北京理工大学计算机学院
《Java 程序设计》课程设计课题报告

班级 07152001 班　　**学号** 1120201198　　**姓名** 史桠彬

1 程序的运行环境、安装步骤

（1）**运行环境**：JDK 17

（2）**程序的组成部份**：一份 jar 文件。

（3）**安装步骤**：

1） 安装 JRE 17。

2） 将本程序 jar 文件复制到目标计算机上。

3） 运行程序：

方法一：打开命令提示符窗口（Win 键+R），输入命令：java -jar myApp.jar 即可运行。

方法二：在资源管理器中双击本文件，即可运行。

2 程序开发平台

（1）**代码行数**：约 1200 行。

（2）**开发环境**：IntelliJ IDEA 2021.2.2 + JDK 17 + SQLite 3

3 程序功能说明：

（1）**程序基本介绍**：

本程序名为“2048 无尽版”，模拟 2048 小游戏，旨在完成原有基本各项功能之余，在可玩性与趣味性上有所增加与创新，实现游戏改进。

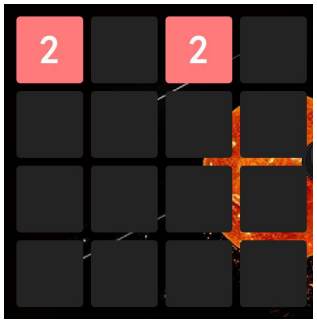
（2）**不同面向用户所能实现的宏观功能用例图**：

玩家：可以通过运行 jar 包启动游戏进入游戏界面进行操作。

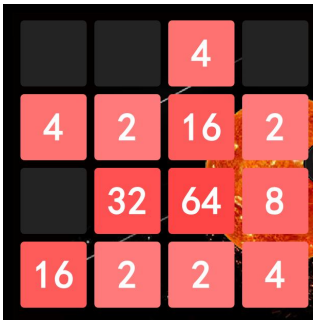
管理者：可以启动游戏，也可以根据玩家游戏得分进入数据库查询玩家历史成绩。

（3）**重点功能详解**:

①. 玩家通过控制键盘上的↑ ↓ ← →实现方格的移动，以期将相同数值的方格进行合并，当游戏面板内无可合并方格或无方格则视为游戏结束。



（游戏开局）



（游戏进行）

（游戏结束）

②. 玩家可以通过控制键盘上的“Enter”、“Control”、“Space”使用鞭炮、魔方、炸弹道具，解决游戏界面方格合并受阻的情况。

（鞭炮未使用）↓

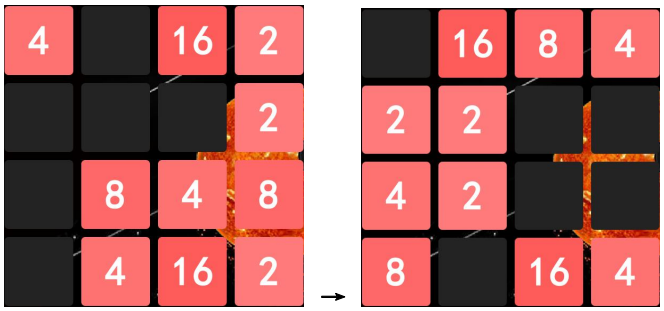
（炸弹未使用）↓

（魔方未使用）↓

（鞭炮已使用）

（炸弹已使用）

（魔方已使用）



（以魔方道具为例，使用后随机扰乱游戏界面中方格位置，增加用户游戏可能性与乐趣性。）

③. 玩家在游戏过程中将不断通过合并方格更新个人得分，并在游戏结束获知个人最终得分。

（以某次游戏过程得分为例）

④. 管理者可以通过数据库及源程序对玩家历史成绩进行查询与监视，更好地发掘用户平均表现，了解确定游戏改进方向。

```
-------
time = 15
score = 6144
-------
time = 15
score = 6888
-------
time = 15
score = 1272
```

（以控制台输出为例）

4 程序算法说明及面向对象实现技术方案

（1）核心算法说明：

①. 整体采用建立 4*4 数组的方法构建游戏面板。

②. 对于玩家进行的移动操作，借对每一特定方格上下左右的值是否相同进行判断与合并。

③. 利用随机数建立索引产生随机方格。

④. 利用列表对部分方格进行值清除与值互换，完成道具制作。

⑤. 在任意操作后利用判定游戏界面空白方块数量的方式确定游戏进程。

合并方格（与移动操作同时进行）：

```
//双层循环判断
for (int i = 0; i < 4; i++) {
    for (int j = 1, k = 0; j < 4; j++) {
        //当前方块不为空白方块则可进行移动
        if (blocks[i][j].value > 0) {
            //当前方块可以和左边方块合并
            if (blocks[i][j].value == blocks[i][k].value) {
                //左边方块值翻倍，分数增加
                mark += blocks[i][k++].value <<= 1;
                //当前方块归为空白方块
                blocks[i][j].value = 0;
                //将flag标志改为true
                flag = true;
            }
            //左边为空白方块
            else if (blocks[i][k].value == 0) {
                //当前方块移至左边
                blocks[i][k].value = blocks[i][j].value;
                //当前方块归为空白方块
                blocks[i][j].value = 0;
                //将flag标志改为true
                flag = true;
            } else if (blocks[i][++k].value == 0) {
                blocks[i][k].value = blocks[i][j].value;
                blocks[i][j].value = 0;
                flag = true;
            }
        }
    }
}
return flag;
}
```

（以左移为例，通过对每一行每一列的方格进行判定确认该方格是否为空白方格，及其与周围方格是否值相同进行合并判断。　时间复杂度：O（n^2））

产生新方格：

```
//产生随机新方块
private void newBlock() {
    //获取游戏板上的空白方块
    ArrayList<Block> list = findBlankBlocks();

    //游戏板上有空白方块
    if (!list.isEmpty() && flag) {
        //创建Random对象
        Random random = new Random();
        //随机获取一个空白方块
        Block randomSelected = list.get(random.nextInt(list.size()));
        //随机获取一个方块数值，2和 4出现概率比为4:1
        randomSelected.value = (random.nextInt( bound: 5) / 4 == 1) ? 4 : 2;
        //获取结束后，将flag标志改为false
        flag = false;
    }
}
```

（获取空白方格位置后，利用随机数产生方格值建立新方格。　　空间复杂度：S（n））

判断游戏进程：

```
//判断游戏是否结束
private boolean judgeEnd() {
    //将分数同步到分数栏
    labelMark.setText(mark + "");
    //获取游戏板上的所有空白方格
    ArrayList <Block> One = findBlankBlocks();
    //建立插入数据库对象
    InsertData insertData= new InsertData();
    //建立获取当前时间对象
    SimpleDateFormat sdf = new SimpleDateFormat();
    //获取日期参数
    String dateNow = sdf.format(new Date());
    String date=dateNow.split( regex: " ")[0];
    //获取游戏终止分数
    String score=mark+"";

    //游戏板中全是空白方格（只有可能是利用道具达成） 游戏结束
    if (One.size()==16) {
    //将成绩插入数据库
        if(c){
            insertData.insertInfo(date,score);
            c=false;
        }
        return true;
    }

    //游戏板中存在空白方格，游戏必然没有结束
    if (One.size()!=0){
        return false;
    }
```

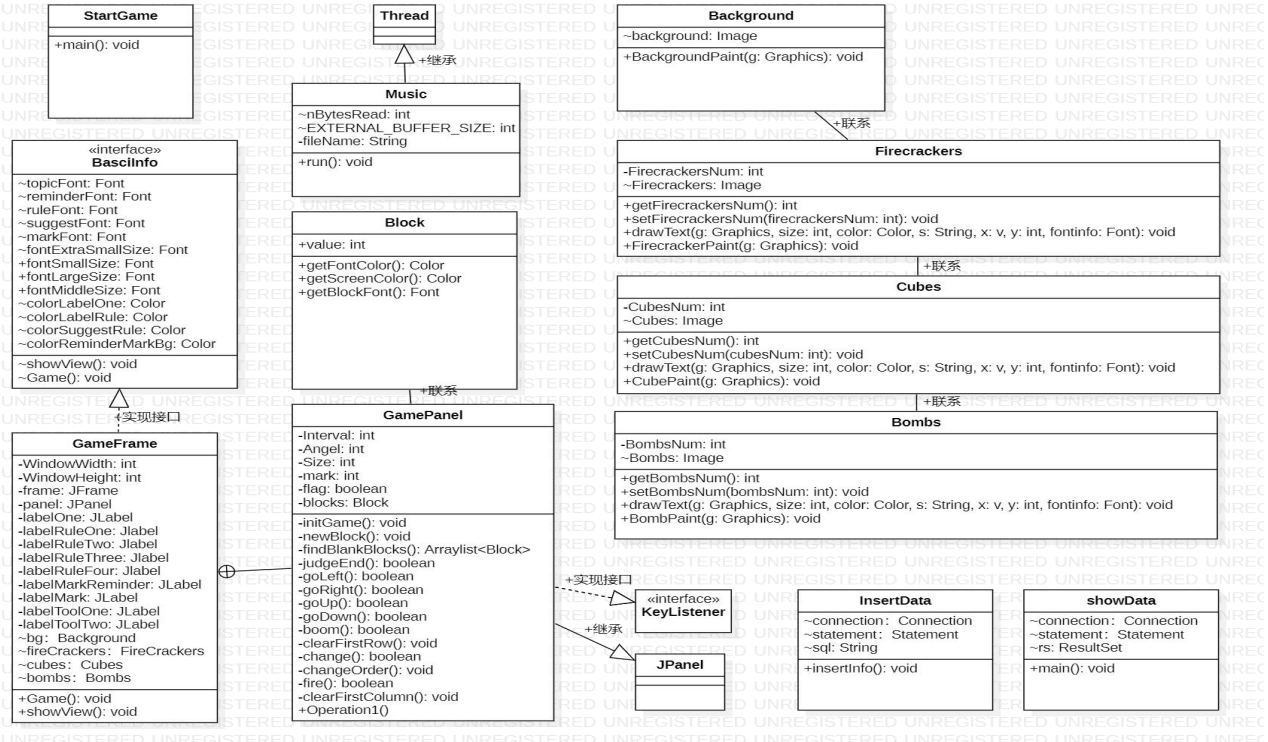
```java
//通过两层循环判断游戏板中是否存在可以合并的方格
for (int i = 0; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
        //对于左上角的3*3方格的合并判断
        if (i < 3 && j < 3) {
            //同行合并、同列合并
            if (blocks[i][j].value == blocks[i][j + 1].value || blocks[i][j].value == blocks[i + 1][j].value) {
                //可以合并
                return false;
            }
        } else {
            //对于靠下的1*3的方格的合并判断
            if (i == 3 && j != 3) {
                //同行合并
                if (blocks[i][j].value == blocks[i][j + 1].value) {
                    //可以合并
                    return false;
                }
            }
            //对于靠右的3*1的方格的合并判断
            if (j == 3 && i != 3) {
                //同列合并
                if (blocks[i][j].value == blocks[i + 1][j].value) {
                    //可以合并
                    return false;
                }
            }
        }

    }
}
```

```java
    //游戏结束，将成绩插入数据库
    if(c){
        insertData.insertInfo(date,score);
        c=false;
    }
    return true;
}
```

（通过获取空白方格信息及能否合并情况判定游戏是否结束　　时间复杂度：O（n^2））

（2）数据结构和算法的面向对象实现技术方案



程序共一个 jar 包，其中包含 11 个类，有 1 个为内部类，1 个接口。

类分别为：
Background（用于绘制背景）
Bombs（用于添加炸弹道具，用于消除首列方格）
Cubes（用于添加魔方道具，用于乱序所有方格）
FireCrackers （用于添加爆竹道具，用于消除首行方格）
Block （方格）
GameFrame（游戏主界面）
Music（设置背景音乐）
InsertData（插入数据到数据库）
ShowData（在控制台展示数据库中玩家历史成绩）
StartGame（开始游戏，游戏入口）
GamePanel(GameFrame 的内部类，用于完成游戏板内容)

接口为：BasicInfo（基本信息及抽象方法）

其中：
GameFrame 实现 BasicInfo 接口。
GamePanel 继承 JPanel 类，实现 KeyListener 接口。
Music 继承 Thread 类。

5 技术亮点、关键点及其解决方案

- **本程序的亮点：**

1. 比原始的 2048 游戏持续力强。原版 2048 游戏当合成一个值为 2048 的方块后即终止游戏，游戏持续性差，本程序支持合成方块值达 16384 及以上数值方块，为“高玩”提供更优游戏体验。

2. 本程序优化了终止游戏判断，使得在判定游戏结束时不会出现最右列上下合并失效或最下行左右合并失效的情况，减少游戏可能存在的 bug。

3. 本程序添加了三种道具，分别是炸弹道具，用于消除首列方格；魔方道具，用于乱序所有方格；爆竹道具，用于消除首行方格，极大地丰富了玩家的游戏选择与游戏策略，玩家可以在特定时机使用道具获得不同效果，极大增强游戏趣味性（ps：游戏道具设计也有一点小心思，三个道具并不能为当前游戏局势带来绝对帮助，甚至可能使得局势更糟，强不确定性与强策略性极大丰富了游戏可能）。

4. 本程序的方格设计采用设计原理，根据值将不同方格颜色由浅红至深红进行排序，当游戏进行到一定程度，相近红色会带来较大干扰性，从外界增加游戏难度。

5. 本程序配有适合的背景音乐，为玩家带来更佳的游戏体验。

- **本程序的技术关键点**

1. 使用 Sqlite3 来保存数据，通过数据库帮助管理者获知玩家的历史成绩，通过数据对游戏平均难度进行估量，帮助设计者根据统计原理更好地获知改进方向。

2. 使用 swing 和 awt 进行界面设计，基本完成游戏界面可视化任务。

- **遇到的技术难点及对应的解决方案：**

（1）数据库不够形象化。

问题描述：基本操作只能通过 cmd 命令行完成，使用 DB Browser for SQLite 也无法理解。

最终的解决方案：网络查找资料使用 Navicat Premium 15 进行形象化数据库操作与管理。

（2）无法在 IDEA 上连接 sqlite 数据库。

问题描述：在完成 sqlite 下载后建立数据库后，无法成功连接到数据库，一直报错。

最终的解决方案：根据老师指导，发现数据库名与表名不相同，导致连接字符串有误，更改表名后完成连接。

（3）图形可视化操作举步维艰，不会操作。

问题描述：JavaFX 使用不流畅，学习吃力。

最终的解决方案：通过查询资料，还是选择用 swing 和 awt 进行一些界面可视操作。

（4）道具消耗存在同步问题。

问题描述：定义游戏道具后，发现使用一个道具，另外两个道具也会一起使用，出现问题。

最终的解决方案：经过查询资料，发现是 switch 语句的后几个选择支忘记加 break 语句，导致选择分支“贯穿”，按一个键完成许多键的相应方法。

（5）背景音乐完成后无音乐产生。

问题描述：在配置背景音乐方面，总是报各种奇奇怪怪的错。

最终的解决方案：经过查询资料，通过继承 Thread 类创建相应音频对象完成音乐设置。

6 简要开发过程

11 月 19 号 互联网查找资料确定选题

11 月 21 号 构思程序葆有的特色与创新玩法

11 月 23 号 绘制整体设计图

11 月 24 号 设计算法

11 月 27 号 查询资料研究相关算

11 月 30 号 查询资料进行相关内容学习

12 月 5 号 开始编写程序

12 月 11 号 程序编写完成（包括整体 java 文件及数据库连接），对程序进行测试

12 月 12 号 程序开发工作完毕，编写及整理文档

7 个人小结及建议

先让我喘口气，是真的没有想到我终于到了这一步——写结课设计的最后一个部分，我可真是太开心啦！JAVA 程序设计完结撒花~~~~

当时选学 JAVA 的原因是因为感觉迟早要学，与其以后自学不如跟着老师学，于是就报了 JAVA 课。一学期下来收获满满，但也确实很累。由于课时设置的原因，明明内容很多的课不得不被压缩时间到 11 周结束，整个人起初有点无语，也后悔过为什么不在前三周听金老师的话“被劝退”，但是如今完成设计后，我不由得感谢那个没有退课的自己哈哈哈哈哈哈。

如果非要说建议，我只希望老师讲课的时候可以节奏慢一点，可能是难度提高，到后面的内容我有点跟不上（.....可能是我太菜了）。

做这个 2048 其实是有私心的，我自己很喜欢益智类小游戏但是做数独确实没什么意思，原版 2048 又是有尽头的——出现一个值为 2048 的块游戏就结束，因此萌生了复刻一个 2048 但要做成无尽版的念头。后来又根据朋友的建议增加了游戏道具，这确实是一个不错的提议。最后就有了这一版 2048 小游戏。

也算把很多 JAVA 课堂上学到的东西用到了自己的小程序里，心里着实很骄傲，也算基本完成了一个功能完整的“小玩意”~

最后，诚挚地感谢一下金老师。我至今还记得金老师会在课上偶尔说一些关

于软件公司的历史以及不同 JDK 的版本更迭，比如 Oracle 的变迁.....我一开始感觉这些与学习 JAVA 无关，可后来竟从中悟出一点点“世事更替”的感觉。在版本更新的一个个故事里，我发现了比 JAVA 学习更加有意义的——一个优秀计算机从业者的必经之路，不断适应崭新世代，持续学习，敢去创新，不然就被淘汰，但我后来想想，这个道理好像整个人生都适用。我还记得金老师嘲笑那些出了新版本就叫苦不迭跑到官网下面喊着“千万不要再更新版本了”的人，自己倍感惭愧，因为我自己原来就是一个不愿意尝试新事物的人，能用固有知识解决问题绝不创新，现在想来真是可耻。

写到这里，这么一想，从 JAVA 课堂收获的东西真不少。

最后的最后，稍微提一嘴，我找到了您在 CSDN 上的账户，发现您把 id 设置为“数字世界一凡人”，这个名称着实让人能想好久。仔细想想，或许我们每个人，在这个看不到尽头的数字世界都是“一凡人”吧。