

# 1.总体描述

## 1.1 关于项目

连珠游戏传统上使用黑白两色的石子，在15x15交叉点的棋盘上进行。黑方先行，玩家轮流在空交叉点上放置自己颜色的棋子。首先在横、竖、斜任一方向上形成连续五个棋子的玩家获胜。本项目将实现玩家间对战、结果判断和排名列表管理的功能。如下图所示，空方和白方玩家在游戏棋盘上交替移动石子。在对角线方向上，有五个连续的黑棋子，因此这次黑方玩家获胜。

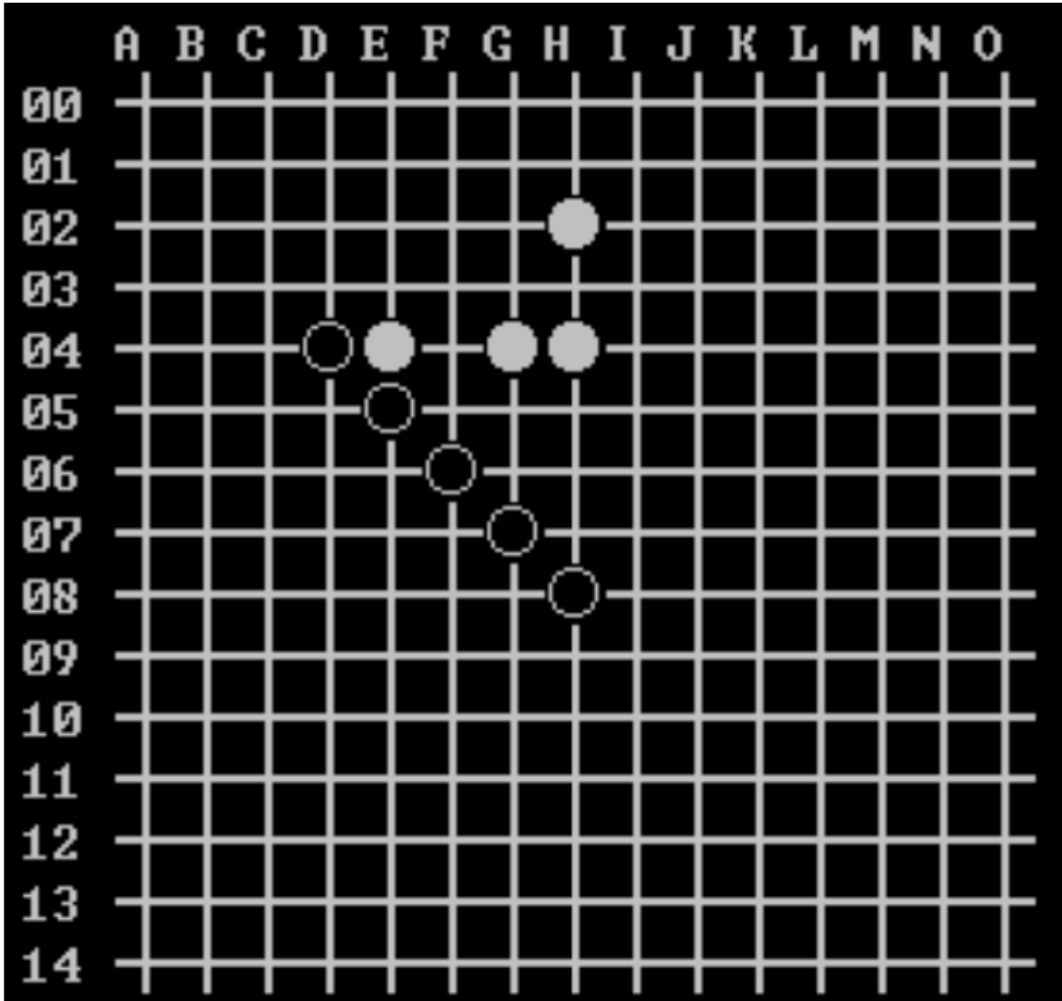


Figure Renju game Command-line Interface

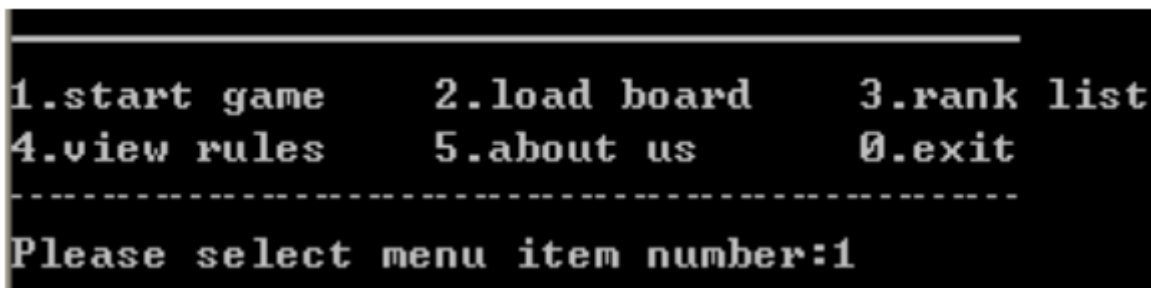
## 1.2 软件功能

### 1.2.1接口：

- **Finish接口：**完成页面显示是为了告知用户程序即将结束。



- **Main接口：**游戏的主菜单被设计成接收玩家的菜单选择以进行游戏操作



- **Welcome接口**:欢迎界面：欢迎页面显示出来，告知玩家程序已经启动，例如：



### 1.2.2游戏：

- **【平局】**（Draw）：

**概述**：当双方都没有获胜，但所有点都被占据时，就发生了平局

**输入**：

- **结果**：输入棋盘上所有点的状态以及被占据点的坐标。

**处理**：通过检查所有点的状态值来确定是否所有点都已填满棋子，并且没有获胜方。如果没有点为空，则发生平局。

**输出**：在平局的情况下，在界面上输出以下信息。



- **【结果】**：（Result）

**概述**：五子棋游戏传统上使用黑白两色的棋子，在15x15交叉点的棋盘上进行。黑方先行，玩家轮流在空交叉点上放置自己颜色的棋子。第一个形成连续五个或更多棋子水平、垂直或对角线排列的玩家获胜。

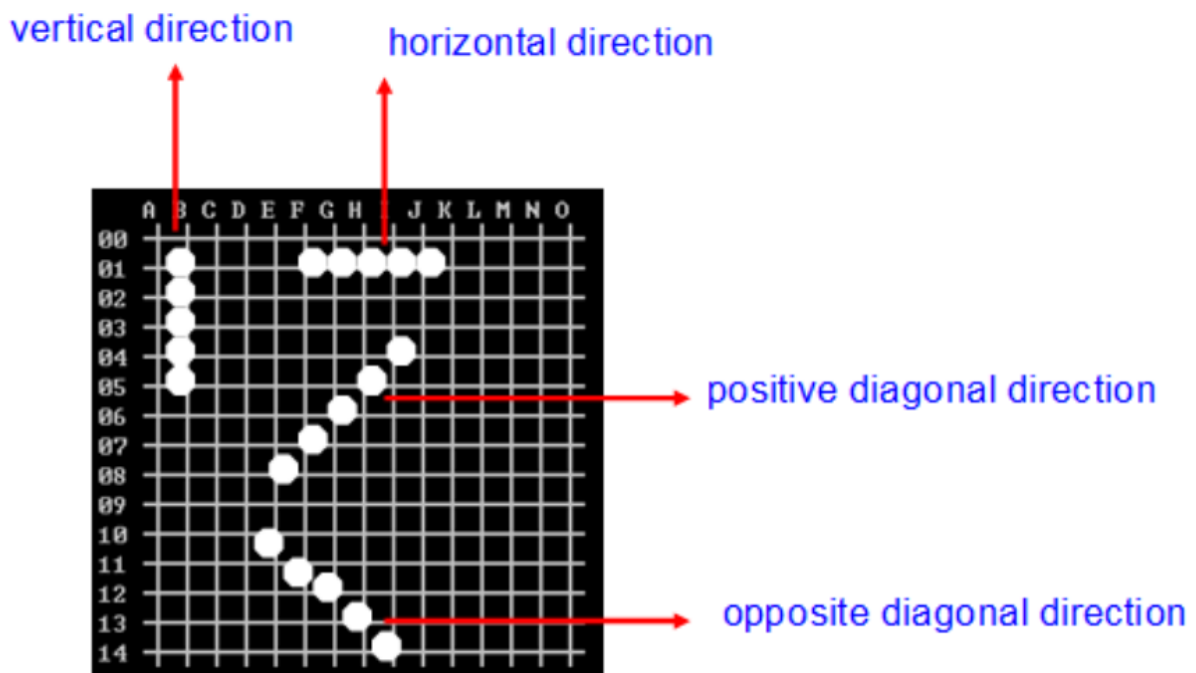


Figure continuous five or more stones

**输入：**输入棋盘上所有点的状态以及被占据点的坐标。

**处理：**获胜者是第一个在水平、垂直或对角线上形成连续五个或更多同色棋子的玩家。从中心棋子开始，向左遍历棋子，当发现不同颜色的棋子时停止遍历，并记录同色棋子的数量。然后，向右遍历棋子，同样当发现不同颜色的棋子时停止遍历，并记录同色棋子的数量。判断同色棋子的数量是否等于或大于五个。如果是，那么它们就形成了一个水平、垂直或对角线上的连续五个或更多棋子的行。同样的方法，遍历四个方向的棋盘。

**输出：**如果既没有胜者也没有平局，则继续游戏。如果出现平局，通知玩家。

如果黑方获胜，输出“黑方获胜”。



如果白方获胜，输出“白方获胜”

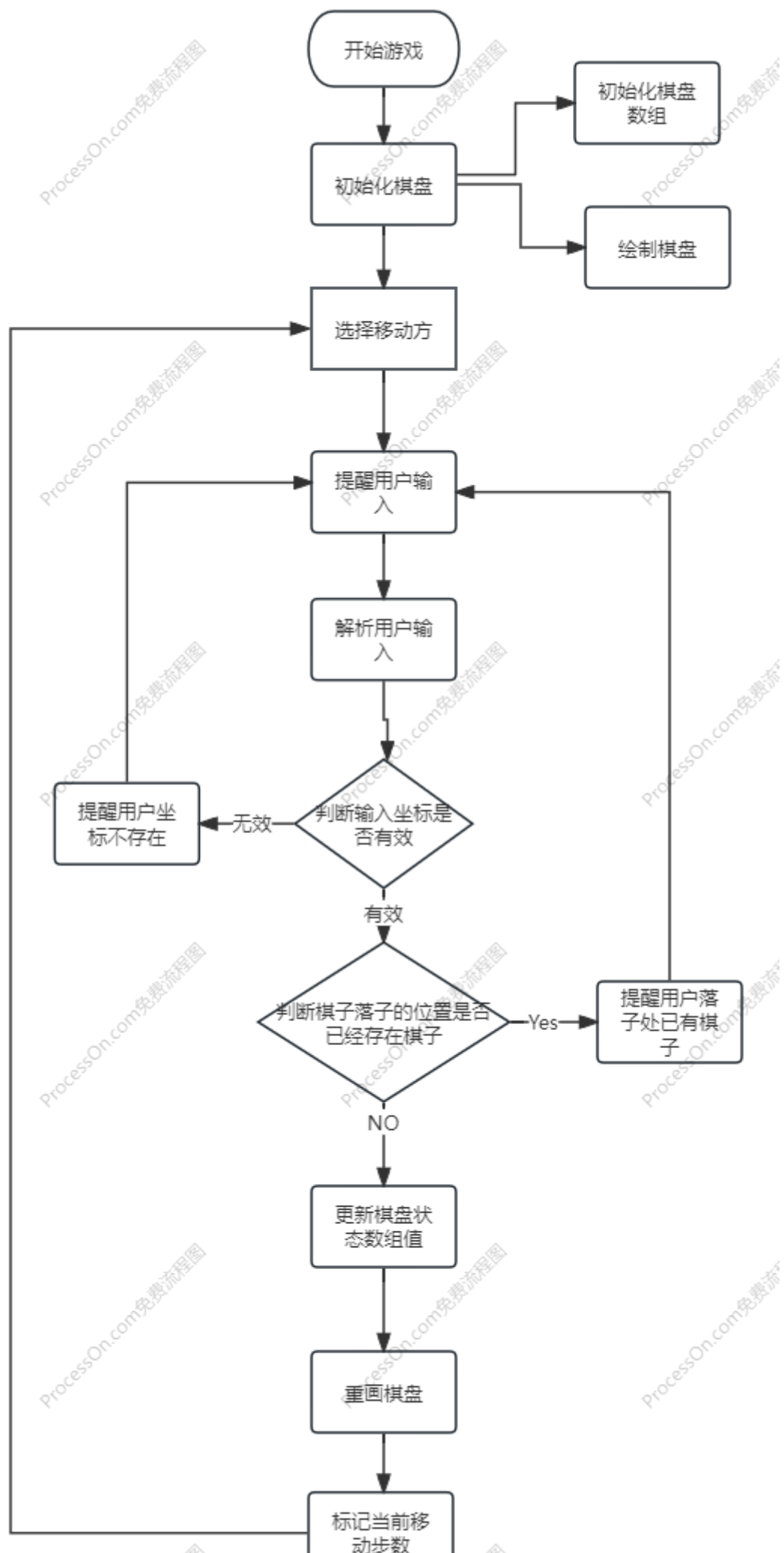


#### • 【移动棋子】（Move）：

**概述：**启用两位玩家相互对战，他们轮流在空交叉点上放置自己颜色的棋子。

**输入：**输入玩家移动，棋盘状态值和用户输入的坐标。

**处理：**确定玩家移动。分析用户输入的坐标。用户输入的坐标由列号和行号组成。列号是字母，从A开始，而行号是数字，从0开始。



(1) 刷新棋盘并通知玩家移动。界面包括标题、棋盘和用户操作通知。界面如下所示。



(2) 功能启动提示：换行，实线，接着是19个"-" 操作指令："黑（白）方请输入坐标数（例如：A0）"。

(3) 输出当前移动的总步数

- 【开始游戏】(Start):

**概述：**系统初始化游戏基础，清空棋盘并开始游戏。

**输入：**输入菜单选项编号"1"。

**处理：**当输入菜单选项编号1时开始游戏。初始化棋盘状态值。绘制一个空棋盘。通知黑方先行。

**输出：**在空棋盘上输出"请白方落子，输入坐标编号（例如：A0）： 。

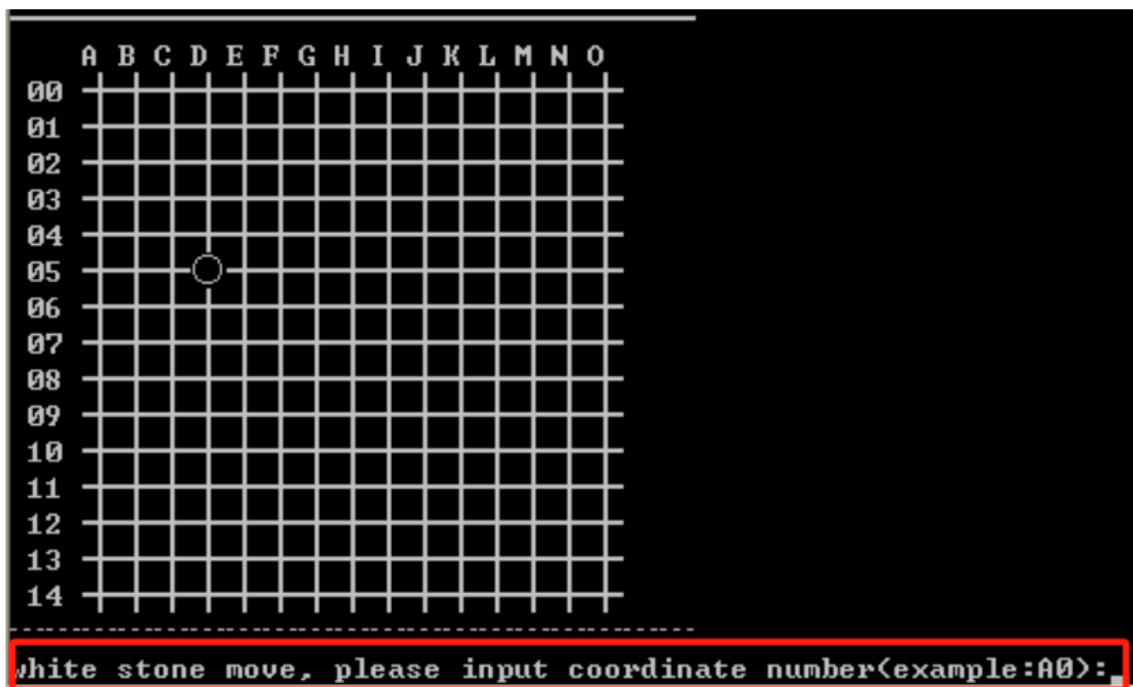


Figure Start game

- 【画/更新 棋盘】 (Draw/Update board):

**概述：** 当前棋盘根据棋盘上棋子的状态进行刷新。

**输入：** 输入棋盘上所有坐标的状态值。

**处理：** 用15\*15的方格棋盘，棋盘的行号和列号进行绘制。列号从A开始，是字母，而行号从0开始，是数字。绘制每个坐标的当前状态。

**绘制：** "十": 交叉点为空。 "●": 放置白色棋子。 "○": 放置黑色棋子。

**输出：** 最新的棋盘状态

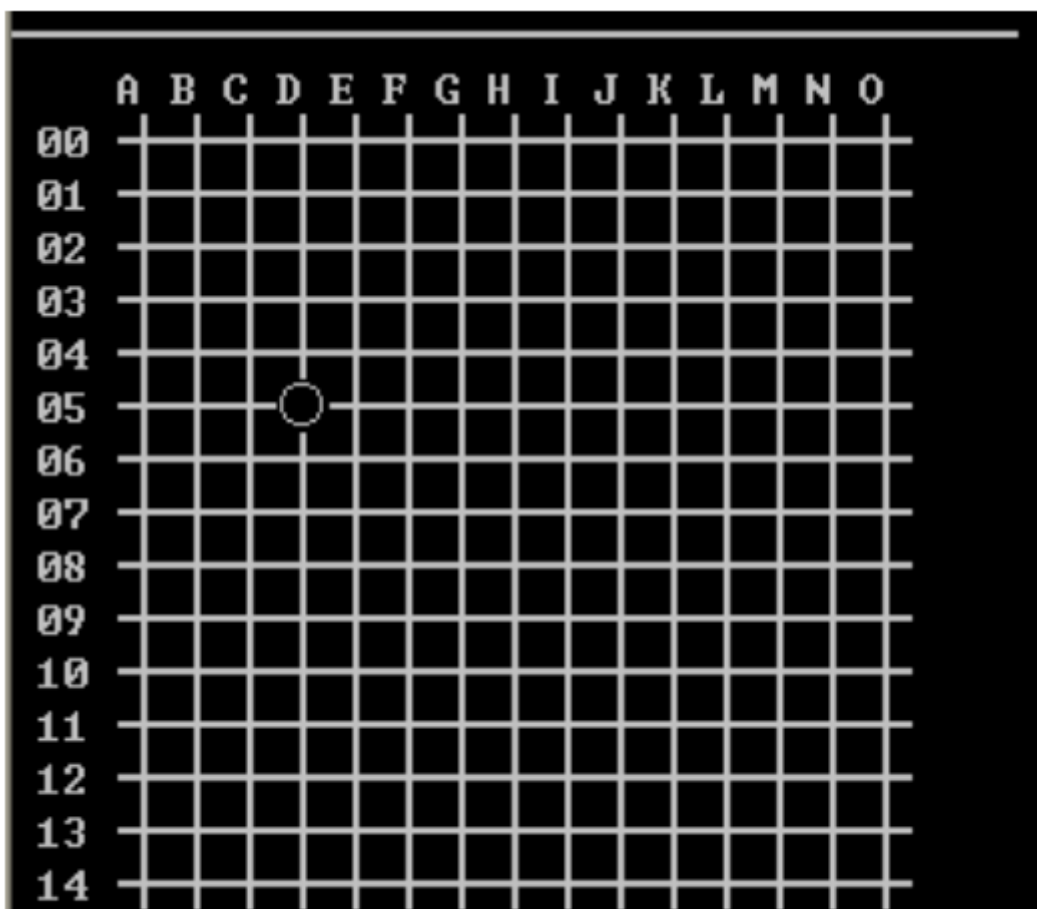


Figure Game board

### 1.2.3 排名表

此模块管理排名信息，包括排名、保存排名列表信息以及显示排名列表信息。

- 【打印排名表】(Print Rank List)

**概述：**此功能允许用户查看当前排名列表信息，包括玩家姓名、排名和移动总和。

**输入：**输入一个菜单选项编号。以及一个排名文件的名称。

**处理：**读取输入的排名文件，逐行读取排名信息，并通过分隔符"##"解析玩家的名字和移动总和。根据移动总和对排名信息进行排序。

意思是：文件中存储的数据是按照如下方式存储

```
ff##9
```

```
f##9
```

根据##来区分姓名和移动总数

**输出：**

rank list		
rank	name	Sum of Moves
1	ff	9
2	f	9
3	aa	9
4	jessie	11
5	uu	17
6	fond	31
press any key to return...		

- 【排名】(Rank)

**概述：**在玩家获胜后进行排名。

**输入：**输入移动的总和、玩家名称，以及现有玩家移动的总和和顺序。

**处理：**

(1) 通过通知用户输入来获取玩家名称。

(2) 确定当前玩家的移动总和排名。排名顺序从小到大上升。如果总和较小，则排名更高。对于移动总和相同的玩家，根据移动的顺序进行排名。

(3) 首先获胜的玩家排名高于后来获胜的玩家。例如，玩家A首先以24步获胜，而玩家B后来以24步获胜。在这种情况下，玩家A的排名高于玩家B。

**输出：**输出最新的排名信息，包括每位玩家的排名、姓名和移动总和。

- 【保存排名表】(Save Rank List)

**概述：**当玩家退出程序时，最新的排名信息会被保存在Ranklist.dat文件中。

**输入：**输入排名序列、排名信息和文件名。

**处理：**将排名玩家的总和保存为 "TOTALCOUNT%d RANKLIST\n" 格式。

#### Note

即相当于只保存前十名玩家的信息。

按顺序保存每个玩家的排名信息，每条排名信息记录占一行。每条排名信息的格式为 "玩家名字##移动次数总和"。分隔符为 "##"。排名信息文件的格式如下。

```
TOTALCOUNT 10 RANKLIST
张三##15
李四##23
.....
```

#### Important

包含排名信息的文件名是Ranklist.dat。以二进制方式wb写入文件



输出：RankList.dat

### 1.2.4 系统信息

该模块主要实现连珠游戏的辅助功能，如显示版本号和相关信息，显示游戏规则的规格，并退出系统。

- 【规则】（Rules）

概述：基本关于五子棋的规则

输入：输入选择菜单数字

处理：确定菜单选项编号是否为4。如果是，显示游戏规则，并通知用户按任意键返回。如果用户按任意键，返回主菜单。

输出：游戏规则包括标题、具体规则以及用户返回操作的通知。界面如下所示。

```
Rules
-----
1. Renju is played by two players against each other.
2. Black stone move first, followed by white.
3. Board: 15×15.
4. The Winner is the first player to get an unbroken row of five stones horizontally, vertically or diagonally.
5. A draw occurs when both players will not get an unbroken row of five stones.
6. Column numbers are letters, while row numbers are numeric.
7. The player who slecets black color stone moves first.
8. player can start game if select Start Game.
9. Move stone: player input row & column number of corresponding point, such as:A0.
-----
Press any key to return...
```

功能开始提示：一个新行，实线，和19个"—"

标题：Game Rules

内容提示：由19个"---"组成的虚线

具体规则：

1. Renju is played by two players against each other.

2. Black stone move first, followed by white.

3. Board: 15×15.

4. The winner is the first player to get an unbroken row of five stones horizontally, vertically or diagonally.

5. A draw occurs when both players will not get an unbroken row of five stones.

6. Column numbers are letters, while row numbers are numeric.

7. The player who slecets black color stone moves first.

8. Player can start game if select Start Game.

9. Move stone: player input row & column number of corresponding point, such as:A0.

操作提示：由19个"---"组成的虚线

操作指令：按任意键返回...

- 
- 【关于我们】（About Us）

**概述：**程序开发者和版本信息显示。

**输入：**输入菜单选择数字

**处理：**确定菜单选项编号是否为5。如果是，则显示有关程序开发者的详细信息，并通知用户按任意键返回。如果用户按下任意键，则返回主菜单

**输出：**输出以下信息：标题，版本信息，版权，以及用户返回操作的通知。

- **【退出】** (exit)

**概述：**这个功能允许玩家在不想要再玩游戏时退出。

**输入：**输入菜单选择数字

**处理：**确定菜单选项编号是否为0。如果是，则结束游戏并退出系统。

**输出：**在界面上输出以下信息。



### 1.2.5 数据信息

按如下数据结构保存排名列表信息

(1) 内存中的格式：在内存中，排名信息保存在一个结构数组或链表中。结构包括玩家的名字和移动次数的总和。

(2) 在文件中格式：将排名玩家的总和保存为格式 "TOTALCOUNT%d RANKLIST \n"。

按顺序保存排名信息，每条排名信息记录占用一行。 每条排名信息的格式为 "玩家名##移动次数总和"。分隔符为 "##"。

排名信息文件的格式如下所示。

```
TOTALCOUNT 10 RANKLIST
张三##15
李四##23
.....
```

- 排名信息文件的格式是Ranklist.dat。

### 1.2.6 技术要求

文件格式：文本模式文件或二进制模式文件格式。

编码标准：C编码标准。

## 1.3 软件质量属性（可以忽略这点，就是对输入有提示即可）

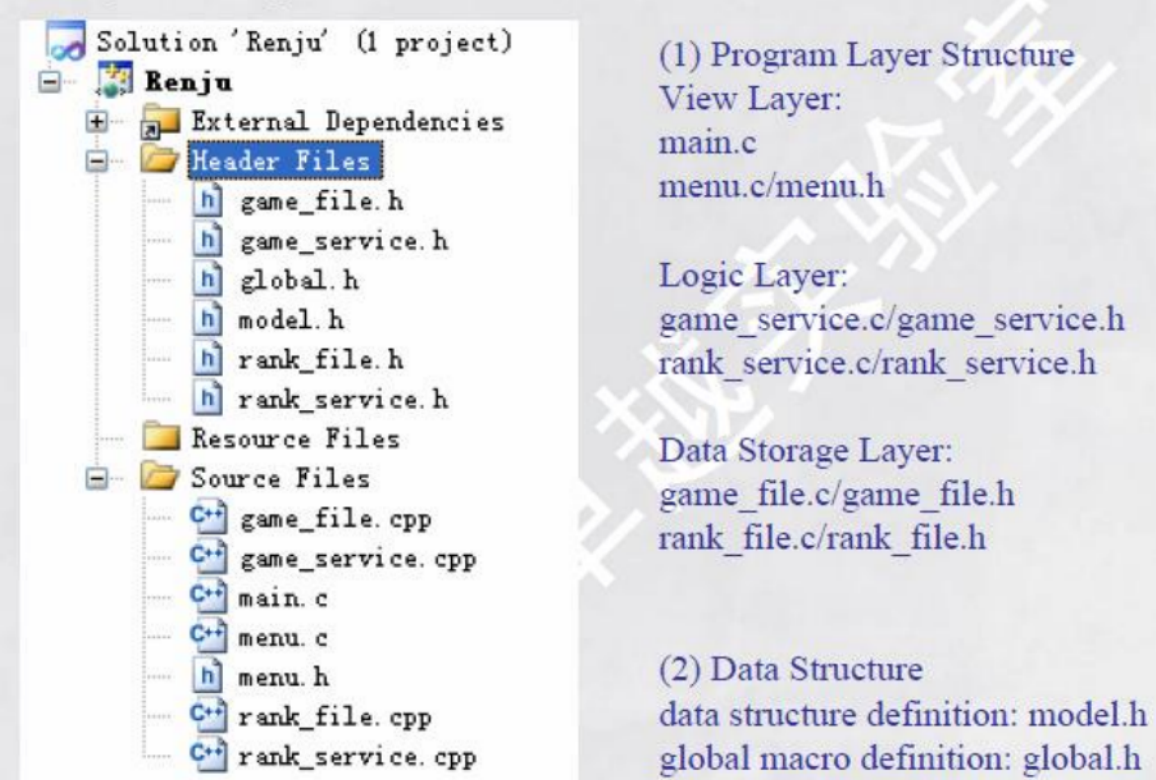
适应性：确保在原有基础功能上进行扩展，并向原有系统添加新的业务功能，这些功能可以轻松添加而不影响原有系统的结构。

容错性：在系统崩溃、内存不足的情况下，不会导致系统故障，系统可以正常关机和重启。

可恢复性：系统应在故障解决后能够正常运行。

可用性：界面设计应当合理，集中系统功能并使系统易于使用。系统应阻止用户的非法输入数据或操作，为复杂处理提供向导和注释，并为用户提供便捷的帮助信息。

## 1.4 程序结构



(1) Program Layer Structure  
View Layer:  
main.c  
menu.c/menu.h

Logic Layer:  
game\_service.c/game\_service.h  
rank\_service.c/rank\_service.h

Data Storage Layer:  
game\_file.c/game\_file.h  
rank\_file.c/rank\_file.h

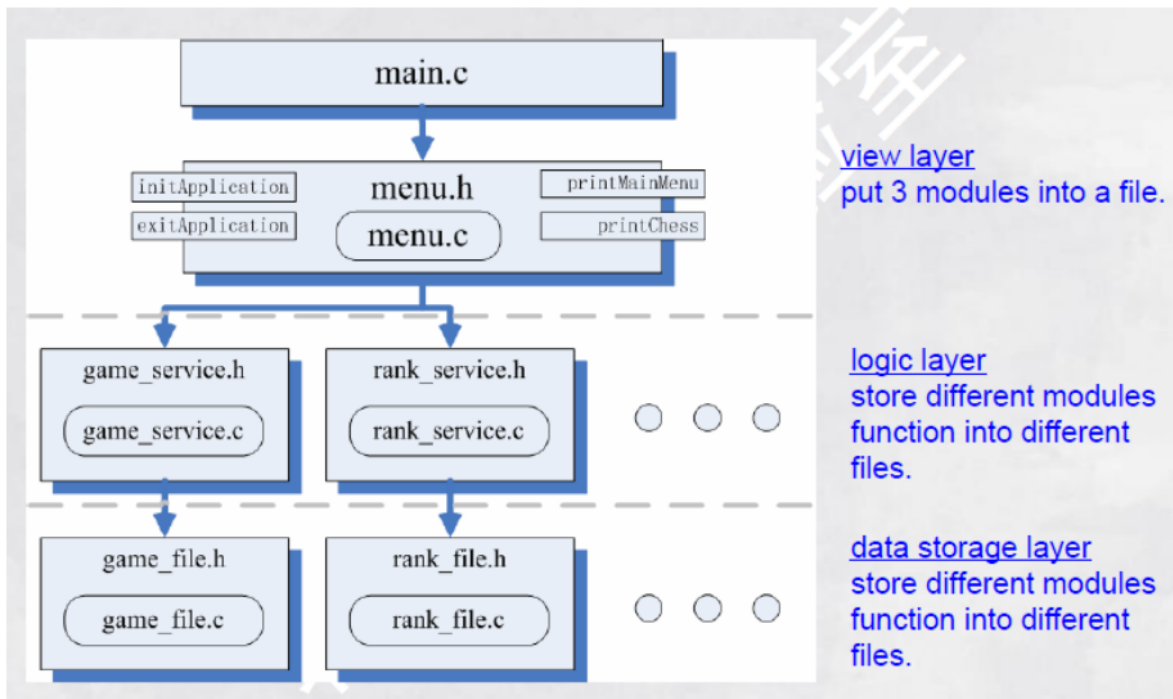
(2) Data Structure  
data structure definition: model.h  
global macro definition: global.h

**接口层**（其实就是界面，你能看到的界面的绘制等）：包括游戏菜单的选择、坐标的输入、绘制棋盘、输出棋盘信息、打印排名表等【就是能看见的所有信息】

**逻辑层**：对数据的处理层，也就是对棋盘状态的判断，比如黑方获胜还是白方获胜；输入棋子坐标，判断在该坐标下是否可以落子；对排名进行排序；维护排名表等。

**数据存储层**：对数据的操作，比如写入文件和从文件中读取数据。

## (2) specific layer structure design



对上图的解释:

- **model.h**

```
//Status type of coordinate point
enum PointStatus
{
    STATUS_BLANK = -1, // Blank
    STATUS_BLACK = 0, // Black stone
    STATUS_WHITE = 1 //white stone
};

// Coordinate point data
typedef struct Point
{
    int row;    //Row number (行信息)
    int col;    // Column number (列信息)
    int status; //当前棋子状态, 其值为PointStatus中的值
}
```

- **global.h**

```
#define CHESS_BLANK "⌚" // Blank icon(empty)
#define CHESS_BLACK "o" // Black stone icon
#define CHESS_WHITE "●" // white stone icon
```

- **main()** 主函数中要包含以下三个菜单函数

main.c函数引用menu.h头文件, 调用对应的菜单函数, 来展示页面信息。

```
welcome Interface: initApplication();
Main Interface : main menu;
Finish Interface : exitApplication()
```

- **menu.c menu.h**

menu.h中需要包含所有的接口文件，也就是所有的页面展示函数的声明。

menu.c中实现对menu.h中声明的函数

```
menu.h: functions prototype declaration
The functions defined in menu.c:
1) void initApplication();//初始化应用程序，输出欢迎信息
2) void exitApplication();//当退出应用程序时，可以输出游戏结束信息并释放程序数据。
3) int printMainMenu();//绘制主菜单页面
4) void playGame(): //开始游戏
5) void printGameRule(): //输出Rules
6) void printAboutUs(): //输出About us
7) void printRanklist(): //输出排名表
```

- **game\_service.c game\_service.h**

game\_service.h定义判断棋子胜负、棋子落子是否有效等函数

game\_service.c实现game\_service.h中声明的函数

```
1) int judgeHorizontal(const Point sPoint); // 判断行是否五个棋子连续
2) int judgeVertical(const Point sPoint); // 判断列是否五个棋子连续
3) int judgeHyperphoria(const Point sPoint); // 判断正对角线上的连续石子数
4) int judgeHypophoria(const Point sPoint); // 判断负对角线上连续石子的数量
5) void initStatus();// 初始化棋盘状态值，状态值设置为空白。
6) int getStatus(const Point spPoint); // 获取指定坐标下的棋盘状态
7) void setStatus(const Point spPoint); // 设置指定坐标下的棋盘状态
8) int judge(Point spPoint); // 判断是否一方获胜
9) int judgeDraw();// 判断是否平局
```

- **rank\_file.c rank\_file.h**

```
1) int writeRanklist(Rank* psrRanks, const int nSize); // 把排名信息写入文件
2) int readRanklist(Rank* psrRanks, const int nMaxSize); // 从文件中读取排名信息
3) int getRanklistCount();// 从排名列表文件中获取排名列表中玩家的数量。
```

## 排名列表管理——链表

- **menu.c**

添加printAddRank()函数：添加玩家排名信息。

添加printRanklist()函数：输出排名列表。

由于链表数据需要手动释放，在exitApplication()函数中，我们只需要调用释放链表数据的函数。

- **rank\_service.c,rank\_services.h**

添加**judgeOrder()**：判断排名顺序，遍历链表节点，在遍历过程中找到第一个大于当前移动数字的节点。在遍历过程中，通过计数来获取排名。

添加**insertRank()**：插入一个排名节点。遍历插入点，将前一个节点指向新节点，同时新节点指向后一个节点

添加**saveRanks()**：保存排名信息。遍历链表以保存排名信息。

添加**getRankSize()**：获取排名玩家的数字。遍历链表，计算并统计玩家数量。

添加**getRanks()**：获取排名信息读取链表；创建数组；将链表数据复制到数组中；返回上层。

添加**initRanks()**：初始化排名信息。创建头节点；读取文件返回的排名列表数组；按顺序遍历并插入。

释放链表：**clearRanks()**；按顺序遍历链表并释放。

- **rank\_file.c rank\_file.h**

**writeRanklist()**：把排名信息写入文件

**readRanklist()**：读取文件中的排名数据，并保存在链表中后返回。

- **game\_file.c game\_file.h**

考虑到每个用户可能有自己的用户名，以及拓展功能的悔棋（利用日志txt存储信息）等，因此这两个文件可以存储上述信息。此外，主界面还有一个load board，我猜测是可以保存未完成的游戏棋盘，然后可以恢复？不太清楚，看作业文件里没给这个

## 1.5 说明

1. 建议采用上述函数的定义，也可以有些出入。
2. 建议设计更美观的五子棋界面，上述界面都可以更改成自己特色的界面。
3. 基本功能只占80%，扩展功能占20%，比如用图形界面实现五子棋界面（可以用EasyX实现图形用户界面，具体安装和学习看官网easyx.cn），再比如加背景音乐，悔棋功能，数据库存信息等

【加背景音乐、悔棋可以考虑】

4. 项目以答辩方式考核

## 1.6 交付物

1. 提交总结报告（见模版）
2. 将整个工程目录下所有文件打包上传，录屏程序运行视频上传。