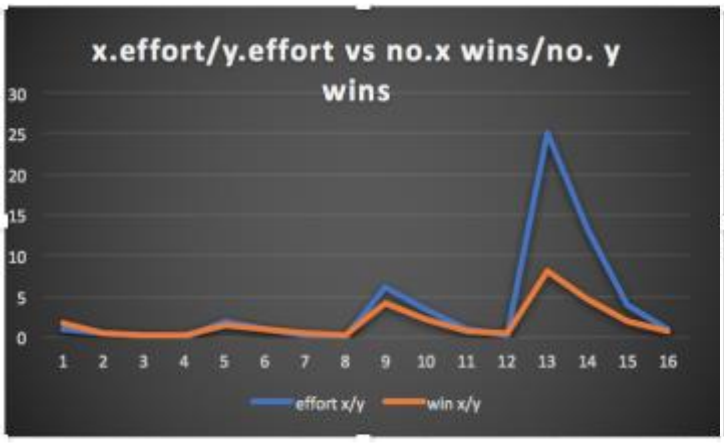
Computer choice of move:
Unless there is a move that wins will led to the winning position and ends the game, the computer will conduct a monte carlo random walk sequences of moves for each player till the end game, then choose the best ratio of wins for the first move.

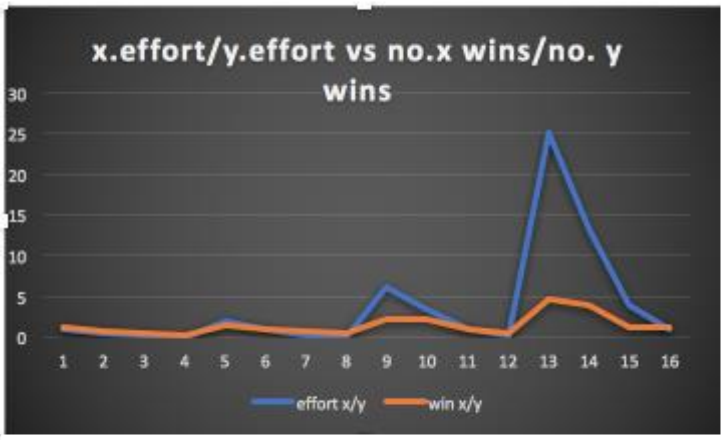
Systematic experimentation
input height "5 5 3 5 5 5 6 7", ignore cell"A3 B1"   input height "5 5 5 6 7",ignore cell "B3 A1"





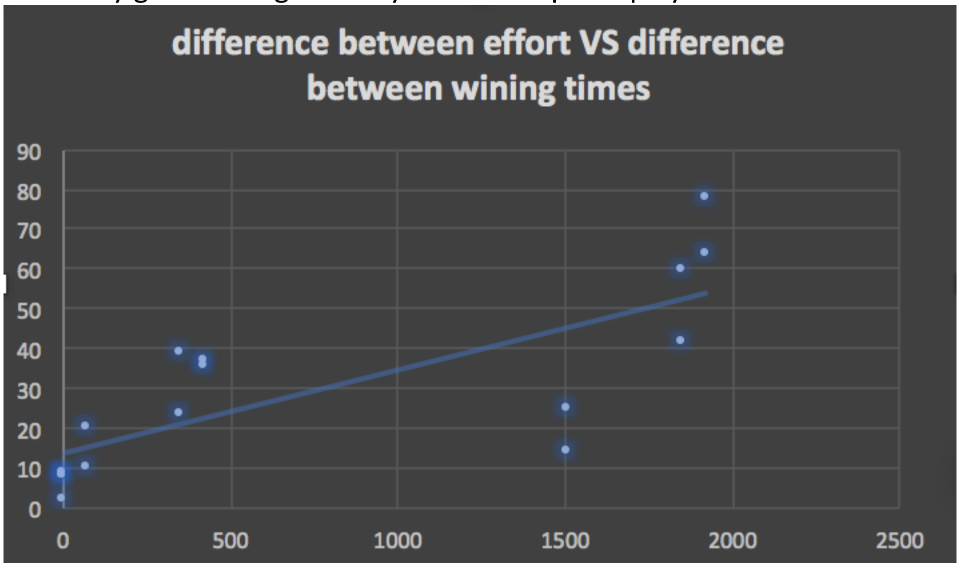
12252511
Zhesi Ning
AI_2_ Report
input height "4 6 7 5 5 6 7", ignore cell"B2 A4"   input height "7 6 3 5 6 7", ignore cell"A4 B4"





above graph was plot using the data generate from the experimentation that let computer to play itself with one player using 80 or 150 or 500 or 2000 sequences, and the other player also using 80 or 150 or 500 or 2000 as well. And for all 16 combinations each combination we run it 100 times. Then we plot the graph x-effort/y-effort VS no of time x win/no of times y wins. We ca observe that if the effort level ratio become larger the ratio between number of win (or say the gap) will be widen as well. But if the effort ratio is 1, then the number of x wins and number of y wins are almost 1 to 1 as well.
Thus, we discovered that the effort level of a player is linear proportional.

After playing with the computer many time, I feel that by using monte carlo search, the computer was able to find an optimal solution to reach the winning position. It was quite difficult to compete with the computer player, even with a low effort level. But higher effort definitely gives stronger ability to the computer player.


difference between effort VS difference between wining times