

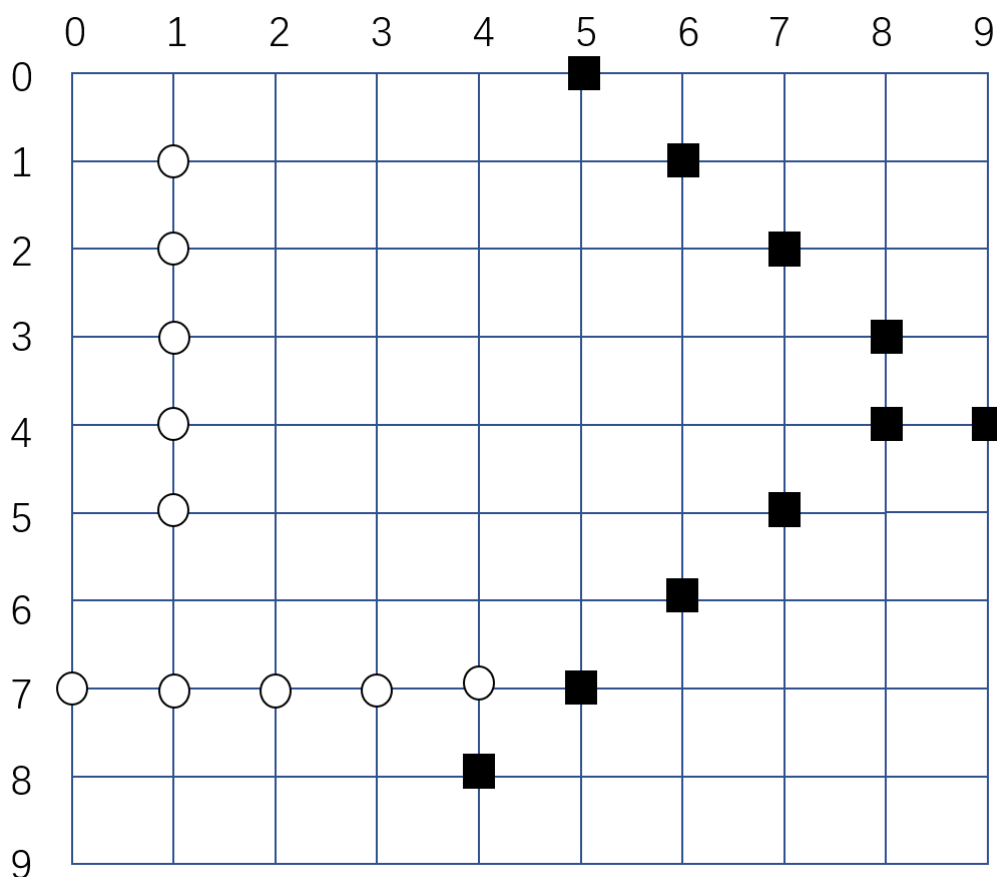
阶段项目--五子棋

目的

串联所学的知识点，从而检测出自己的不足，然后弥补自己的不足

需求说明

五子棋棋盘为一个10 × 10的棋盘，五子棋玩家共2个（这里分别称为A和B），A在棋盘上落子后，B再落子，依次往复，直到一方胜利或者棋盘空间用完为止。判断胜利的条件是一条直线或者任意斜线上同时存在A或者B的连续5颗棋子（见下图）。



技术实现

1. 静态变量

语法

```
1 | public static 数据类型 变量名 = 变量值;
```

解释说明

静态变量只能定义类中，不能定义在方法中。静态变量可以在static修饰的方法中使用，也可以在非静态的方法中访问。主要解决在静态方法中不能访问非静态的变量。

2. 静态方法

语法

```
1 public static 返回值类型 方法名(){
2
3 }
```

解释说明

静态方法就相当于一个箱子，只是这个箱子中装的是代码，需要使用这些代码的时候，就把这个箱子放在指定的位置即可。

示例

```
1 public static void main(String[] args) {
2     show();
3     show();
4     show();
5 }
6
7 public static void show(){
8     System.out.println("张三");
9     System.out.println("男");
10    System.out.println("20");
11 }
```

实现步骤分析

1. 制作棋盘

a. 使用输入法中的制表符在控制台直接打印出棋盘，然后寻找落子位置的特征

```
1 System.out.println("┌───┬───┬───┬───┬───┬───┬───┬───┬───┬───┐");
2 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
3 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
4 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
5 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
6 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
7 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
8 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
9 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
10 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
11 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
12 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
13 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
14 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
15 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
16 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
17 System.out.println("├───┴───┴───┴───┴───┴───┴───┴───┴───┴───┤");
18 System.out.println("│   │   │   │   │   │   │   │   │   │   │");
19 System.out.println("└───┴───┴───┴───┴───┴───┴───┴───┴───┴───┘");
```

b. 利用二维数组重新制作棋盘

```
1 public class Gobang {
```

```

2
3     public static char[][] chessboard = {
4         {'r','t','t','t','t','t','t','t','t','r'},
5         {'t','t','t','t','t','t','t','t','t','t'},
6         {'t','t','t','t','t','t','t','t','t','t'},
7         {'t','t','t','t','t','t','t','t','t','t'},
8         {'t','t','t','t','t','t','t','t','t','t'},
9         {'t','t','t','t','t','t','t','t','t','t'},
10        {'t','t','t','t','t','t','t','t','t','t'},
11        {'t','t','t','t','t','t','t','t','t','t'},
12        {'t','t','t','t','t','t','t','t','t','t'},
13        {'l','l','t','t','t','t','t','t','t','r'}
14    };
15
16    public static String separator = "——";
17
18    public static void main(String[] args) {
19        System.out.println("    0    1    2    3    4    5    6    7
20    8    9");
21        for(int i=0; i<chessboard.length; i++){ //外层循环控制行
22            System.out.print(i + "    ");
23            for(int j=0; j<chessboard[i].length; j++){//内层循环控制列
24                if(j == chessboard[i].length - 1){ //最后一列
25                    System.out.print(chessboard[i][j]);
26                } else {
27                    System.out.print(chessboard[i][j] + separator); //使用
28                    print打印, 不换行
29                }
30            }
31            System.out.println();
32            if(i < chessboard.length - 1){ //排除最后一行
33                System.out.println("    |    |    |    |    |    |
34    |    |    |    |");
35            }
36        }
37    }
38 }

```

c. 棋盘在玩家使用过程中会反复展示，需要使用方法来进行优化

```

1     public class Gobang {
2
3         public static char[][] chessboard = {
4             {'r','t','t','t','t','t','t','t','t','r'},
5             {'t','t','t','t','t','t','t','t','t','t'},
6             {'t','t','t','t','t','t','t','t','t','t'},
7             {'t','t','t','t','t','t','t','t','t','t'},
8             {'t','t','t','t','t','t','t','t','t','t'},
9             {'t','t','t','t','t','t','t','t','t','t'},
10            {'t','t','t','t','t','t','t','t','t','t'},
11            {'t','t','t','t','t','t','t','t','t','t'},
12            {'t','t','t','t','t','t','t','t','t','t'},
13            {'l','l','t','t','t','t','t','t','t','r'}
14        };
15
16        public static String separator = "——";
17

```

```

18     public static void main(String[] args) {
19         showChessboard();
20     }
21
22     public static void showChessboard(){
23         System.out.println("    0    1    2    3    4    5    6    7
24         8    9");
25         for(int i=0; i<chessboard.length; i++){ //外层循环控制行
26             System.out.print(i + "    ");
27             for(int j=0; j<chessboard[i].length; j++){//内层循环控制列
28                 if(j == chessboard[i].length - 1){ //最后一列
29                     System.out.print(chessboard[i][j]);
30                 } else {
31                     System.out.print(chessboard[i][j] + separator);//使用
32                     print打印, 不换行
33                 }
34             }
35             System.out.println();
36             if(i < chessboard.length -1){ //排除最后一行
37                 System.out.println("    |    |    |    |    |    |    |
38                 |    |    |    |    |    |    |");
39             }
40         }
41     }

```

2. 落子

a. 玩家A、B会交替落子

```

1  /**
2   * 五子棋
3   */
4  public class Gobang {
5
6      public static char[][] chessboard = {
7          {'r','t','t','t','t','t','t','t','t','t'},
8          {'t','t','t','t','t','t','t','t','t','t'},
9          {'t','t','t','t','t','t','t','t','t','t'},
10         {'t','t','t','t','t','t','t','t','t','t'},
11         {'t','t','t','t','t','t','t','t','t','t'},
12         {'t','t','t','t','t','t','t','t','t','t'},
13         {'t','t','t','t','t','t','t','t','t','t'},
14         {'t','t','t','t','t','t','t','t','t','t'},
15         {'t','t','t','t','t','t','t','t','t','t'},
16         {'l','l','l','l','l','l','l','l','l','l'}
17     };
18
19     public static String separator = "——";
20
21     public static void main(String[] args) {
22         showChessboard();
23         int totalPosition = chessboard.length * chessboard[0].length;
24         for(int times=0; times < totalPosition; times++){

```

```

25         System.out.println(times % 2 == 0 ? "请玩家A落子: " : "请玩家B落
子");
26     }
27 }
28
29 public static void showChessboard(){
30     System.out.println("    0    1    2    3    4    5    6    7
8    9");
31     for(int i=0; i<chessboard.length; i++){ //外层循环控制行
32         System.out.print(i + "    ");
33         for(int j=0; j<chessboard[i].length; j++){//内层循环控制列
34             if(j == chessboard[i].length - 1){ //最后一列
35                 System.out.print(chessboard[i][j]);
36             } else {
37                 System.out.print(chessboard[i][j] + separator);//使用
print打印, 不换行
38             }
39         }
40         System.out.println();
41         if(i < chessboard.length - 1){ //排除最后一行
42             System.out.println("    |    |    |    |    |    |
|    |    |    |");
43         }
44     }
45 }
46 }
47 }

```

b. 落子位置必须是0~100之间的整数, 且不能使用已经存在棋子的位置

```

1  import java.util.Scanner;
2
3  /**
4   * 五子棋
5   */
6  public class Gobang {
7
8      public static char[][] chessboard = {
9          {'r','t','t','t','t','t','t','t','t','t'},
10         {'t','t','t','t','t','t','t','t','t','t'},
11         {'t','t','t','t','t','t','t','t','t','t'},
12         {'t','t','t','t','t','t','t','t','t','t'},
13         {'t','t','t','t','t','t','t','t','t','t'},
14         {'t','t','t','t','t','t','t','t','t','t'},
15         {'t','t','t','t','t','t','t','t','t','t'},
16         {'t','t','t','t','t','t','t','t','t','t'},
17         {'t','t','t','t','t','t','t','t','t','t'},
18         {'l','l','l','l','l','l','l','l','l','l'}
19     };
20
21     public static String separator = "——";
22
23     public static char pieceA = 'o'; //玩家A的棋子
24     public static char pieceB = '■'; //玩家B的棋子
25
26     public static void main(String[] args) {
27         showChessboard();

```

```

28     int totalPosition = chessboard.length * chessboard[0].length;
29     Scanner sc = new Scanner(System.in);
30     for(int times=0; times < totalPosition; times++){
31         System.out.println(times % 2 == 0 ? "请玩家A落子: " : "请玩家B落
子: ");
32         while(true){//保证落子成功的一个循环
33             //检测Scanner中是否有输入的数据并且判断数据是否为整数，如果没有数据，
就需要从控制台输入
34             if(sc.hasNextInt()){
35                 int position = sc.nextInt();
36                 if(position >= 0 && position < totalPosition){
37                     char currentPiece = (times % 2 == 0) ? pieceA :
pieceB;
38                     // position = 21    13x13        21 / 13 = 1    21 % 13
= 8
39                     int row = position / chessboard.length; //位置除以棋
盘数组的长度得到行号
40                     int col = position % chessboard[0].length; //位置取模
棋盘数组的总列数得到列号
41                     if(chessboard[row][col] == pieceA ||
chessboard[row][col] == pieceB){
42                         System.out.println("非法落子，请重新选择落子位置");
43                         continue;
44                     } else {
45                         chessboard[row][col] = currentPiece;
46                         break;
47                     }
48                 } else {
49                     System.out.println("非法落子，请重新选择落子位置");
50                 }
51             } else {
52                 System.out.println("非法落子，请重新选择落子位置");
53                 sc.next();//将Scanner中存储的数据取出来，防止死循环
54             }
55         }
56         //落子完成后，棋盘需要重新展示
57         showChessboard();
58     }
59 }
60
61 public static void showChessboard(){
62     System.out.println("    0    1    2    3    4    5    6    7
8    9");
63     for(int i=0; i<chessboard.length; i++){ //外层循环控制行
64         System.out.print(i + "    ");
65         for(int j=0; j<chessboard[i].length; j++){//内层循环控制列
66             if(j == chessboard[i].length - 1){ //最后一列
67                 System.out.print(chessboard[i][j]);
68             } else {
69                 System.out.print(chessboard[i][j] + separator);//使用
print打印，不换行
70             }
71         }
72         System.out.println();
73         if(i < chessboard.length -1){//排除最后一行
74             System.out.println("    |    |    |    |    |    |
|    |    |    |");
75         }

```

```

76     }
77
78     }
79 }

```

c. 落子完成后，需要校验是否获胜

```

1  import java.util.Scanner;
2
3  /**
4   * 五子棋
5   */
6  public class Gobang {
7
8      public static char[][] chessboard = {
9          {'r','t','t','t','t','t','t','t','t','t','t'},
10         {'t','t','t','t','t','t','t','t','t','t','t'},
11         {'t','t','t','t','t','t','t','t','t','t','t'},
12         {'t','t','t','t','t','t','t','t','t','t','t'},
13         {'t','t','t','t','t','t','t','t','t','t','t'},
14         {'t','t','t','t','t','t','t','t','t','t','t'},
15         {'t','t','t','t','t','t','t','t','t','t','t'},
16         {'t','t','t','t','t','t','t','t','t','t','t'},
17         {'t','t','t','t','t','t','t','t','t','t','t'},
18         {'t','t','t','t','t','t','t','t','t','t','t'}
19     };
20
21     public static String separator = "——";
22
23     public static char pieceA = 'o'; //玩家A的棋子
24     public static char pieceB = '■'; //玩家B的棋子
25
26     public static void main(String[] args) {
27         showChessboard();
28         int totalPosition = chessboard.length * chessboard[0].length;
29         Scanner sc = new Scanner(System.in);
30         outer:
31         for(int times=0; times < totalPosition; times++){
32             System.out.println(times % 2 == 0 ? "请玩家A落子: " : "请玩家B落
33             子: ");
34             char currentPiece = (times % 2 == 0) ? pieceA : pieceB; //当前使
35             用的棋子
36             while(true){ //保证落子成功的一个循环
37                 //检测Scanner中是否有输入的数据并且判断数据是否为整数，如果没有数
38                 据，就需要从控制台输入
39                 if(sc.hasNextInt()){
40                     int position = sc.nextInt();
41                     if(position >= 0 && position < totalPosition){
42                         // position = 21    13x13    21 / 13 = 1    21 % 13
43                         = 8
44                         int row = position / chessboard.length; //位置除以棋
45                         盘数组的长度得到行号
46                         int col = position % chessboard[0].length; //位置取
47                         模棋盘数组的总列数得到列号
48                         if(chessboard[row][col] == pieceA ||
49                         chessboard[row][col] == pieceB){
50                             System.out.println("非法落子，请重新选择落子位置");

```

```

44         continue;
45     } else {
46         chessboard[row][col] = currentPiece;
47         break;
48     }
49 } else {
50     System.out.println("非法落子，请重新选择落子位置");
51 }
52 } else {
53     System.out.println("非法落子，请重新选择落子位置");
54     sc.next(); //将Scanner中存储的数据取出来，防止死循环
55 }
56 }
57 //落子完成后，棋盘需要重新展示
58 showChessboard();
59 //判断获胜情况，一共4种
60 for(int i=0; i<chessboard.length; i++){
61     for(int j=0; j<chessboard[i].length; j++){
62         //第一种：水平方向上存在同一玩家的连续5颗棋子
63         //(i, j) (i, j+1) (i, j+2) (i, j+3) (i, j+4)
64         boolean case1 = (j + 4 < chessboard[i].length)
65             && chessboard[i][j] == currentPiece
66             && chessboard[i][j+1] == currentPiece
67             && chessboard[i][j+2] == currentPiece
68             && chessboard[i][j+3] == currentPiece
69             && chessboard[i][j+4] == currentPiece;
70         //第二种：锤子方向上存在同一玩家的连续5颗棋子
71         //(i, j) (i+1, j) (i+2, j) (i+3, j) (i+4, j)
72         boolean case2 = (i + 4 < chessboard.length)
73             && chessboard[i][j] == currentPiece
74             && chessboard[i+1][j] == currentPiece
75             && chessboard[i+2][j] == currentPiece
76             && chessboard[i+3][j] == currentPiece
77             && chessboard[i+4][j] == currentPiece;
78         //第三种：135°角上存在同一玩家的连续5颗棋子
79         //(i, j) (i+1, j+1) (i+2, j+2) (i+3, j+3) (i+4, j+4)
80         boolean case3 = (i + 4 < chessboard.length)
81             && (j + 4 < chessboard[i].length)
82             && chessboard[i][j] == currentPiece
83             && chessboard[i+1][j+1] == currentPiece
84             && chessboard[i+2][j+2] == currentPiece
85             && chessboard[i+3][j+3] == currentPiece
86             && chessboard[i+4][j+4] == currentPiece;
87         //第四种：45°角上存在同一玩家的连续5颗棋子
88         //(i, j) (i-1, j+1) (i-2, j+2) (i-3, j+3) (i-4, j+4)
89         boolean case4 = (i > 4) && (j + 4 <
chessboard[i].length)
90             && chessboard[i][j] == currentPiece
91             && chessboard[i-1][j+1] == currentPiece
92             && chessboard[i-2][j+2] == currentPiece
93             && chessboard[i-3][j+3] == currentPiece
94             && chessboard[i-4][j+4] == currentPiece;
95         if(case1 || case2 || case3 || case4){
96             System.out.println(times % 2 == 0 ? "玩家A获得胜利"
: "玩家B获得胜利");
97             break outer;
98         }
99     }

```



```

100     }
101     }
102 }
103
104 public static void showChessboard(){
105     system.out.println("  0    1    2    3    4    5    6    7
8    9");
106     for(int i=0; i<chessboard.length; i++){ //外层循环控制行
107         System.out.print(i + " ");
108         for(int j=0; j<chessboard[i].length; j++){//内层循环控制列
109             if(j == chessboard[i].length - 1){ //最后一列
110                 System.out.print(chessboard[i][j]);
111             } else {
112                 System.out.print(chessboard[i][j] + separator);//使用
113                 print打印, 不换行
114             }
115         }
116         System.out.println();
117         if(i < chessboard.length - 1){//排除最后一行
118             System.out.println(" | | | | | | |");
119         }
120     }
121 }
122 }

```

d. 棋盘使用完毕还未分出胜负，需要提示

```

1  import java.util.Scanner;
2
3  /**
4   * 五子棋
5   */
6  public class Gobang {
7
8      public static char[][] chessboard = {
9          {'r','t','t','t','t','t','t','t','t','t','r'},
10         {'t','t','t','t','t','t','t','t','t','t','t'},
11         {'t','t','t','t','t','t','t','t','t','t','t'},
12         {'t','t','t','t','t','t','t','t','t','t','t'},
13         {'t','t','t','t','t','t','t','t','t','t','t'},
14         {'t','t','t','t','t','t','t','t','t','t','t'},
15         {'t','t','t','t','t','t','t','t','t','t','t'},
16         {'t','t','t','t','t','t','t','t','t','t','t'},
17         {'t','t','t','t','t','t','t','t','t','t','t'},
18         {'t','t','t','t','t','t','t','t','t','t','t'}
19     };
20
21     public static String separator = "——";
22
23     public static char pieceA = 'o'; //玩家A的棋子
24
25     public static char pieceB = '■';//玩家B的棋子
26
27     public static int times = 0; //记录棋盘使用次数，偶数次玩家A落子，奇数次玩家B
落子

```

```

28
29     public static void main(String[] args) {
30         showChessboard();
31         int totalPosition = chessboard.length * chessboard[0].length;
32         Scanner sc = new Scanner(System.in);
33         outer:
34         while (times < totalPosition){
35             System.out.println(times % 2 == 0 ? "请玩家A落子: " : "请玩家B落
子: ");
36             char currentPiece = (times % 2 == 0) ? pieceA : pieceB;//当前使
用的棋子
37             while(true){//保证落子成功的一个循环
38                 //检测Scanner中是否有输入的数据并且判断数据是否为整数，如果没有数
据，就需要从控制台输入
39                 if(sc.hasNextInt()){
40                     int position = sc.nextInt();
41                     if(position >= 0 && position < totalPosition){
42                         // position = 21    13x13        21 / 13 = 1    21 % 13
= 8
43                         int row = position / chessboard.length; //位置除以棋
盘数组的长度得到行号
44                         int col = position % chessboard[0].length; //位置取
模棋盘数组的总列数得到列号
45                         if(chessboard[row][col] == pieceA ||
chessboard[row][col] == pieceB){
46                             System.out.println("非法落子，请重新选择落子位置");
47                             continue;
48                         } else {
49                             chessboard[row][col] = currentPiece;
50                             break;
51                         }
52                     } else {
53                         System.out.println("非法落子，请重新选择落子位置");
54                     }
55                 } else {
56                     System.out.println("非法落子，请重新选择落子位置");
57                     sc.next();//将Scanner中存储的数据取出来，防止死循环
58                 }
59             }
60             //落子完成后，棋盘需要重新展示
61             showChessboard();
62             //判断获胜情况，一共4种
63             for(int i=0; i<chessboard.length; i++){
64                 for(int j=0; j<chessboard[i].length; j++){
65                     //第一种：水平方向上存在同一玩家的连续5颗棋子
66                     //(i, j) (i, j+1) (i, j+2) (i, j+3) (i, j+4)
67                     boolean case1 = (j + 4 < chessboard[i].length)
68                         && chessboard[i][j] == currentPiece
69                         && chessboard[i][j+1] == currentPiece
70                         && chessboard[i][j+2] == currentPiece
71                         && chessboard[i][j+3] == currentPiece
72                         && chessboard[i][j+4] == currentPiece;
73                     //第二种：锤子方向上存在同一玩家的连续5颗棋子
74                     //(i, j) (i+1, j) (i+2, j) (i+3, j) (i+4, j)
75                     boolean case2 = (i + 4 < chessboard.length)
76                         && chessboard[i][j] == currentPiece
77                         && chessboard[i+1][j] == currentPiece
78                         && chessboard[i+2][j] == currentPiece

```

```

79         && chessboard[i+3][j] == currentPiece
80         && chessboard[i+4][j] == currentPiece;
81         //第三种: 135°角上存在同一玩家的连续5颗棋子
82         //(i, j) (i+1, j+1) (i+2, j+2) (i+3, j+3) (i+4, j+4)
83         boolean case3 = (i + 4 < chessboard.length)
84         && (j + 4 < chessboard[i].length)
85         && chessboard[i][j] == currentPiece
86         && chessboard[i+1][j+1] == currentPiece
87         && chessboard[i+2][j+2] == currentPiece
88         && chessboard[i+3][j+3] == currentPiece
89         && chessboard[i+4][j+4] == currentPiece;
90         //第四种: 45°角上存在同一玩家的连续5颗棋子
91         //(i, j) (i-1, j+1) (i-2, j+2) (i-3, j+3) (i-4, j+4)
92         boolean case4 = (i > 4) && (j + 4 <
chessboard[i].length)
93         && chessboard[i][j] == currentPiece
94         && chessboard[i-1][j+1] == currentPiece
95         && chessboard[i-2][j+2] == currentPiece
96         && chessboard[i-3][j+3] == currentPiece
97         && chessboard[i-4][j+4] == currentPiece;
98         if(case1 || case2 || case3 || case4){
99             System.out.println(times % 2 == 0 ? "玩家A获得胜利"
: "玩家B获得胜利");
100             break outer;
101         }
102     }
103 }
104 times++;
105 }
106 if(times == totalPosition){//说明棋盘已经用完还未分出胜负
107     System.out.println("平局");
108 }
109 }
110
111 public static void showChessboard(){
112     System.out.println("  0    1    2    3    4    5    6    7
8    9");
113     for(int i=0; i<chessboard.length; i++){ //外层循环控制行
114         System.out.print(i + " ");
115         for(int j=0; j<chessboard[i].length; j++){//内层循环控制列
116             if(j == chessboard[i].length - 1){ //最后一列
117                 System.out.print(chessboard[i][j]);
118             } else {
119                 System.out.print(chessboard[i][j] + separator);//使用
print打印, 不换行
120             }
121         }
122         System.out.println();
123         if(i < chessboard.length - 1){//排除最后一行
124             System.out.println("    |    |    |    |    |    |
|    |    |    |");
125         }
126     }
127 }
128 }
129 }

```

3. 声音特效

为落子、非法落子及获胜添加音效

```
1  import java.applet.Applet;
2  import java.applet.AudioClip;
3  import java.net.URL;
4  import java.util.Scanner;
5
6  /**
7   * 五子棋
8   */
9  public class Gobang {
10
11     public static char[][] chessboard = {
12         {'r','t','t','t','t','t','t','t','t','t','t'},
13         {'t','t','t','t','t','t','t','t','t','t','t'},
14         {'t','t','t','t','t','t','t','t','t','t','t'},
15         {'t','t','t','t','t','t','t','t','t','t','t'},
16         {'t','t','t','t','t','t','t','t','t','t','t'},
17         {'t','t','t','t','t','t','t','t','t','t','t'},
18         {'t','t','t','t','t','t','t','t','t','t','t'},
19         {'t','t','t','t','t','t','t','t','t','t','t'},
20         {'t','t','t','t','t','t','t','t','t','t','t'},
21         {'t','t','t','t','t','t','t','t','t','t','t'}
22     };
23
24     public static String separator = "———";
25
26     public static char pieceA = 'o'; //玩家A的棋子
27
28     public static char pieceB = '■'; //玩家B的棋子
29
30     public static int times = 0; //记录棋盘使用次数，偶数次玩家A落子，奇数次玩家B
落子
31
32     public static void main(String[] args) {
33         showChessboard();
34         int totalPosition = chessboard.length * chessboard[0].length;
35         Scanner sc = new Scanner(System.in);
36         outer:
37         while (times < totalPosition){
38             System.out.println(times % 2 == 0 ? "请玩家A落子: " : "请玩家B落
子: ");
39             char currentPiece = (times % 2 == 0) ? pieceA : pieceB; //当前使
用的棋子
40             while(true){ //保证落子成功的一个循环
41                 //检测Scanner中是否有输入的数据并且判断数据是否为整数，如果没有数
据，就需要从控制台输入
42                 if(sc.hasNextInt()){
43                     int position = sc.nextInt();
44                     if(position >= 0 && position < totalPosition){
45                         // position = 21    13x13    21 / 13 = 1    21 % 13
= 8
46                         int row = position / chessboard.length; //位置除以棋
盘数组的长度得到行号
```

```

47         int col = position % chessboard[0].length; //位置取
模棋盘数组的总列数得到列号
48         if(chessboard[row][col] == pieceA ||
chessboard[row][col] == pieceB){
49             System.out.println("非法落子，请重新选择落子位置");
50             playAudio("illegal.wav");
51             continue;
52         } else {
53             chessboard[row][col] = currentPiece; //落子
54             playAudio("fill.wav");
55             break;
56         }
57     } else {
58         System.out.println("非法落子，请重新选择落子位置");
59         playAudio("illegal.wav");
60     }
61 } else {
62     System.out.println("非法落子，请重新选择落子位置");
63     playAudio("illegal.wav");
64     sc.next(); //将Scanner中存储的数据取出来，防止死循环
65 }
66 }
67 //落子完成后，棋盘需要重新展示
68 showChessboard();
69 //判断获胜情况，一共4种
70 for(int i=0; i<chessboard.length; i++){
71     for(int j=0; j<chessboard[i].length; j++){
72         //第一种：水平方向上存在同一玩家的连续5颗棋子
73         //(i, j) (i, j+1) (i, j+2) (i, j+3) (i, j+4)
74         boolean case1 = (j + 4 < chessboard[i].length)
75             && chessboard[i][j] == currentPiece
76             && chessboard[i][j+1] == currentPiece
77             && chessboard[i][j+2] == currentPiece
78             && chessboard[i][j+3] == currentPiece
79             && chessboard[i][j+4] == currentPiece;
80         //第二种：锤子方向上存在同一玩家的连续5颗棋子
81         //(i, j) (i+1, j) (i+2, j) (i+3, j) (i+4, j)
82         boolean case2 = (i + 4 < chessboard.length)
83             && chessboard[i][j] == currentPiece
84             && chessboard[i+1][j] == currentPiece
85             && chessboard[i+2][j] == currentPiece
86             && chessboard[i+3][j] == currentPiece
87             && chessboard[i+4][j] == currentPiece;
88         //第三种：135°角上存在同一玩家的连续5颗棋子
89         //(i, j) (i+1, j+1) (i+2, j+2) (i+3, j+3) (i+4, j+4)
90         boolean case3 = (i + 4 < chessboard.length)
91             && (j + 4 < chessboard[i].length)
92             && chessboard[i][j] == currentPiece
93             && chessboard[i+1][j+1] == currentPiece
94             && chessboard[i+2][j+2] == currentPiece
95             && chessboard[i+3][j+3] == currentPiece
96             && chessboard[i+4][j+4] == currentPiece;
97         //第四种：45°角上存在同一玩家的连续5颗棋子
98         //(i, j) (i-1, j+1) (i-2, j+2) (i-3, j+3) (i-4, j+4)
99         boolean case4 = (i > 4) && (j + 4 <
chessboard[i].length)
100             && chessboard[i][j] == currentPiece
101             && chessboard[i-1][j+1] == currentPiece

```

```

102         && chessboard[i-2][j+2] == currentPiece
103         && chessboard[i-3][j+3] == currentPiece
104         && chessboard[i-4][j+4] == currentPiece;
105         if(case1 || case2 || case3 || case4){
106             System.out.println(times % 2 == 0 ? "玩家A获得胜利"
: "玩家B获得胜利");
107             playAudio("win.wav");
108             break outer;
109         }
110     }
111 }
112 times++;
113 }
114 if(times == totalPosition){//说明棋盘已经用完还未分出胜负
115     System.out.println("平局");
116 }
117 }
118
119 public static void showChessboard(){
120     System.out.println("    0    1    2    3    4    5    6    7
8    9");
121     for(int i=0; i<chessboard.length; i++){ //外层循环控制行
122         System.out.print(i + "    ");
123         for(int j=0; j<chessboard[i].length; j++){//内层循环控制列
124             if(j == chessboard[i].length - 1){ //最后一列
125                 System.out.print(chessboard[i][j]);
126             } else {
127                 System.out.print(chessboard[i][j] + separator);//使用
print打印, 不换行
128             }
129         }
130         System.out.println();
131         if(i < chessboard.length -1){//排除最后一行
132             System.out.println("    |    |    |    |    |    |
|    |    |    |");
133         }
134     }
135 }
136
137 /**
138  * 播放声音
139  * @param fileName
140  */
141 public static void playAudio(String fileName){
142     URL url = Gobang.class.getResource(fileName);
143     AudioClip clip = Applet.newAudioClip(url);
144     clip.play();
145     try {
146         Thread.sleep(50);
147     } catch (InterruptedException e) {
148     }
149 }
150 }

```

