

1 文件结构

本 lab 由 7 个文件构成，Lab5.java 中存放初始化、游戏主体及结束语句；Boarddata.java 中存放棋盘数据类相关方法；Board.java 中存放单一棋盘上的操作；Color.java 中存放棋盘用枚举类型；Player.java 中存放用户类和相关方法；Rule.java 中存放规则枚举类型；Direction.java 中存放方向枚举类型。

2 文件内容说明

2.1 Lab5.java

Line:8~Line:17 初始化游戏信息，输入双方玩家名、清屏打印空地图。

Line:20~Line:24 游戏主体循环，包含游戏逻辑、清屏、打印地图。

Line:27~Line:29 游戏结束，关闭输入缓冲区，打印结束语。

2.2 Boarddata.java

游戏棋盘相关方法。

2.2.1 存放数据

height 棋盘的高。

width 棋盘的宽。

boardNum 棋盘数量。

currentBoard 当前棋盘编号（从 1 开始）。

ifQuit 是否退出游戏。

boardDrift 棋盘列表打印偏移量。

playerBlack 黑方姓名。

playerWhite 白方姓名。

board 棋盘数据。

2.2.2 构造方法

有参数构造方法中，参数输入棋盘高、宽，棋盘数量，黑方姓名，白方姓名。将相关变量赋初值，计算棋盘列表打印偏移量为玩家姓名长度最大值加 40，创建棋盘 ArrayList，通过 ArrayList 的 add 方法创建棋盘。

2.2.3 boardGame

游戏主体方法，通过 `input` 参数接收输入缓冲区。`ifCorrect` 用于记录用户输入的合法性。通过 `Ascii` 转义字符将控制台光标定位到最后一行。当棋盘结束标识符为 1 时，打印结算界面。`for` 循环判断当前棋盘落子玩家并根据当前棋盘规则打印提示信息。`while` 循环接收玩家输入内容，当输入值为纯数字时判断为更换棋盘，判断是否越界；当输入值长度为 2 且游戏未结束时，进入落子逻辑；当输入值为 `peace` 或 `reversi` 或 `gomoku` 时，添加新棋盘；当输入值为 `pass` 时，判断是否允许放弃行棋；当输入值为 `quit` 时，退出游戏；否则提示错误信息，重新接收输入。

2.3 boardListPrint

棋盘列表打印方法，先将控制台光标定位至第一行合适位置，然后打印对应内容，每打印 8 个棋盘名另起一列继续打印。最后将控制台光标定位到末尾。

2.4 boardDataPrint

棋盘打印入口。

2.5 Board.java

2.5.1 存放数据

`height` 地图的高。

`width` 地图的宽。

`emptyRemain` 地图上剩余空位。

`blackCount` 棋盘上黑子个数。

`whiteCount` 棋盘上白子个数。

`boardCount` 当前棋盘编号

`ifFinished` 结束标识符

`canPass` 双方是否允许跳过。

`currentColor` 当前玩家颜色。

`rule` 当前棋盘规则。

`playerDrift` 玩家列表打印偏移量。

`boardDrift` 棋盘列表打印偏移量。

`winner` 记录胜方

player 玩家数据。

boardColor 枚举类型棋盘。

boardString 字符类型棋盘。

2.5.2 构造类型

有参数构造方法接收地图高、宽，当前棋盘编号，棋盘规则，黑方姓名，白方姓名。逐一计算，赋值，通过循环创建空白棋盘，对特殊位置进行赋值，然后使用 `colorToChar` 方法完成字符棋盘赋值，最后添加行列索引。

2.5.3 colorToChar

将枚举类型棋盘逐一转换为字符类型棋盘正确位置。

2.5.4 printBoard

通过循环打印当前棋盘，通过 `playerPrint` 和 `boardListPrint` 方法打印玩家列表和棋盘列表。

2.5.5 boardUpdate

游戏下子逻辑方法，通过 `move` 参数接收玩家输入。将输入的两位分别通过内码转换成棋盘上行列，然后通过分支判断输入是否合法（是否在棋盘上，是否该位置已有子，`reversi` 规则下是否在合法位置），输入合法的情况下将枚举类型棋盘对应位置赋对应值，进入 `reversi` 规则翻转逻辑，进入五子棋判断，棋盘空格减一，若空位为一，改变游戏结束标识符为 1，游戏轮次加 1，更新字符类型棋盘，棋子计数，翻转当前玩家颜色，判断新玩家合法落子位置。

2.5.6 playerPrint

玩家列表打印方法，通过 `Ascii` 转义字符将控制台光标跳转至第 4 行合适位置，打印对应内容，通过分支语句判断是否需要打印玩家棋子、棋子个数、游戏轮次。

2.5.7 ifPlaceable

`reversi` 规则下合法落子位置判断方法。若为 `peace` 规则，直接跳出。默认无合法落子点，将对应玩家 `canPass` 赋 1，遍历棋盘获取空白位置，遍历该位置的八个方向，若该方向离开棋盘或不是对方棋子则直接跳过，然后根据各方向步进判断是否是连续的对方棋子，且由己方棋子结束，如果是则该空白位置是合法落子位置，在字符类型棋盘上赋 +，且不允许放弃行棋。

2.5.8 reversiConvert

`reversi` 规则下棋盘翻转方法。若为 `peace` 规则，直接跳出。与 `ifPlaceable` 方法类似，判断是否连续对方棋子被己方棋子所夹，若是，则将对方棋子翻转，反之则进入下一个枚举值。

2.5.9 printResult

游戏结算界面方法，打印提示语后若为 `peace` 规则直接跳出，若为 `reversi` 规则，打印双方得分和游戏结果。

2.5.10 count

棋盘计数方法，通过遍历计算棋盘上黑子和白子的个数。

2.5.11 checkGomoku

五子棋检查方法，接收玩家落子位置。若非 gomoku 规则，直接跳出。通过循环遍历四个方向，向两头步进，若为己方则计数器加 1，最后判断计数器数值，大于等于 5 则获胜。

2.6 Color.java

枚举类型，三个枚举值对应有黑子、有白子和空状态。

2.6.1 toString

通过 switch 分支获得枚举类型对应的字符。

2.6.2 convert

翻转当前颜色，若为空则不变。

2.7 Player.java

2.7.1 存放数据

name 玩家姓名。

color 玩家颜色。

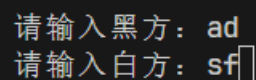
2.8 Rule.java

枚举类型，分别为 peace 规则、reversi 规则和 gomoku 规则。

2.9 Direction.java

枚举类型，存放棋盘上八个方向，以及对应方向的行列步进值。

3 运行截图



```
请输入黑方: ad
请输入白方: sf
```

图 1: 玩家姓名输入

```

  A B C D E F G H
1 . . . . .
2 . . . . .
3 . . . . .
4 . . . ● ○ . .
5 . . . ○ ● . .
6 . . . . .
7 . . . . .
8 . . . . .
棋盘1
玩家[ad]○
玩家[sf]

Game List
1、peace
2、reversi
3、gomoku

请玩家[ad]输入落子位置 (1A) /棋盘编号 (1-3) /新游戏类型 (peace或reversi或gomoku) /退出游戏 (quit)
```

图 2: 初始 peace 棋盘

```

  A B C D E F G H
1 . . . . .
2 . . . . .
3 . . . + . . .
4 . . + ● ○ . .
5 . . . ○ ● + .
6 . . . + . . .
7 . . . . .
8 . . . . .
棋盘2
玩家[ad]○ 2
玩家[sf] 2

Game List
1、peace
2、reversi
3、gomoku

请玩家[ad]输入落子位置 (1A) /棋盘编号 (1-3) /新游戏类型 (peace或reversi或gomoku) /放弃行棋 (pass) /退出游戏 (quit)
```

图 3: 初始 reversi 棋盘

```

  A B C D E F G H
1 . . . . .
2 . . . . .
3 . . . . .
4 . . . . .
5 . . . . .
6 . . . . .
7 . . . . .
8 . . . . .
棋盘3
玩家[ad]○
玩家[sf]
Current round: 0

Game List
1、peace
2、reversi
3、gomoku

请玩家[ad]输入落子位置 (1A) /棋盘编号 (1-3) /新游戏类型 (peace或reversi或gomoku) /退出游戏 (quit)
```

图 4: 初始 gomoku 棋盘

```

  A B C D E F G H
1 . . . . .
2 . . . . .
3 . . . . .
4 . . . . .
5 . . . . .
6 . . . . .
7 . . . . .
8 . . . . .
棋盘4
玩家[ad]○
玩家[sf]
Current round: 0

Game List
1、peace
2、reversi
3、gomoku
4、gomoku

请玩家[ad]输入落子位置 (1A) /棋盘编号 (1-4) /新游戏类型 (peace或reversi或gomoku) /退出游戏 (quit)
```

图 5: 创建新棋盘

```

  A B C D E F G H
1 . . . . .
2 . . . . .
3 . . . . .
4 . . . . .
5 . . . . .
6 . . . . .
7 . . . . .
8 . . . . .
棋盘4
玩家[ad]○
玩家[sf]
Current round: 0
Game List
1、peace
2、reversi
3、gomoku
4、gomoku
请玩家[ad]输入落子位置（1A）/棋盘编号（1-4）/新游戏类型（peace或reversi或gomoku）/退出游戏（quit）
6
不存在该棋盘，请重新输入！
1q
落子不在棋盘上，请重新输入！
haf
输入非法，请重新输入！

```

图 6: 非法输入检测

```

  A B C D E F G H
1 ○ ○ ○ ○ ○ ○ ○ ○
2 ● ● ● ● ● ● ● ●
3 ○ ○ ○ ○ ○ ○ ○ ○
4 ● ● ● ● ○ ● ● ●
5 ○ ○ ○ ○ ● ○ ○ ○
6 ● ● ● ● ● ● ● ●
7 ○ ○ ○ ○ ○ ○ ○ ○
8 ● ● ● ● ● ● ● ●
棋盘1
玩家[ad]○
玩家[sf]
Game List
1、peace
2、reversi
3、gomoku
4、gomoku
游戏结束！请玩家[ad]输入棋盘编号（1-4）/新游戏类型（peace或reversi或gomoku）/退出游戏（quit）

```

图 7: peace 棋盘结算界面

```

  A B C D E F G H
1 ● ● ● ● ● ● ○ ○
2 ● ● ● ● ● ● ○ ○
3 ● ● ● ● ● ● ○ ○
4 ● ● ● ● ● ● ○ ○
5 ● ● ● ● ● ● ○ ○
6 ● ● ● ● ● ● ○ ○
7 ● ● ● ● ● ● ○ ○
8 ○ ○ ○ ○ ○ ○ ● ●
棋盘2
玩家[ad]○ 14
玩家[sf] 50
Game List
1、peace
2、reversi
3、gomoku
4、gomoku
游戏结束！Player 1 [ad]○14
Player 2 [sf]●50
Player 2 [sf]获胜！
请玩家[ad]输入棋盘编号（1-4）/新游戏类型（peace或reversi或gomoku）/放弃行棋（pass）/退出游戏（quit）

```

图 8: reversi 棋盘结算界面

```

  A B C D E F G H
1 ○ ○ ○ ○ ○ . . .
2 . ● . ● . . . .
3 ● . . . . . . .
4 . . . . . . . .
5 . . . . . . . .
6 . . . . . . . .
7 . . . . . . . .
8 . . . . . . . .
棋盘3
玩家[ad]
玩家[sf]●
Current round: 9
Game List
1、peace
2、reversi
3、gomoku
4、gomoku
游戏结束！
玩家[ad]获胜！
请玩家[sf]输入棋盘编号（1-4）/新游戏类型（peace或reversi或gomoku）/退出游戏（quit）

```

图 9: gomoku 棋盘结算界面

```

  A B C D E F G H
1 ● ● ● ● ● ● ● ○
2 ● ● ● ● ● ● ● ○
3 ● ● ● ● ● ● ○ ○
4 ● ● ● ● ● ● ● ○
5 ● ● ● ● ● ● ● ○
6 ● ● ● ● ● ● ● ○
7 ● ● ● ● ● ● ● ○
8 ○ ○ ○ ○ ○ ○ ● ●
游戏结束!
PS F:\vscode\oop\lab\lab5>

Game List
1、 peace
2、 reversi
3、 gomoku
4、 gomoku

棋盘2
玩家[ad]○ 14
玩家[sf] 50
```

图 10: 结束界面