# 大作业2048书面报告

1.概览

开发工具：

Visual Studio 2022（Console控制台版，提交的版本）

Qt Creator（Qt6.0内核，GUI版，WIP，未提交）

完成的功能：

1. 2048在某个2生成概率下基础功能
2. 三个预设难度与一个自定义难度功能
3. 趣味模式——盲盒模式的设计与实现
4. 积分机制与基于fstream的排行榜

当前版本：v2.1

2. 开发历程

2.1 最初的设计与模块化思路对这个程序开发带来的便利

在这个控制台版2048的开发过程开始，我考虑到了我目前所学到的C++语言知识，以及有限的时间中我可以做到的内容，最终列出了一版2048的设计稿。如图所示。

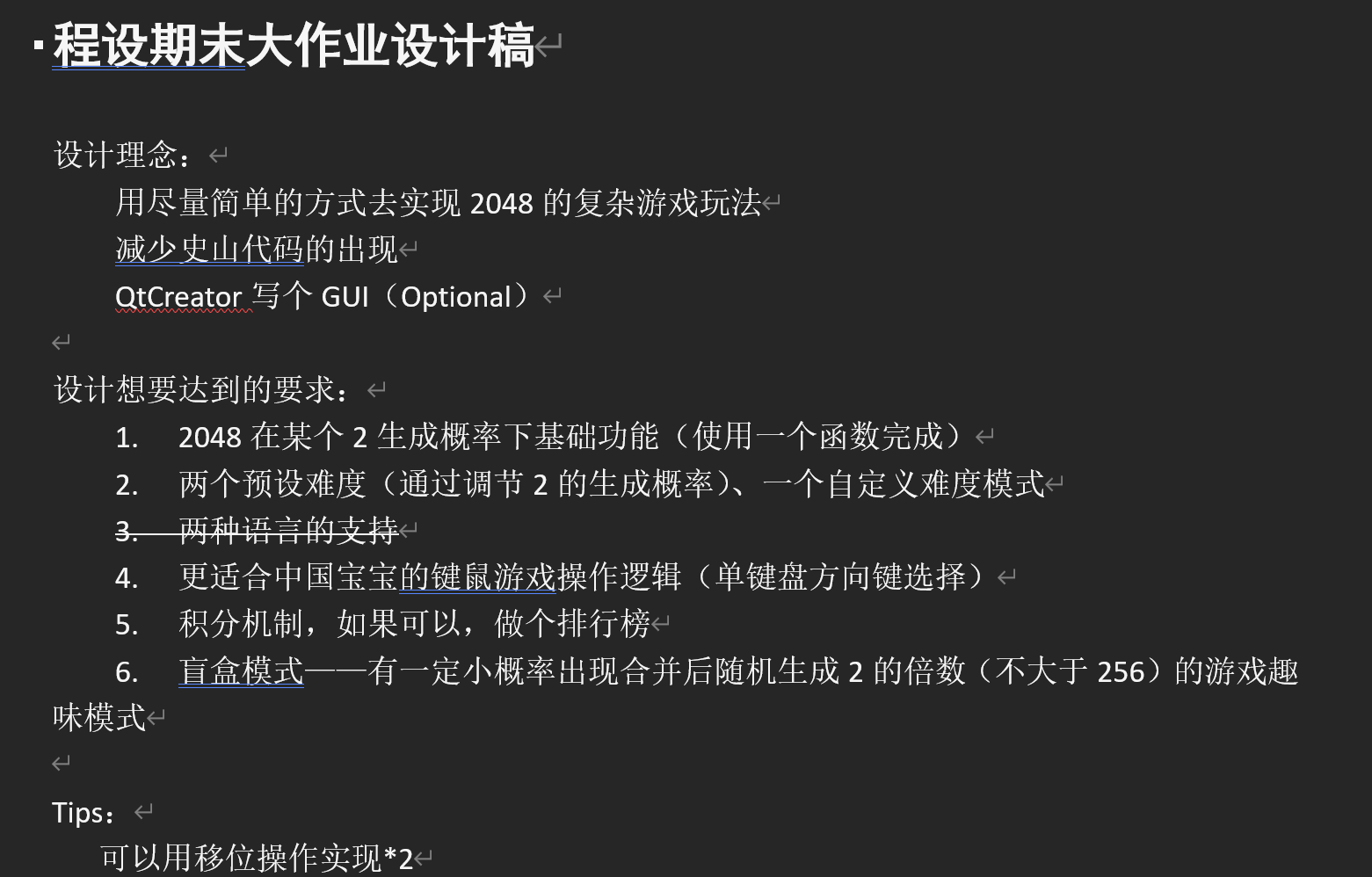


图2.1-设计稿

有了这个设计稿明确我接下来需要在这个程序中实现怎样的功能之后，在后续的程序编写过程中我会有意识的给程序留一些可供后续调试、维护与增加新功能的接口，这样的思路为我之后更新与修改Bug的时候提供了很大的便利。基础的操作由于可以被拆分成很多简单而重复的问题，我就尽量将这些问题写入函数，尽管整体来看这会使代码中函数的数量大幅增加，但是实现功能的代码可读性同样也被提升了。

**下面我将举一个简单的例子来说明我的这种编写代码风格给我带来的便利。**如图2.2，这是我编写的一个函数，命名为RandGenFlag，用于在每轮操作前判断是否需要生成一个随机的2/4，在判断的过程中，我们需要对上下/左右是否需要合并进行判断，但是这样的合并判断的过程是高度重复的，所以我将这个函数独立出来，用一个int值传入操作方向，通过这种模块化的方式在上下左右的操作函数中调用。而在2.0版中，经过对比原版2048，我发现我的生成随机2/4的条件出现了问题，所以在修复这个问题的时候，我只需要修改RandGenFlag函数中的条件语句就可以了，而不需要分别进入我的操作函数UpCopy、DnCopy、LfCopy、RtCopy去修改这个判断功能。（图2.3，更新日志）

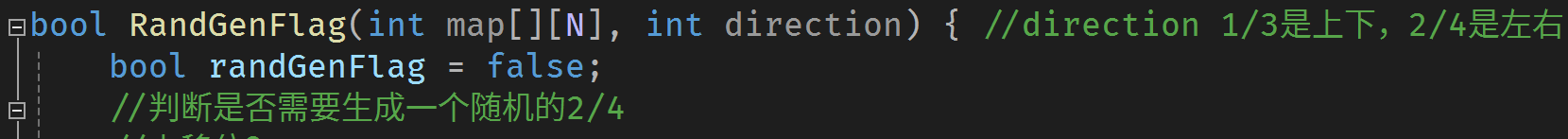


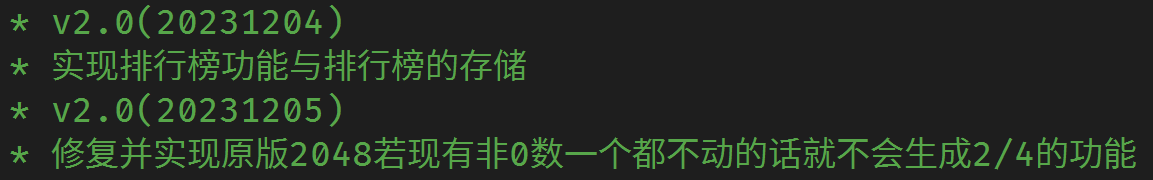
图2.2-RandGenFlag函数，用于在移位前判断是否需要生成随机的2/4

图2.3-Changelog，这个更新只动了RandGenFlag

2.2 Debug的方式

在v1.0功能基本编写完成后，我遇到了一个很大的问题。操作会打乱整体棋盘的数字分布。在基本的试玩中，我将问题锁定在了向右操作的过程，在程序中对应了RtCopy函数。我设定了断点来测试，但是由于整个函数中设计了三轮循环体，且第一轮和第三轮是三重for，第二轮是双重for，如果用断点Debug一方面不方便直观地看到存储棋盘的二维数组的动态变化，另一方面由于for循环的循环次数过多导致调试时间长、调试没有效率。而由于整个棋盘每次的游戏都是随机分布的，也无法用断言的方式去抛出问题。所以我想起了许多大型的软件会在目录中生成一个log.txt存储程序运行的状态，于是我便着手用文件流操作生成调试2048的Log。为了编写log的输出函数，我想起了曾经在OI中用过的fstream头文件，并花了一定的时间在CSDN上了解了fstream的用法（这另一方面也降低了我在v2.0的排行榜功能的实现难度）。Log的输出格式，我选取了这样一种较为方便阅读的方式输出。通过在操作函数中调用LogMap，我就可以在log.txt中较为直观的看见每轮操作棋盘的变化，最后发现了RtCopy的编写问题。（Swap(map[j][i+1],map[j][i]);写成了Swap(map[j][i+1],map[i][j]);）

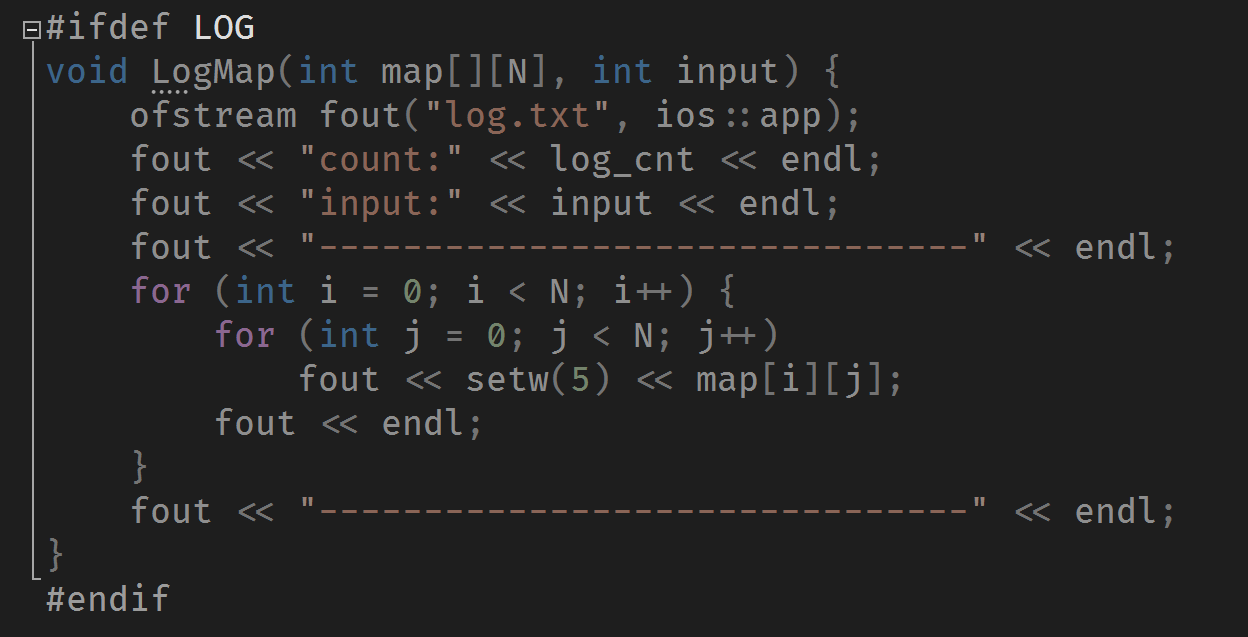


图2.4-LogMap函数的代码

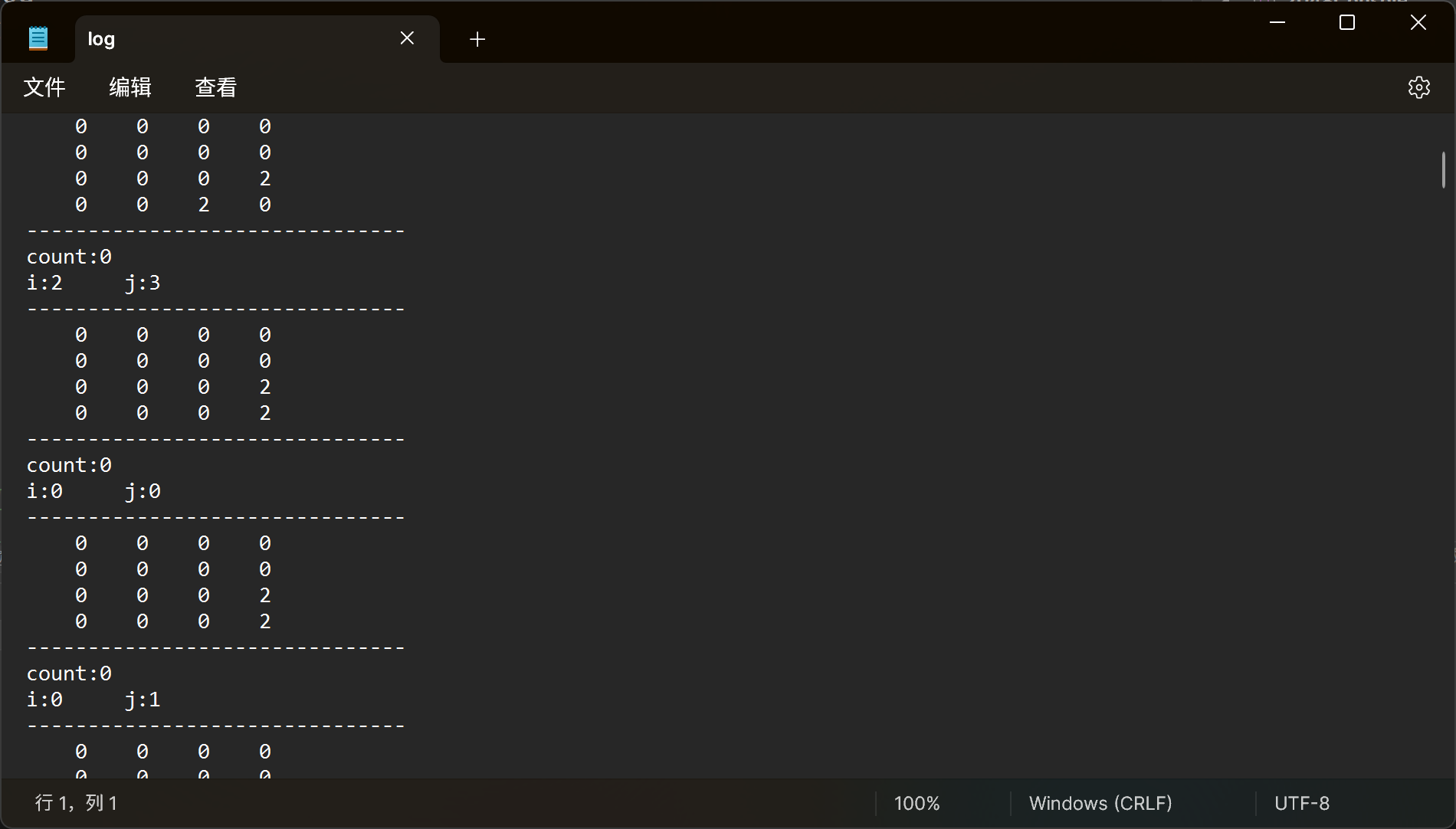


图2.5-log.txt的内容，上述LogMap函数在调试完这个bug被修改成图2.4所示更加泛用的函数

3 特色功能

3.1 使用fstream实现的排行榜功能

排行榜在每次程序开始的初始化过程中会从DoNotDelete.rank中读入，存入结构体数组struct Rank rank[10]中。而在每轮游戏结束后，会判断当前分数是否可以被列入排行榜，如果可以，就使用一种类似插入排序的算法，将它放入正确的排位并且返回排位值，最后UI输出。最后在程序正常退出时，会将此时的rank[]数组内的内容重新覆盖写入DoNotDelete.rank，从而实现了长期存储排行榜的功能。

而在这个过程中，fstream的文件流操作是核心。通过设置ifstream fin(…)与ofstream fout(…)可以实现独立于cout cin的UI读入读出的文件输入输出的操作。

3.2 趣味模式——盲盒模式

在盲盒模式中，玩家的每次合并操作产生的数不一定是原来的数\*2，而是可能是一个比合并数更大的2的倍数（最大256）。这个功能一方面可以加快游戏进程，使玩家在尽可能短的时间里得到较多的分；另一方面由于随机的存在，棋盘中数的值会较为分散，从而导致游戏难度隐藏地增加了。