February 24, 2019

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

class Board{

    public class Point{
        int x, y;
        Point(int x, int y){
            this.x = x;
            this.y = y;
        }

        @Override
        public String toString(){
            return "["+x+", "+y+"]";
        }

        @Override
        public boolean equals(Object o){
            return o.hashCode()==this.hashCode();
        }

        @Override
        public int hashCode() {
            return Integer.parseInt(x+""+y);
        }
    }

    private final char[][] board;
    int WScore, BScore, remaining;
    private final char boardX[] = new char[]{’A’,’B’,’C’,’D’,’E’,’F’,’G’,’H’};

    public Board(){
        board = new char[][]{
            {’_’,’_’,’_’,’_’,’_’,’_’,’_’,’_’,},
            {’_’,’_’,’_’,’_’,’_’,’_’,’_’,’_’,},
            {’_’,’_’,’_’,’_’,’_’,’_’,’_’,’_’,},
            {’_’,’_’,’_’,’W’,’B’,’_’,’_’,’_’,},
            {’_’,’_’,’_’,’B’,’W’,’_’,’_’,’_’,},
            {’_’,’_’,’_’,’_’,’_’,’_’,’_’,’_’,},
            {’_’,’_’,’_’,’_’,’_’,’_’,’_’,’_’,},
            {’_’,’_’,’_’,’_’,’_’,’_’,’_’,’_’,},
        };
    }

    private void findPlaceableLocations(char player, char opponent, HashSet<Point> placeablePositions){
        for(int i=0;i<8;++i){
            for(int j=0;j<8;++j){
                if(board[i][j] == opponent){
                    int I = i, J = j;
                if(i-1>=0 && j-1>=0 && board[i-1][j-1] == '_'){
                    i = i+1; j = j+1;
                    while(i<7 && j<7 && board[i][j] == opponent){i++;j++;}
                    if(i<=7 && j<=7 && board[i][j] == player) placeablePositions.add(new Point(I-1, J-1));
                }
                i=I;j=J;
                if(i-1>=0 && board[i-1][j] == '_'){
                    i = i+1;
                    while(i<7 && board[i][j] == opponent) i++;
                    if(i<=7 && board[i][j] == player) placeablePositions.add(new Point(I-1, J));
                }
                i=I;
                if(i-1>=0 && j+1<=7 && board[i-1][j+1] == '_'){
                    i = i+1; j = j-1;
                    while(i<7 && j>0 && board[i][j] == opponent){i++;j--;}
                    if(i<=7 && j>=0 && board[i][j] == player) placeablePositions.add(new Point(I-1, J+1));
                }
                i=I;j=J;
                if(j-1>=0 && board[i][j-1] == '_'){
                    j = j+1;
                    while(j<7 && board[i][j] == opponent)j++;
                    if(j<=7 && board[i][j] == player) placeablePositions.add(new Point(I, J-1));
                }
                j=J;
                if(j+1<=7 && board[i][j+1] == '_'){
                    j=j-1;
                    while(j>0 && board[i][j] == opponent)j--;
                    if(j>=0 && board[i][j] == player) placeablePositions.add(new Point(I, J+1));
                }
                j=J;
                if(i+1<=7 && j-1>=0 && board[i+1][j-1] == '_'){
                    i=i-1;j=j+1;
                    while(i>0 && j<7 && board[i][j] == opponent){i--;j++;}
                    if(i>=0 && j<=7 && board[i][j] == player) placeablePositions.add(new Point(I+1, J-1));
                }
                i=I;j=J;
                if(i+1 <= 7 && board[i+1][j] == '_'){
                    i=i-1;
                    while(i>0 && board[i][j] == opponent) i--;
                    if(i>=0 && board[i][j] == player) placeablePositions.add(new Point(I+1, J));
                }
                i=I;
                if(i+1 <= 7 && j+1 <=7 && board[i+1][j+1] == '_'){
                    i=i-1;j=j-1;
                    while(i>0 && j>0 && board[i][j] == opponent){i--;j--;}
                    if(i>=0 && j>=0 && board[i][j] == player)placeablePositions.add(new Point(I+1, J+1));
                }
                i=I;j=J;
                }
            }
    }
}

public void displayBoard(Board b){
    System.out.print("\n  ");
    for(int i=0;i<8;++i)System.out.print(boardX[i]+" ");
    System.out.println();
    for(int i=0;i<8;++i){
        System.out.print((i+1)+" ");
        for(int j=0;j<8;++j)
```

```
            System.out.print(b.board[i][j]+" ");
        System.out.println();
    }
    System.out.println();
}

public int gameResult(Set<Point> whitePlaceableLocations, Set<Point> blackPlaceableLocations){
    updateScores();
    if(remaining == 0){
        if(WScore > BScore) return 1;
        else if(BScore > WScore) return -1;
        else return 0; //Draw
    }
    if(WScore==0 || BScore == 0){
        if(WScore > 0) return 1;
        else if(BScore > 0) return -1;
    }
    if(whitePlaceableLocations.isEmpty() && blackPlaceableLocations.isEmpty()){
        if(WScore > BScore) return 1;
        else if(BScore > WScore) return -1;
        else return 0; //Draw
    }
    return -2;
}

public HashSet<Point> getPlaceableLocations(char player, char opponent){
    HashSet<Point> placeablePositions = new HashSet<>();
    findPlaceableLocations(player, opponent, placeablePositions);
    return placeablePositions;
}

public void showPlaceableLocations(HashSet<Point> locations, char player, char opponent){
    for(Point p:locations)
        board[p.x][p.y]=’*’;
    displayBoard(this);
    for(Point p:locations)
        board[p.x][p.y]=’_’;
}

//Although we know that if W is player, O will be the opponent but still...
public void placeMove(Point p, char player, char opponent){
    int i = p.x, j = p.y;
    board[i][j] = player;
    int I = i, J = j;

    if(i-1>=0 && j-1>=0 && board[i-1][j-1] == opponent){
        i = i-1; j = j-1;
        while(i>0 && j>0 && board[i][j] == opponent){i--;j--;}
        if(i>=0 && j>=0 && board[i][j] == player) {while(i!=I-1 && j!=J-1)board[++i][++j]=player;}
    }
    i=I;j=J;
    if(i-1>=0 && board[i-1][j] == opponent){
        i = i-1;
        while(i>0 && board[i][j] == opponent) i--;
        if(i>=0 && board[i][j] == player) {while(i!=I-1)board[++i][j]=player;}
    }
    i=I;
    if(i-1>=0 && j+1<=7 && board[i-1][j+1] == opponent){
        i = i-1; j = j+1;
        while(i>0 && j<7 && board[i][j] == opponent){i--;j++;}
```

```
            if(i>=0 && j<=7 && board[i][j] == player) {while(i!=I-1 && j!=J+1)board[++i][--j] = player;}
        }
        i=I;j=J;
        if(j-1>=0 && board[i][j-1] == opponent){
            j = j-1;
            while(j>0 && board[i][j] == opponent)j--;
            if(j>=0 && board[i][j] == player) {while(j!=J-1)board[i][++j] = player;}
        }
        j=J;
        if(j+1<=7 && board[i][j+1] == opponent){
            j=j+1;
            while(j<7 && board[i][j] == opponent)j++;
            if(j<=7 && board[i][j] == player) {while(j!=J+1)board[i][--j] = player;}
        }
        j=J;
        if(i+1<=7 && j-1>=0 && board[i+1][j-1] == opponent){
            i=i+1;j=j-1;
            while(i<7 && j>0 && board[i][j] == opponent){i++;j--;}
            if(i<=7 && j>=0 && board[i][j] == player) {while(i!=I+1 && j!=J-1)board[--i][++j] = player;}
        }
        i=I;j=J;
        if(i+1 <= 7 && board[i+1][j] == opponent){
            i=i+1;
            while(i<7 && board[i][j] == opponent) i++;
            if(i<=7 && board[i][j] == player) {while(i!=I+1)board[--i][j] = player;}
        }
        i=I;

        if(i+1 <= 7 && j+1 <=7 && board[i+1][j+1] == opponent){
            i=i+1;j=j+1;
            while(i<7 && j<7 && board[i][j] == opponent){i++;j++;}
            if(i<=7 && j<=7 && board[i][j] == player)while(i!=I+1 && j!=J+1)board[--i][--j] = player;}
    }

    public void updateScores(){
        WScore = 0; BScore = 0; remaining = 0;
        for(int i=0;i<8;++i){
            for(int j=0;j<8;++j){
                if(board[i][j]==’W’)WScore++;
                else if(board[i][j]==’B’)BScore++;
                else remaining++;
            }
        }
    }

    public int coordinateX(char x){
        for(int i=0;i<8;++i)if(boardX[i]==Character.toLowerCase(x)||boardX[i]==Character.toUpperCase(x))return
        return -1; // Illegal move received
    }
}

public class Reversi{
    public static void twoPlayers(Board b){
        Scanner scan = new Scanner(System.in);
        Board.Point move = b.new Point(-1, -1);
        System.out.println("Black Moves first");

        int result;
        Boolean skip;
        String input;
```

```
while(true){
    skip = false;

    HashSet<Board.Point> blackPlaceableLocations = b.getPlaceableLocations(’B’, ’W’);
    HashSet<Board.Point> whitePlaceableLocations = b.getPlaceableLocations(’W’, ’B’);

    b.showPlaceableLocations(blackPlaceableLocations, ’B’, ’W’);
    result = b.gameResult(whitePlaceableLocations, blackPlaceableLocations);

    if(result == 0){System.out.println("It is a draw.");break;}
    else if(result==1){System.out.println("White wins: "+b.WScore+":"+b.BScore);break;}
    else if(result==-1){System.out.println("Black wins: "+b.BScore+":"+b.WScore);break;}

    if(blackPlaceableLocations.isEmpty()){
            System.out.println("Black needs to skip... Passing to white");
            skip = true;
    }

    if(!skip){
        System.out.println("Place move (Black): ");
        input = scan.next();
        move.y = b.coordinateX(input.charAt(0));
        move.x = (Integer.parseInt(input.charAt(1)+"")-1);

        while(!blackPlaceableLocations.contains(move)){
            System.out.println("Invalid move!\n\nPlace move (Black): ");
            input = scan.next();
            move.y = b.coordinateX(input.charAt(0));
            move.x = Integer.parseInt((input.charAt(1)+""))-1;
        }
        b.placeMove(move, ’B’, ’W’);
        b.updateScores();
        System.out.println("\nBlack: "+b.BScore+" White: "+b.WScore);
    }
    skip = false;

    whitePlaceableLocations = b.getPlaceableLocations(’W’, ’B’);
    blackPlaceableLocations = b.getPlaceableLocations(’B’, ’W’);

    b.showPlaceableLocations(whitePlaceableLocations, ’W’, ’B’);
    result = b.gameResult(whitePlaceableLocations, blackPlaceableLocations);

    if(result==0){System.out.println("It is a draw.");break;}
    else if(result==1){System.out.println("White wins: "+b.WScore+":"+b.BScore);break;}
    else if(result==-1){System.out.println("Black wins: "+b.BScore+":"+b.WScore);break;}

    if(whitePlaceableLocations.isEmpty()){
            System.out.println("White needs to skip... Passing to Black");
            skip = true;
    }

    if(!skip){
    System.out.println("Place move (White): ");
    input = scan.next();
    move.y = b.coordinateX(input.charAt(0));
    move.x = (Integer.parseInt(input.charAt(1)+"")-1);

    while(!whitePlaceableLocations.contains(move)){
        System.out.println("Invalid move!\n\nPlace move (White): ");
```

```
                input = scan.next();
                move.y = b.coordinateX(input.charAt(0));
                move.x = (Integer.parseInt(input.charAt(1)+"")-1);
            }
            b.placeMove(move, ’W’, ’B’);
            b.updateScores();
            System.out.println("\nWhite: "+b.WScore+" Black: "+b.BScore);
            }
        }
    }

    public static void main(String[] args){
        Board b = new Board();
        twoPlayers(b);
    }
}
```