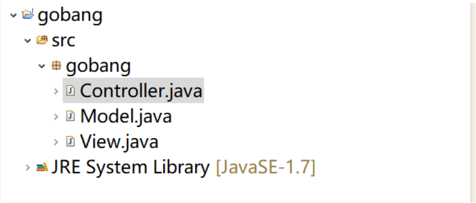
五子棋

一、　代码:

①项目的文件:



```
gobang
  src
    gobang
      Controller.java
      Model.java
      View.java
  JRE System Library [JavaSE-1.7]
```

②Model.java:

```
package gobang;
public class Model {
    public static final int BLACK=1;
    public static final int WHITE=-1;
    public static final int SPACE=0;
    public static final int WIDTH=12;              //定义WIDTH=12, 可修改
    private int[][] map=new int[WIDTH][WIDTH];
    private int lastRow,lastCol,lastColor;
    private Model(){}                   //采用单例模式
    private static Model instance=null;
    public static Model getInstance(){
        if(instance==null){
            instance=new Model();
        }
        return instance;
    }
    public boolean playChess(int row,int col,int color){    //在棋盘模型（二维数组）中
保存下棋信息
        if(row>=1&&row<=WIDTH&&col>=1&&col<=WIDTH&&(color==WHITE||color==BLACK)){
            if(map[row-1][col-1]==SPACE){
                map[row-1][col-1]=color;
                lastCol=col;
                lastRow=row;
                lastColor=color;
                return true;
            }
            else{
                View.getinstance().outPutInfo("Sorry,there is already a piece in 
this position.");
            }
        }
        else{
            View.getinstance().outPutInfo("Out of bounds!please try again.");
        }
        return false;
    }
    public int getChessInfo(int row,int col){                //获取row行col列的棋子颜色
        if(row>=1&&row<=WIDTH&&col>=1&&col<=WIDTH){
            return map[row-1][col-1];
        }
        else{
            View.getinstance().outPutInfo("Out of bounds!");
            return 0;
        }
    }
    public int whoWin(){              //判断是否有赢家产生，若有则返回赢家所执棋子颜色
        int row,col,sum=1;
        for(row=lastRow+1;row<lastRow+5;row++){
            if(row>=1&&row<=WIDTH){
                if(map[row-1][lastCol-1]==lastColor){
                    sum++;
                }
                else{break;}
            }
            else{break;}
        }
        for(row=lastRow-1;row>lastRow-5;row--){
            if(row>=1&&row<=WIDTH){
                if(map[row-1][lastCol-1]==lastColor){
                    sum++;
                }
                else{break;}
            }
            else{break;}
        }
        if(sum==5){
            return lastColor;
        }
        sum=1;
        for(col=lastCol+1;col<lastCol+5;col++){
            if(col>=1&&col<=WIDTH){
                if(map[lastRow-1][col-1]==lastColor){
                    sum++;
                }
                else{break;}
            }
            else{break;}
        }
        for(col=lastCol-1;col>lastCol-5;col--){
            if(col>=1&&col<=WIDTH){
                if(map[lastRow-1][col-1]==lastColor){
                    sum++;
                }
                else{break;}
            }
            else{break;}
        }
        if(sum==5){
            return lastColor;
        }
        sum=1;
        for(row=lastRow+1,col=lastCol+1;row<lastRow+5&&col<lastCol+5;row++,col++){
            if(row>=1&&row<=WIDTH&&col>=1&&col<=WIDTH){
                if(map[row-1][col-1]==lastColor){
                    sum++;
                }
                else{break;}
            }
            else{break;}
        }
        for(row=lastRow-1,col=lastCol-1;row>lastRow-5&&col>lastCol-5;row--,col--){
            if(row>=1&&row<=WIDTH&&col>=1&&col<=WIDTH){
                if(map[row-1][col-1]==lastColor){
                    sum++;
                }
                else{break;}
            }
            else{break;}
        }
        if(sum==5){
            return lastColor;
        }
        sum=1;
        for(row=lastRow-1,col=lastCol+1;row>lastRow-5&&col<lastCol+5;row--,col++){
            if(row>=1&&row<=WIDTH&&col>=1&&col<=WIDTH){
                if(map[row-1][col-1]==lastColor){
                    sum++;
                }
                else{break;}
```

```
            }
            else{break;}
        }
        for(row=lastRow+1,col=lastCol-1;row<lastRow+5&&col>lastCol-5;row++,col--){
            if(row>=1&&row<=WIDTH&&col>=1&&col<=WIDTH){
                if(map[row-1][col-1]==lastColor){
                    sum++;
                }
                else{break;}
            }
            else{break;}
        }
        if(sum==5){
            return lastColor;
        }
        return SPACE;
    }
    public void withdraw(){                             //悔棋
        map[lastRow-1][lastCol-1]=SPACE;
    }
    public boolean alreadyWithdraw(){                   //判断是否刚进行过一次悔棋
        return map[lastRow-1][lastCol-1]==SPACE;
    }
}
```

③View.java:

```
package gobang;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class View {
    private View(){}
    private static View instance=null;
    public static View getinstance(){
        if(instance==null)
            instance=new View();
        return instance;
    }
    public void input(){                  //获取下棋位置信息
        BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
        String s;
        String w=new String("w");
        String color=Controller.getinstance().getColor();
        try {
            System.out.println(color+" please row");
            s=in.readLine();
            if(s.equals(w)){            //如果用户输入w，则表示他要悔棋
                 if(Model.getInstance().alreadyWithdraw())     //判断他是否才进行过一次悔棋
                     outPutInfo("Sorry.You have retracted a false move in the chess game.");
                 else
                     Controller.getinstance().userWithdraw();
            }
            else{
                 int row=Integer.parseInt(s);
                 System.out.println(color+" please col");
                 s=in.readLine();
                 if(s.equals(w)){
                     if(Model.getInstance().alreadyWithdraw())
                         outPutInfo("Sorry.You have retracted a false move in the chess game.");
                     else
                         Controller.getinstance().userWithdraw();
                 }
                 else{
                     int col=Integer.parseInt(s);
                     Controller.getinstance().userPlayChess(row, col);     //将下棋位置传递给Controller执行下棋操作
                 }
            }
        } catch (IOException e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
    public void inputFirstColor(){       //输入先手棋子颜色
        BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
        String s;
        try {
            System.out.println("Please choose the first color.Input 1 for black or 2 for white.");
            while(true){
                 s=in.readLine();
                 int color=Integer.parseInt(s);
                 if(color==1){
                     Controller.getinstance().modifyColor(Model.BLACK);
                     break;
                 }
                 else if(color==2){
                     Controller.getinstance().modifyColor(Model.WHITE);
                     break;
                 }
                 else{
                     System.out.println("Input error, please re-enter it correctly.");
                 }
            }

        } catch (IOException e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
    public void outPutInfo(String s){    //向用户输出信息
        System.out.println(s);
    }
    public void outPutMap(){             //输出当前棋盘
        int row,col;
        for(row=1;row<=Model.WIDTH;row++){
            for(col=1;col<=Model.WIDTH;col++){
                 switch(Model.getInstance().getChessInfo(row, col)){
                 case Model.WHITE:
                     System.out.print(" ○");
                     break;
                 case Model.BLACK:
                     System.out.print(" ●");
                     break;
                 case Model.SPACE:
                     System.out.print(" ┼");
                     break;
                 }
            }
            System.out.println();
        }
    }
}
```

④Controller.java:

```
package gobang;
```

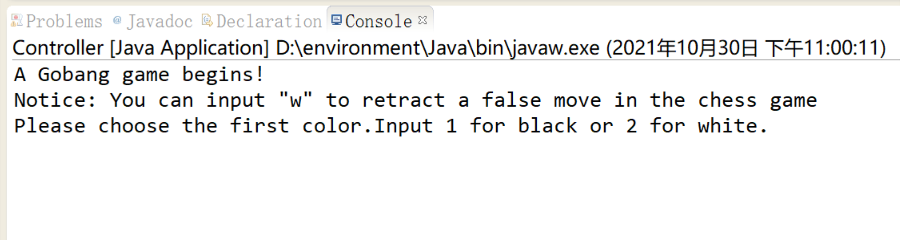
```
public class Controller {

    private int color;
    private static boolean continueGame=true;
    private Controller(){}
    private static Controller instance=null;
    public static Controller getinstance(){
        if(instance==null){
            instance=new Controller();
        }
        return instance;
    }
    public void modifyColor(int color){
        this.color=color;
    }
    public String getColor(){      //获得颜色信息，并以字符串形式返回
        String s=new String();
        switch(color){
        case Model.WHITE:
            s="WHITE";
            break;
        case Model.BLACK:
            s="BLACK";
            break;
        }
        return s;
    }
    public void userPlayChess(int row,int col){          //在棋盘模型中下棋并判断是否
有赢家
        boolean success=Model.getInstance().playChess(row, col, color);
        if(success){
            color=-color;
            int winner=Model.getInstance().whoWin();
            switch(winner){
            case Model.WHITE:
                View.getinstance().outPutInfo("White chess wins!");
                View.getinstance().outPutMap();       //输出下棋后棋盘
                continueGame=false;
                break;
            case Model.BLACK:
                View.getinstance().outPutInfo("Black chess wins!");
                View.getinstance().outPutMap();       //输出下棋后棋盘
                continueGame=false;
                break;
            case Model.SPACE:
                View.getinstance().outPutMap();       //输出下棋后棋盘
                break;

            }
        }
    }
    public void userWithdraw(){
        color=-color;
        Model.getInstance().withdraw();
        View.getinstance().outPutMap();
    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
            View.getinstance().outPutInfo("A Gobang game begins!"+"\n"
        +"Notice: You can input \"w\" to retract a false move in the chess game");
            View.getinstance().inputFirstColor();
            while(continueGame){
                View.getinstance().input();
            }
    }
}
```

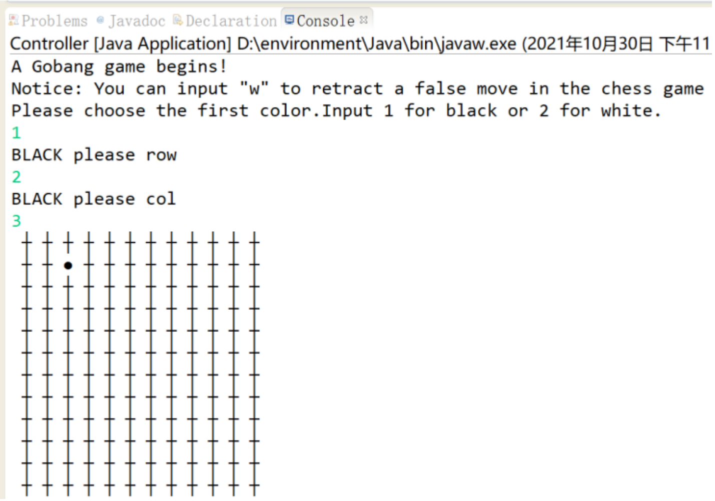
二、　运行试验:

①首先会显示如下信息:



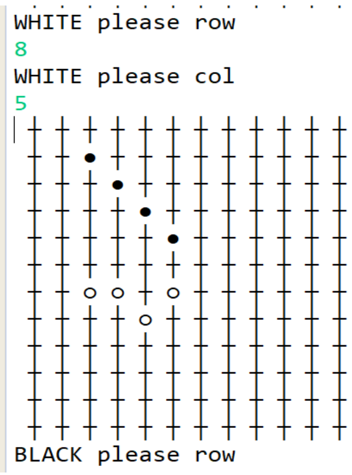
```
Controller [Java Application] D:\environment\Java\bin\javaw.exe (2021年10月30日 下午11:00:11)
A Gobang game begins!
Notice: You can input "w" to retract a false move in the chess game
Please choose the first color.Input 1 for black or 2 for white.
```

②输入1或2选择先手棋子颜色，这里我们选择1，黑棋先下。并在（2，3）下第一颗棋子。如图:



```
Controller [Java Application] D:\environment\Java\bin\javaw.exe (2021年10月30日 下午11
A Gobang game begins!
Notice: You can input "w" to retract a false move in the chess game
Please choose the first color.Input 1 for black or 2 for white.
1
BLACK please row
2
BLACK please col
3
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼●┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
┼┼┼┼┼┼┼┼┼┼┼┼
```

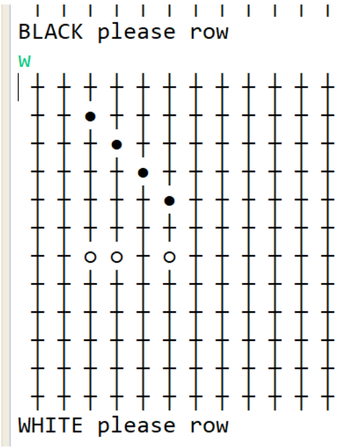
③这样黑白棋交替下几个回合之后，如图：



```
WHITE please row
8
WHITE please col
5
```

```
BLACK please row
```

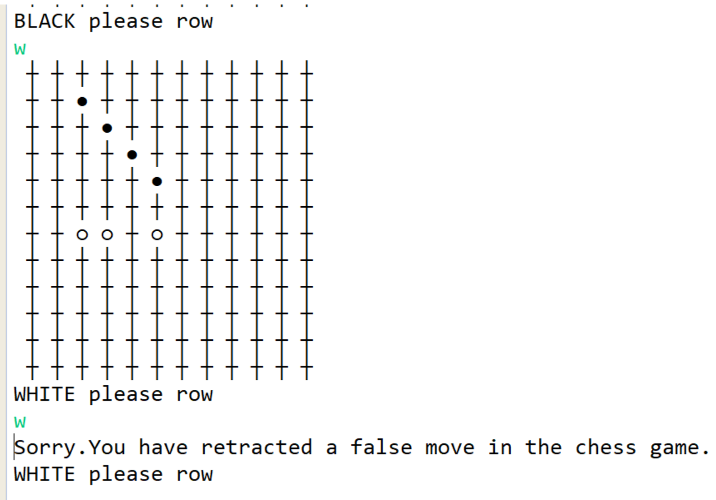
④若白棋想要悔棋，可在下一步输入“w”，如图，会悔棋成功并重新回到白棋的回合：



```
BLACK please row
w
```

```
WHITE please row
```

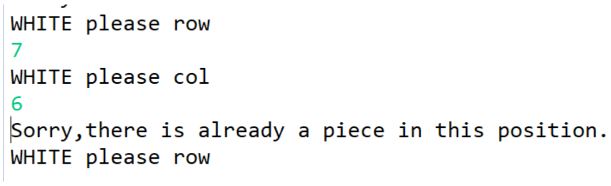
⑤但同一方不可以连续两次悔棋，否则会有错误信息提示：



```
BLACK please row
w
```

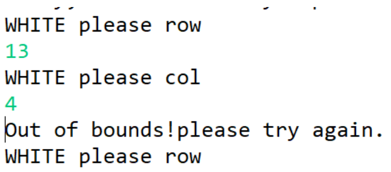
```
WHITE please row
w
Sorry.You have retracted a false move in the chess game.
WHITE please row
```

⑥也不可以在已有棋子的地方下棋，否则不能下棋成功并有错误信息提示：



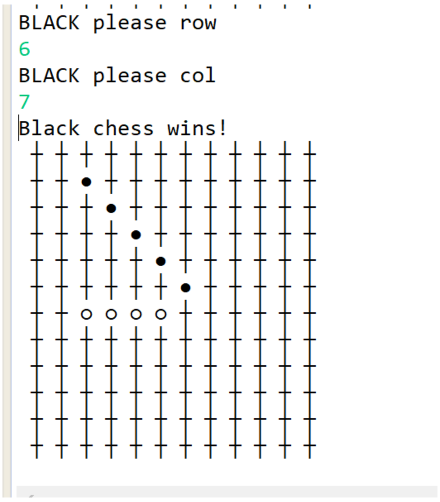
```
WHITE please row
7
WHITE please col
6
Sorry,there is already a piece in this position.
WHITE please row
```

⑦下棋时也不可以超出棋盘界限，否则不能下棋成功并有错误信息提示：

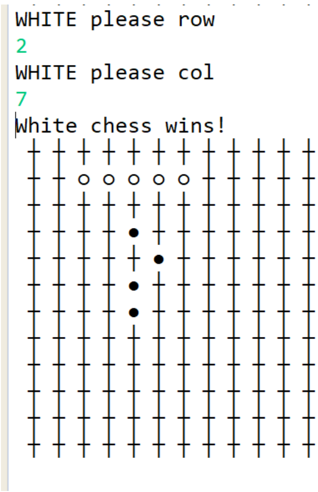


```
WHITE please row
13
WHITE please col
4
Out of bounds!please try again.
WHITE please row
```

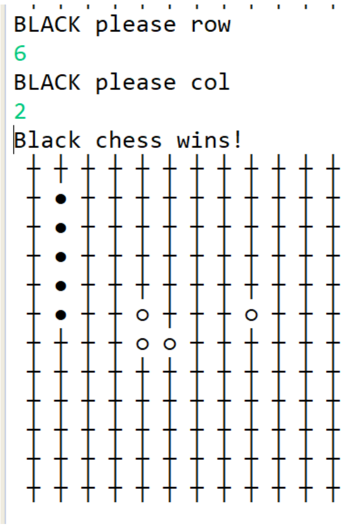
⑧若有一方连成五子，则会输出获胜者信息和最终棋盘，并结束游戏（由布尔值continueGame控制实现），不会再输出“** please row”。获胜情况如下图：



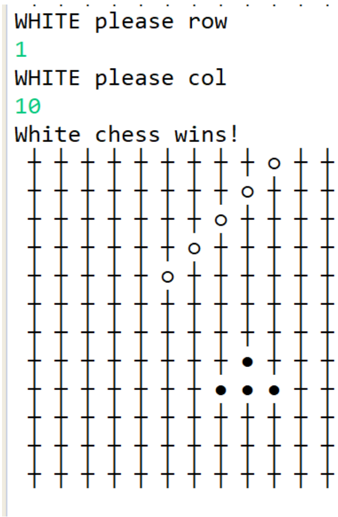
```
BLACK please row
6
BLACK please col
7
Black chess wins!
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼
┼ ┼ ○ ○ ○ ○ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
```



```
WHITE please row
2
WHITE please col
7
White chess wins!
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ○ ○ ○ ○ ○ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
```



```
BLACK please row
6
BLACK please col
2
Black chess wins!
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ● ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ● ┼ ┼ ○ ┼ ┼ ┼ ○ ┼ ┼ ┼
┼ ┼ ┼ ┼ ○ ○ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
```



```
WHITE please row
1
WHITE please col
10
White chess wins!
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ○ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ○ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ○ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ○ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ○ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ● ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ● ● ● ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼ ┼
```