Assignment: Implement Tic Tac Toe using Min Max algorithm
Batch: A1
Roll no: 3330

TicTacToe.java

```
package tictactoe;
import java.util.Scanner;
import java.util.Random;
public class TicTacToe {
	public static final Random RANDOM=new Random();

	 public static void main(String[] args) {

		Board b=new Board();
		Scanner sc =new Scanner(System.in);
		b.DisplayBoard();

		System.out.println("Select turn:\n 1.Computer(X) / 2.User(O):");
		int ch=sc.nextInt();

		if(ch==Board.PlayerX){

		   Point p=new Point(RANDOM.nextInt(3),RANDOM.nextInt(3));
		    b.PlaceMove(p, Board.PlayerX);
		    b.DisplayBoard();
		 }

		while(!b.isGameOver()){

		   boolean moveOk=true;
		   do {

		         if(!moveOk)
		          System.out.println("Cell already filled");

		        System.out.println("Your move:");
		        Point userMove=new Point(sc.nextInt(),sc.nextInt());
		        moveOk=b.PlaceMove(userMove, Board.PlayerO);

		   }while(!moveOk);

		   b.DisplayBoard();

		    if(b.isGameOver())
		        break;

		     b.MinMax(0,Board.PlayerX);
		     System.out.println("Computer choose position: "+b.ComputerMove);
		     b.PlaceMove(b.ComputerMove,Board.PlayerX);
		     b.DisplayBoard();

		  }//close while(!b.isGameOver())

		if(b.hasPlayerWon(Board.PlayerX))
		   System.out.println("You Lost!!");


		else if(b.hasPlayerWon(Board.PlayerO))
		    System.out.println("You Win!!");
```

```
            else
               System.out.println("Draw");

        }//close main()
}
```

Board.java

```
package tictactoe;
import java.util.*;
public class Board {

       public static final int NoPlayer =0;
       public static final int PlayerX=1;
       public static final int PlayerO=2;

       private int[][]board =new int[3][3];
       public Point ComputerMove;

       public boolean isGameOver(){
            return hasPlayerWon(PlayerX)||hasPlayerWon(PlayerO)||getAvailableCells().isEmpty();
       }

       public boolean hasPlayerWon(int Player) {


       if((board[0][0]==board[1][1]&&board[0][0]==board[2][2]&&board[0][0]==Player)||(board[0][2]=
=board[1][1]
                           &&board[0][2]==board[2][0]&&board[0][2]==Player)){
                    return true;
             }

              for(int i=0;i<3;i++){

       if((board[i][0]==board[i][1]&&board[i][0]==board[i][2]&&board[i][0]==Player)||(board[0][i]=
=board[1][i]
                                  &&board[0][i]==board[2][i]&&board[0][i]==Player)){
                    return true;
                }
        }

              return false;
        }//close hasPlayerWon

        public List<Point>getAvailableCells(){
              List<Point> AvailableCells=new ArrayList<>();
                  for(int i=0;i<3;i++){
                        for(int j=0;j<3;j++) {
                           if(board[i][j]==NoPlayer){
                               AvailableCells.add(new Point(i,j));
                           }
                     }
                   }
              return AvailableCells;
        }

        public boolean PlaceMove(Point point,int Player){
               if(board[point.x][point.y]!=NoPlayer){
                    return false;
               }
             board[point.x][point.y]=Player;
```

```
     return true;
}
public void DisplayBoard(){
    System.out.println();
    for(int i=0;i<3;i++){
       for(int j=0;j<3;j++){
             String value="_";


     if(board[i][j]==PlayerX){
         value="X";
   }
else if(board[i][j]==PlayerO){
          value="0";
      }
     System.out.print(value+" ");
   }
   System.out.println();
}
System.out.println();
}
public int MinMax(int Depth,int turn){

       if(hasPlayerWon(PlayerX))
        return 1;

    if(hasPlayerWon(PlayerO))
       return -1;

     List<Point> availableCells=getAvailableCells();

     if(availableCells.isEmpty())
      return 0;

    int min=Integer.MAX_VALUE;
    int max=Integer.MIN_VALUE;

    for(int i=0;i<availableCells.size();i++){

       Point point=availableCells.get(i);

          if(turn==PlayerX){

               PlaceMove(point,PlayerX);
             int currentScore=MinMax(Depth+1,PlayerO);
             max=Math.max(currentScore, max);

             if(Depth==0)
              System.out.println("Computer score for position "+point+"="+currentScore);


            if(currentScore>=0)
             if(Depth==0)
                    ComputerMove=point;


             if(currentScore==1)
              board[point.x][point.y]=NoPlayer;


           }//close if(turn==PlayerX)
```

```
                else if(turn==PlayerO){

                     PlaceMove(point,PlayerO);
                    int currentScore=MinMax(Depth+1,PlayerX);
                    min=Math.min(currentScore,min);

                    if(min==-1){
                        board[point.x][point.y]=NoPlayer;
                        break;
                     }
                  }//close else if

                board[point.x][point.y]=NoPlayer;

            }//close for loop

              return turn==PlayerX?max:min;
        }//close minmax()
}
```

PointAndScore.java

```
package tictactoe;
public class PointAndScore {
            public int score;
            public Point point;

            public PointAndScore(int score,Point point){

                  this.score=score;
                this.point=point;
            }
}
```

Point.java

```
package tictactoe;
public class Point {

      public int x,y;

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

            return "["+x+","+y+"]";
      }

}
```

OUTPUT

```
_ _ _
_ _ _
_ _ _

Select turn:
 1.Computer(X) / 2.User(O):
1
```

```
_ _ X
_ _ _
_ _ _

Your move:
1
1

_ _ X
_ 0 _
_ _ _

Computer score for position [0,0]=0
Computer score for position [0,1]=0
Computer score for position [1,0]=0
Computer score for position [1,2]=0
Computer score for position [2,0]=0
Computer score for position [2,1]=0
Computer score for position [2,2]=0
Computer choose position: [2,2]

_ _ X
_ 0 _
_ _ X

Your move:
1
2

_ _ X
_ 0 0
_ _ X

Computer score for position [0,0]=-1
Computer score for position [0,1]=-1
Computer score for position [1,0]=0
Computer score for position [2,0]=-1
Computer score for position [2,1]=-1
Computer choose position: [1,0]

_ _ X
X 0 0
_ _ X

Your move:
0
1

_ 0 X
X 0 0
_ _ X

Computer score for position [0,0]=-1
Computer score for position [2,0]=-1
Computer score for position [2,1]=0
Computer choose position: [2,1]

_ 0 X
X 0 0
_ X X
```

```
Your move:
2
0

_ 0 X
X 0 0
0 X X

Computer score for position [0,0]=0
Computer choose position: [0,0]

X 0 X
X 0 0
0 X X

Draw
```