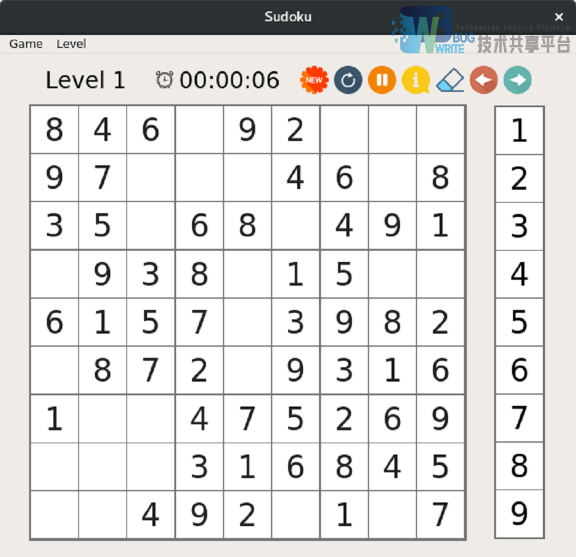
**实习二 基于回溯算法的数独游戏设计与实现**

**一、实习目的与要求**

**【问题描述】**

数独，是源自18世纪瑞士的一种数学游戏。是一种运用纸、笔进行演算的逻辑游戏。玩家需要根据9×9盘面上的已知数字，推理出所有剩余空格的数字，并满足每一行、每一列、每一个粗线宫（3\*3）内的数字均含1-9，不重复。本次实习要求设计并实现一款数独游戏软件（下图为9x9数独游戏示例）。



**【基本要求】**

1、采用回溯算法或舞蹈链（Dancing Link）实现数独游戏的求解；

2、采用控制台或GUI，实现语言不限；

3、设计并实现数独的基本功能，可根据个人情况对以下功能进行裁剪：

（1）新游戏：玩家可以开始一局新的游戏

（2）重玩：玩家可以重新开始本局游戏

（3）暂停：玩家可以暂停该局游戏（即暂停计时）

（4）提示：如果当前已经确定的数都是正确的，玩家将会得到一个未填空格的正确数字；如果当前已经确定的数和答案矛盾，导致整个数独无解，那么所有与答案矛盾的数字将会被粗体标出

（5）清除：清除当前选中格子的所有数字

（6）撤销：撤销前一步的操作，以及取消撤销（最多可支持 50 步撤销）

（7）同时可以通过菜单来实现多达 10 种难度的游戏选择，可以求解任意用户输入的数独问题。

**二、分析与设计**

1. **需求分析与类设计**

**本实习是用C++、easyX图形库编写的**

**需求分析：**

1. 自动生成数独：游戏开始的时候，自动生成地图；
2. 关卡难度选择：用户选择简单、一般、困难模式进行游戏，然后模式不一样生成不一样的地图；
3. 计时：游戏开始，开始计时，暂停、弹窗等的时候计时停止；
4. 高亮需求：当用户将鼠标放在数独格子中时，显示相同的数字高亮（绿色），显示需要避免重复的所有格子（灰色）；
5. 暂停：游戏中途可以暂停，此时计时停止；
6. 重玩此关：用户可以重新玩本关卡；
7. 新游戏：用户可以重新开始一个地图；
8. 自动求解：用户可以自动查看求解答案；
9. 最终得分：当用户完成之后，会根据用户选择的难度等级和用时进行评分；

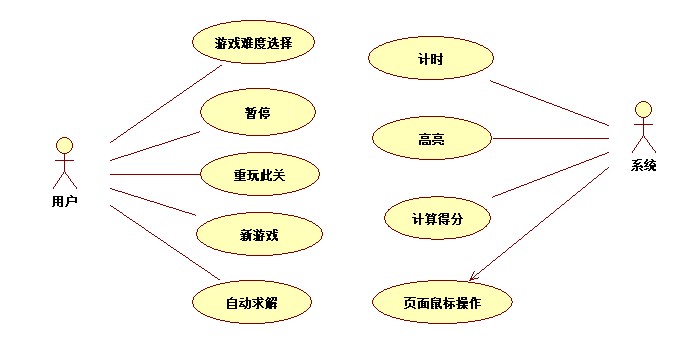


图1-用例图需求分析

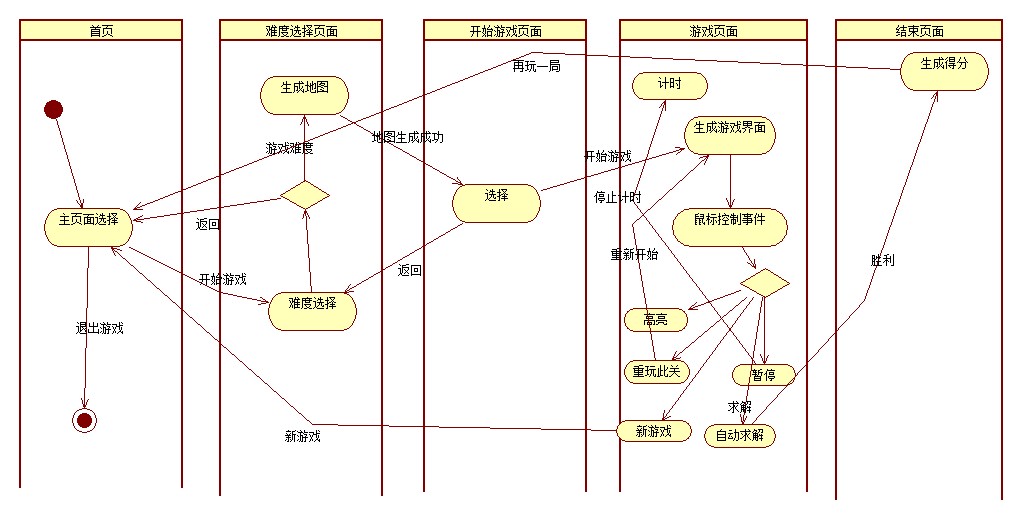


图2-活动图

**函数设计：**

* + - 1. 界面函数

1. 游戏开始界面

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void start1() | 显示首页，有开始游戏和退出游戏两个选择项，包括字体颜色、大小、格式等； |
| void select1() | 设置鼠标事件、捕获用户选择了哪个选项； |

1. 难度选择界面

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void start2() | 显示选择游戏难度页面，有简单、一般、困难、返回四个选择项，包括字体颜色、大小、格式等； |
| void select2() | 设置鼠标事件、捕获用户选择了哪个选项； |

1. 游戏正式开始界面

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void start3() | 显示选择页面，有开始游戏和返回两个选择项，包括字体颜色、大小、格式等； |
| void select3() | 设置鼠标事件、捕获用户选择了哪个选项； |

1. 游戏主页面

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void GameDraw() | 对页面进行绘画和数据的显示：   1. 页面背景、字体、颜色、格式、位置等信息 2. 画九宫格的框线 3. 先高亮后outtextxy数字，否则数字会被覆盖 ，高亮提示数字 4. 画宫格数字 5. 画数字键盘内外框 6. 画数字键盘数字 & 计时器 7. 用红框标注数字键盘中的当前所选数字 8. 功能显示：暂停、重玩此关、自动求解、新游戏 |
| void GameUpdate() | 设置鼠标事件，响应对应的数字和高亮效果，和功能选择；游戏主要函数； |

1. 游戏结束页面

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void end() | 显示游戏胜利页面，计算游戏得分，提示再来一局或者退出游戏； |

* + - 1. 核心算法函数

1. 自动生成数独地图函数

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void GameInit(int post) | 根据post选择的难度，生成不同的数独地图，最终生成两个二维数组，一个用来存地图，空的地方用0表示，一个用来存哪些地方不是空的，两个地图互相对应，对后面的高亮显示做准备； |
| void Dig(int post) | 对生成的满地图进行挖空，同时也需要结合post进行挖空的多少，从而进行实现难度的不同； |

1. 检测错误函数

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| int FindMistake(int mode) | 检测当前地图中是否有错误情况，如果发现存在一行或者一列或者一宫中有相同的数字，则将该数字显示为红色高亮，并且记录下错误次数。 |

1. 检测胜利函数

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| int WinJudge() | 判断当前用户是否完成该数独游戏，如果完成，则进行检查是否正确，如果正确则进入胜利界面。 |

1. 高亮函数

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void HeighLight(int post) | 检测高亮函数，根据用户鼠标事件的范围，对界面颜色和高亮进行更新。如果鼠标放在空的格子中，则用灰色高亮显示所在行、列、宫；如果鼠标放在有数字格子中，则用绿色高亮显示当前地图中所有的相同数字； |
| COORD FindLeftUpperCornor(COORD c1) | 该函数是对鼠标所在游戏界面像素位置的一个解析。 |

* + - 1. 回溯自动求解函数

1. 回溯求解函数

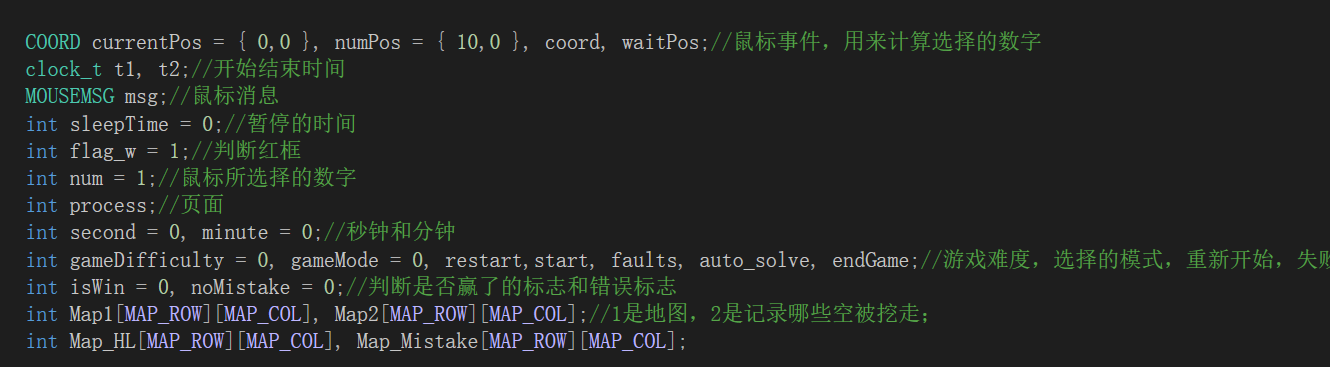
|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| void solve\_soduku(int x[MAP\_ROW][MAP\_COL]) | 传入一个用0表示空的二维数组，通过暴力遍历的方法查找合适的解。 |

1. 判断是否合适函数

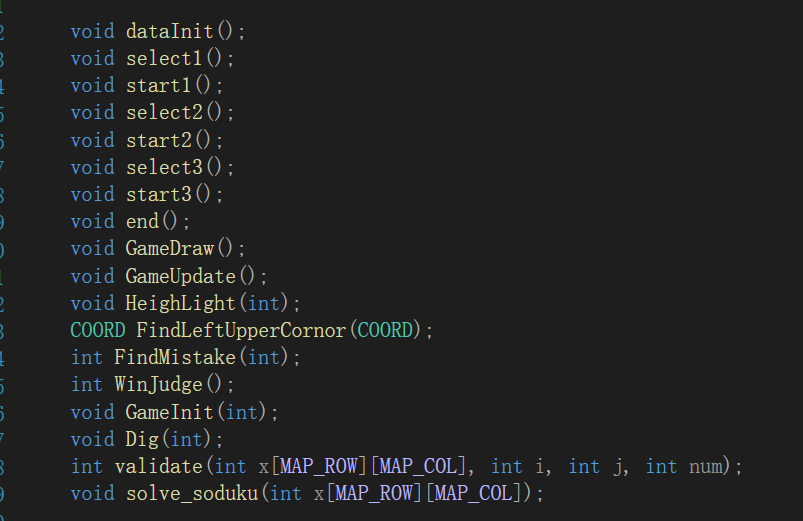
|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| int validate(int x[MAP\_ROW][MAP\_COL], int i, int j, int num) | 判断当前填入的数字是否满足行、列、宫都没有重复的要求； |

1. **算法设计与分析**

全局变量：



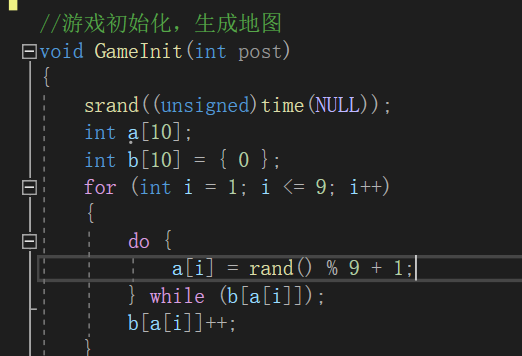
函数：



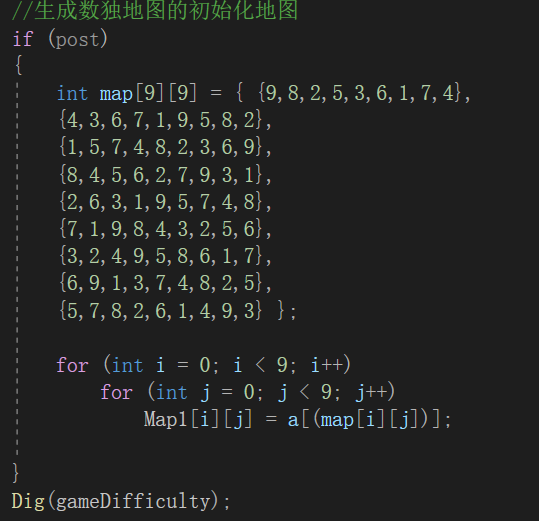
主要函数算法思路：

1. 自动生成地图：

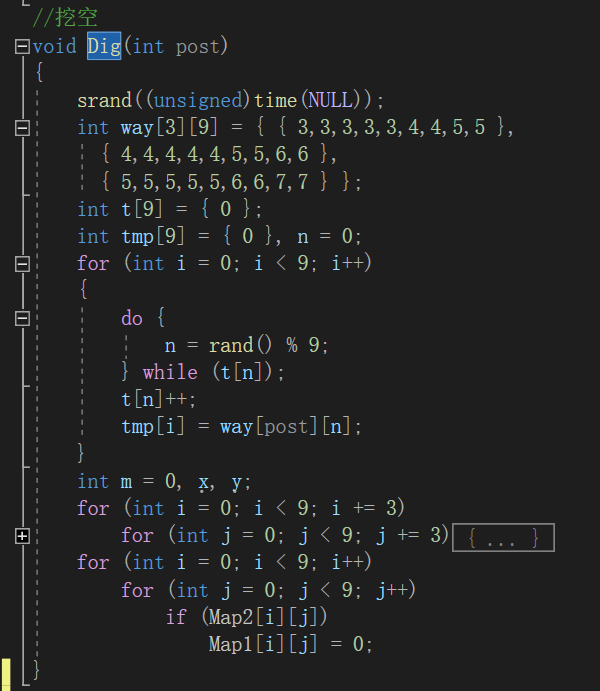
首先通过随机数的方法生成两个一维数组，两个数组一一对应着，当a数组不为空的时候，b数组值为1；



现在两个数组a和b已经生成了对应关系了(很乱，但一一对应)，然后根据提供的数组生成数独全地图；



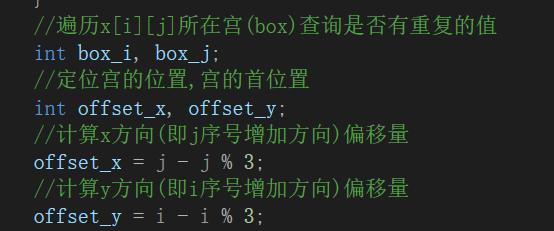
之后再将全地图随机挖空，空的地方用0代替，根据post的等级进行挖空的多少。



通过自动生成函数，生成了两个二维数组，Map1存真实的地图数据，Map2存对应于空的数据；

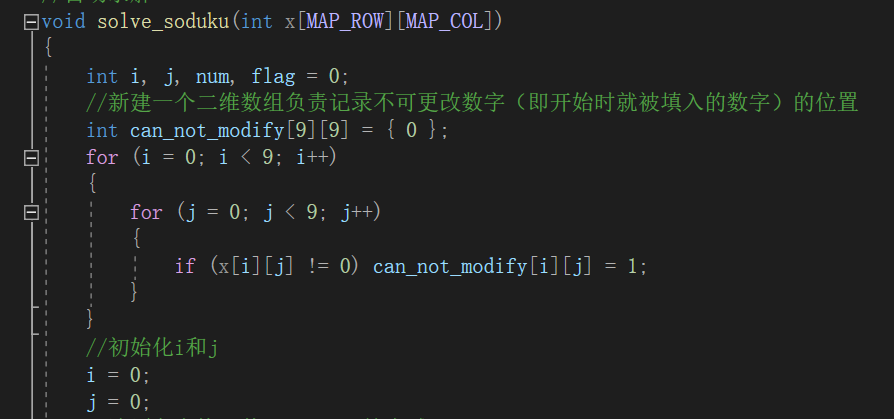
1. 自动求解：

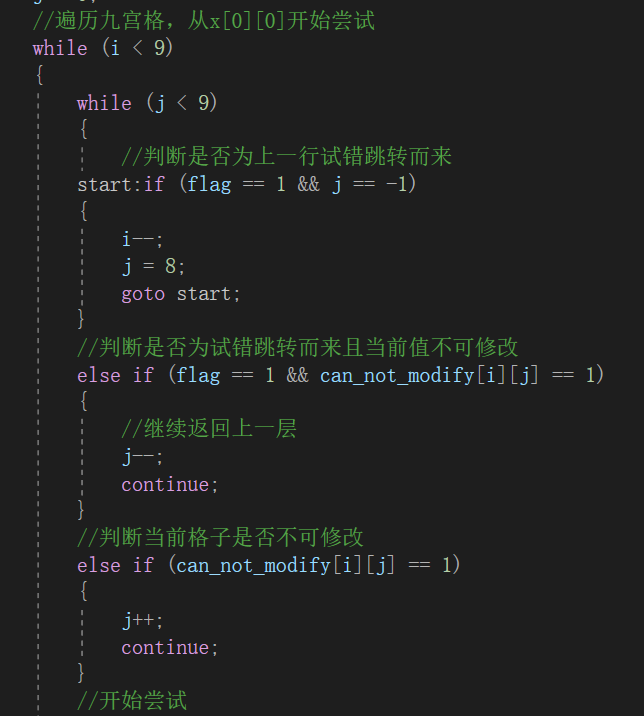
首先设置一个flag判断是否需要回溯，新建一个二维数组can\_not\_modify负责记录不可更改数字（即开始时就被填入的数字）的位置，在开始尝试之前要判断三个条件：判断是否为上一行试错跳转而来、判断是否为试错跳转而来且当前值不可修改、判断当前格子是否不可修改，如果这三个条件都通过了之后，就可以开始尝试了。从1开始遍历，对每个num进行正确性判断，调用validate(x, i, j, num)。具体就是对行、列、宫进行遍历，其中值得注意的是宫的变量，需要确定其对应数组的位置

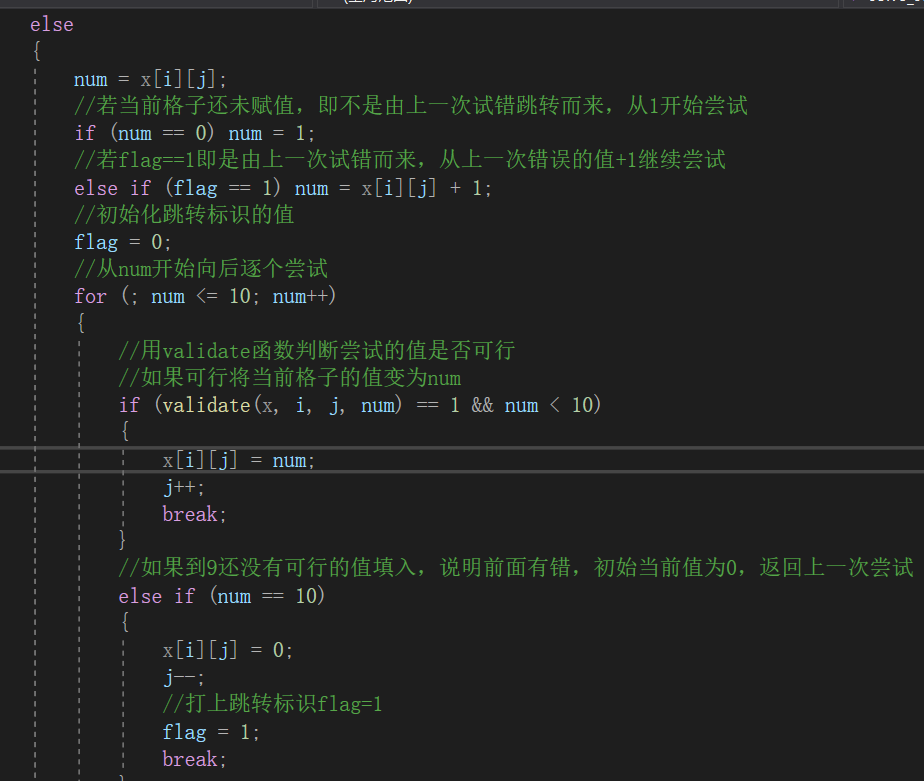


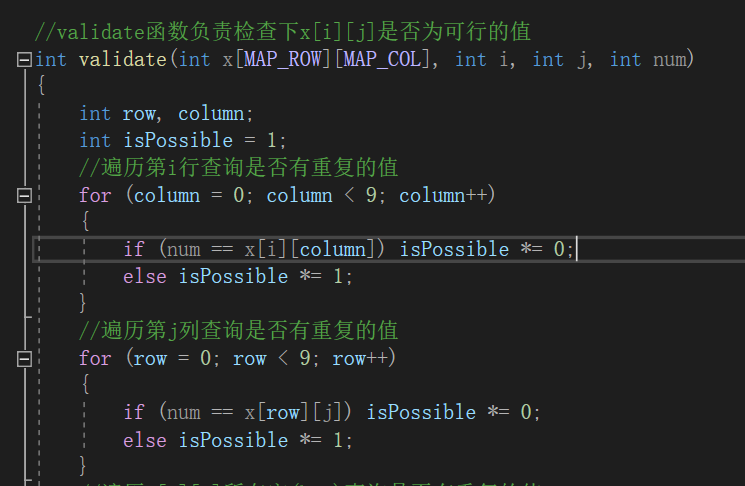
如果满足三个条件就暂时确定为该数据为正确数据，进行下一个空的遍历。直到有一个空白空从1到9都没有合适的数据，则需要回溯，回到上一个空的地方进行往下继续遍历，这样往前遍历，直到最后遍历出一个解则结束；

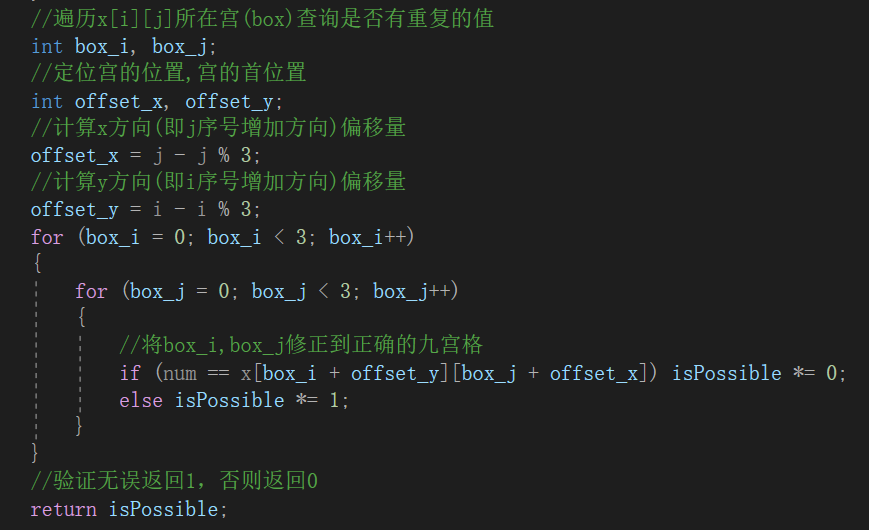
具体代码如下：







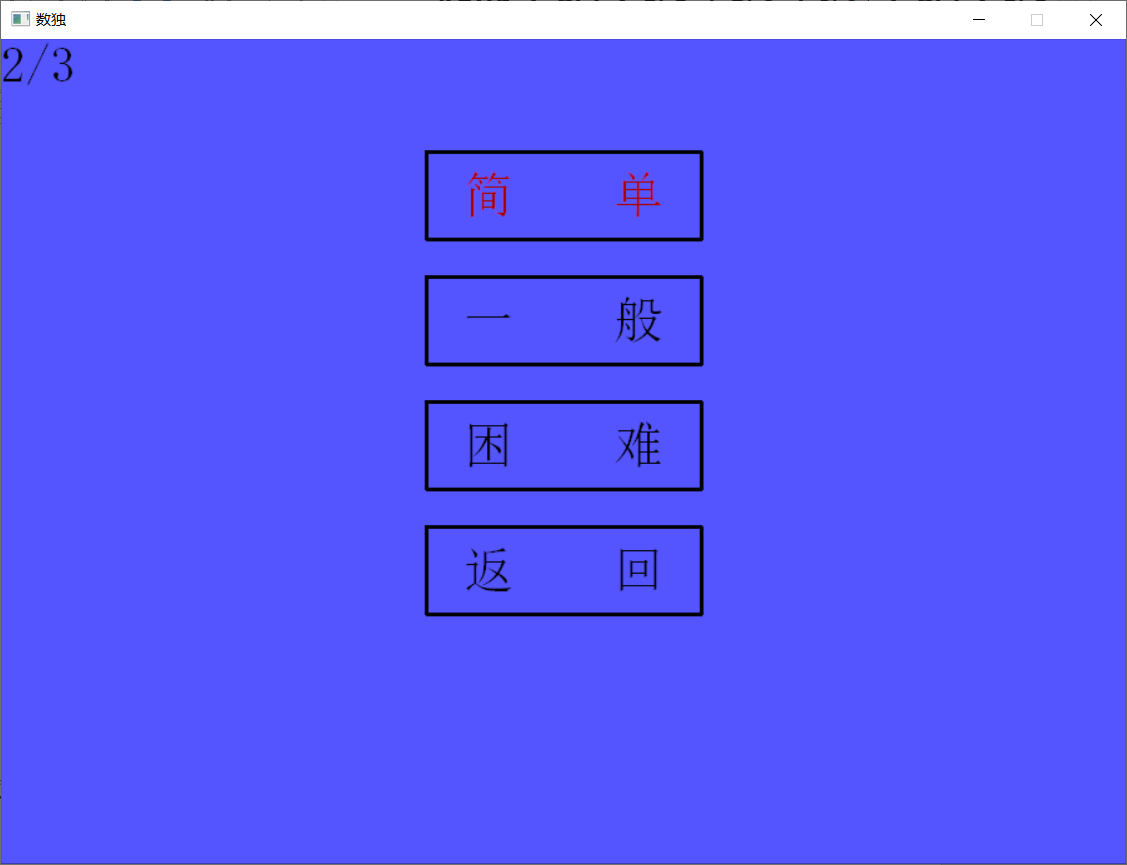




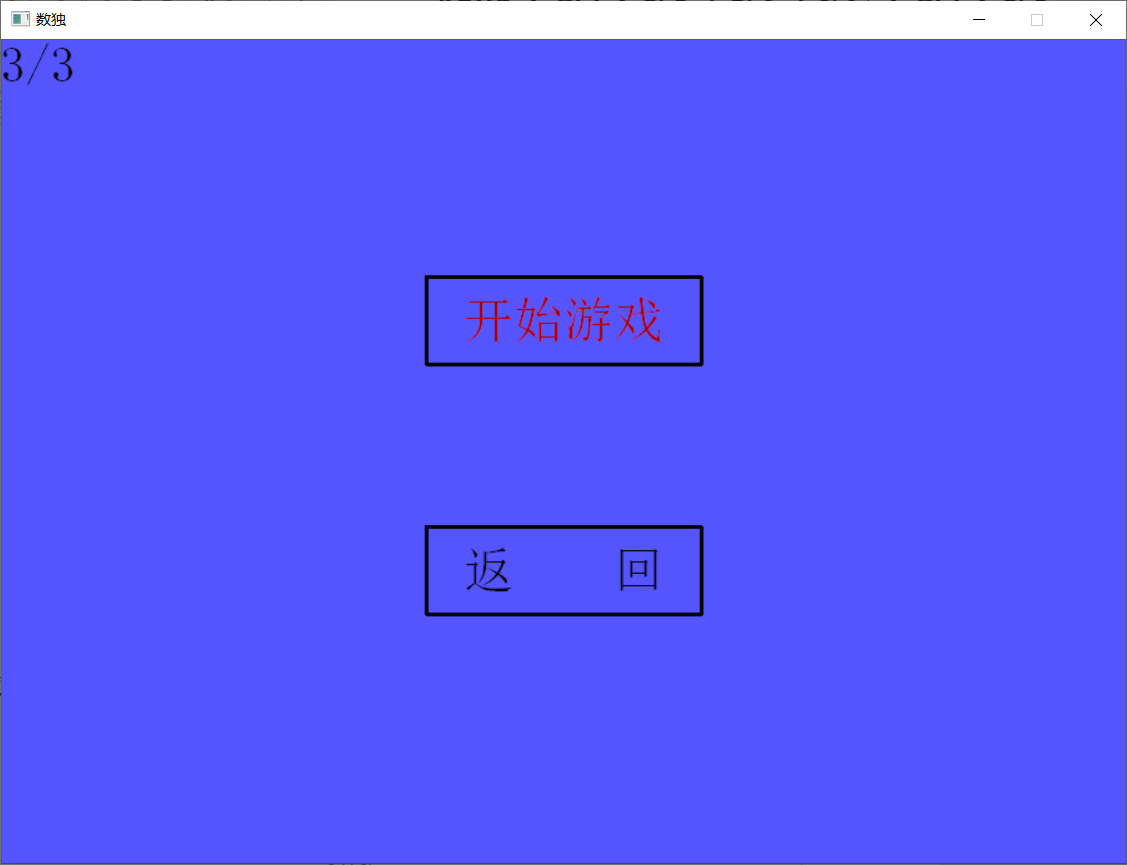
1. **测试与改进**
2. **主页面**



1. **难度选择页面**

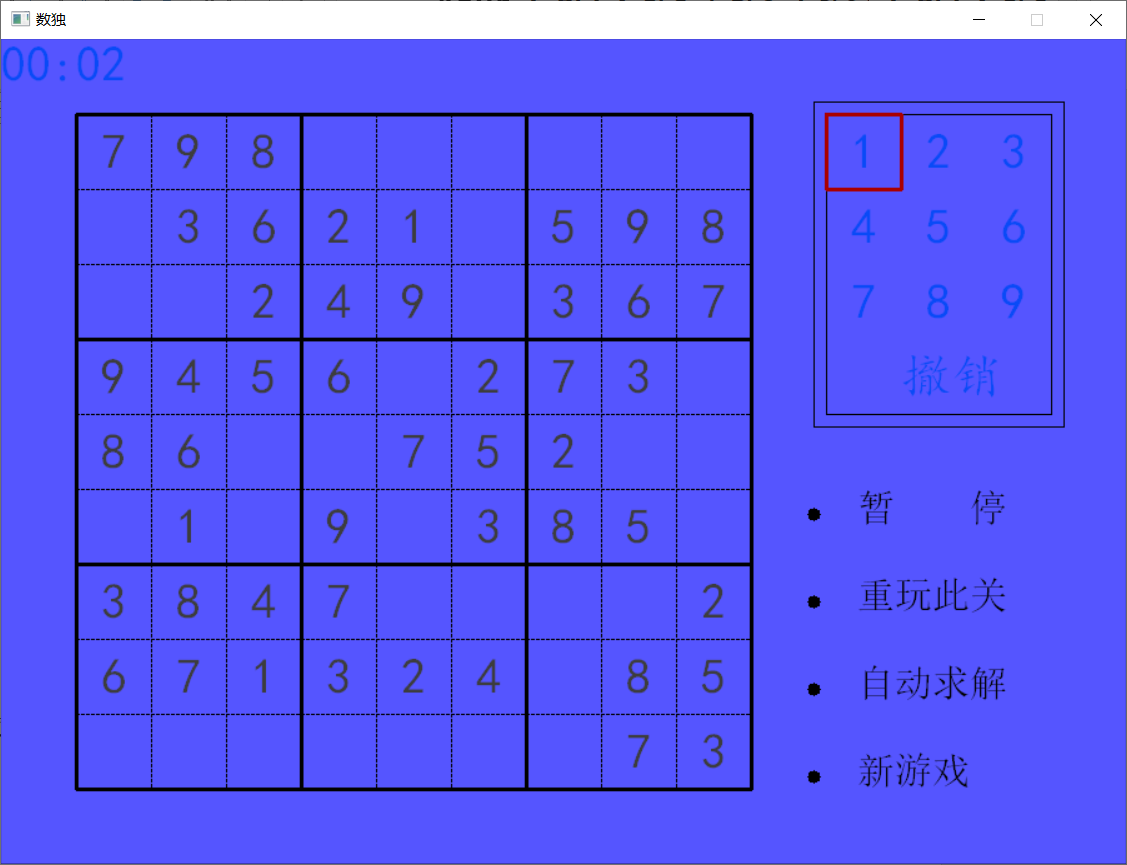


1. **游戏开始页面**

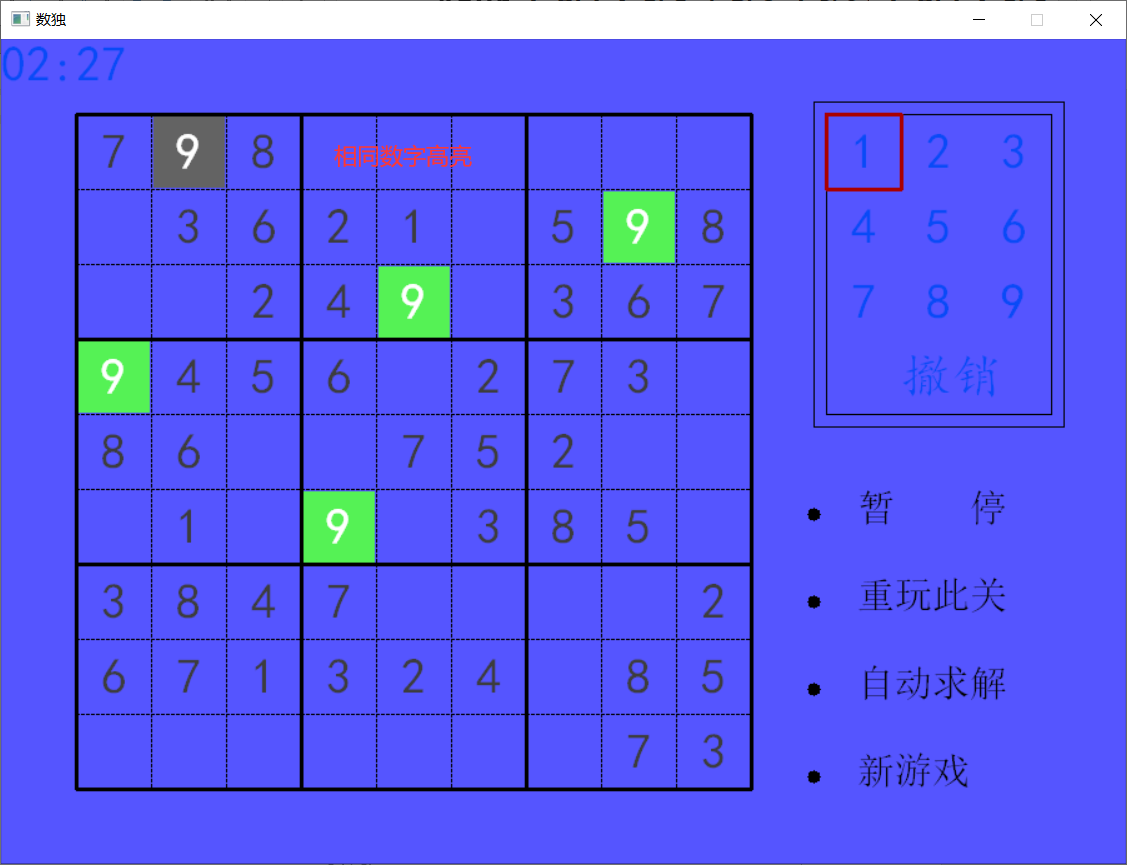


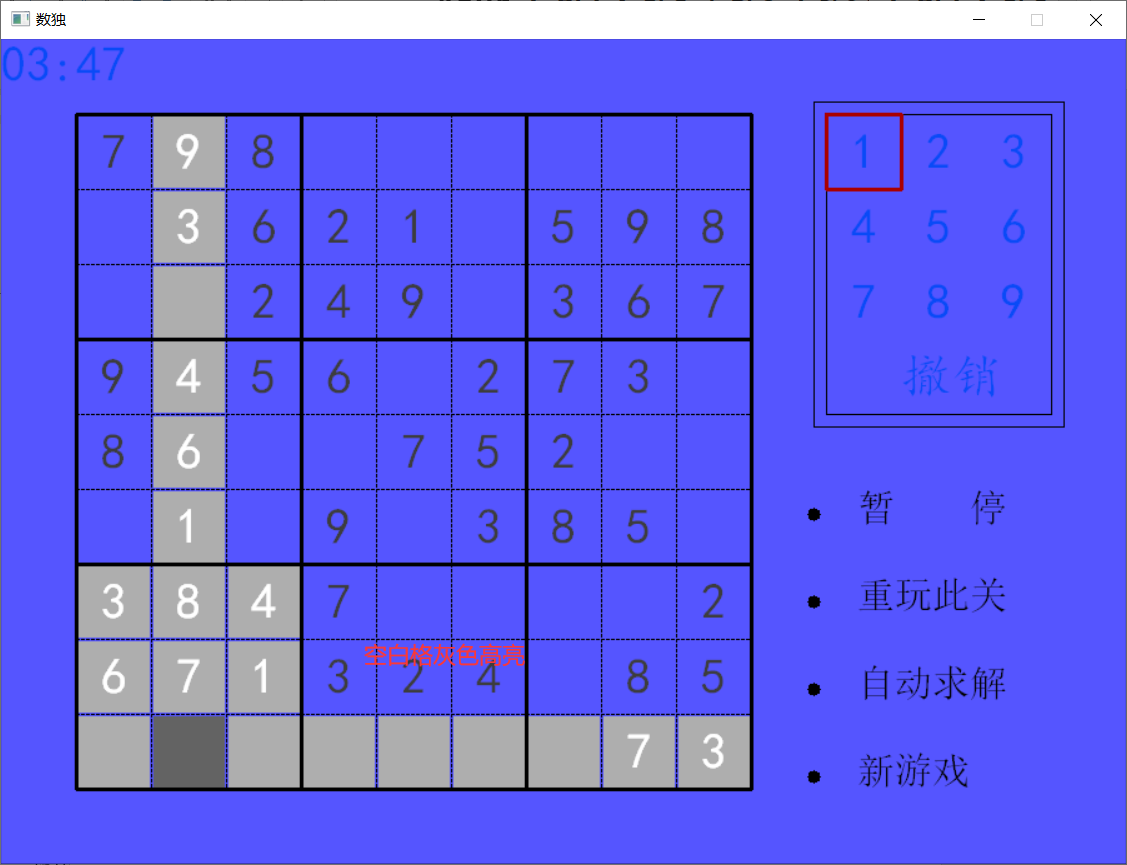
1. **游戏页面**

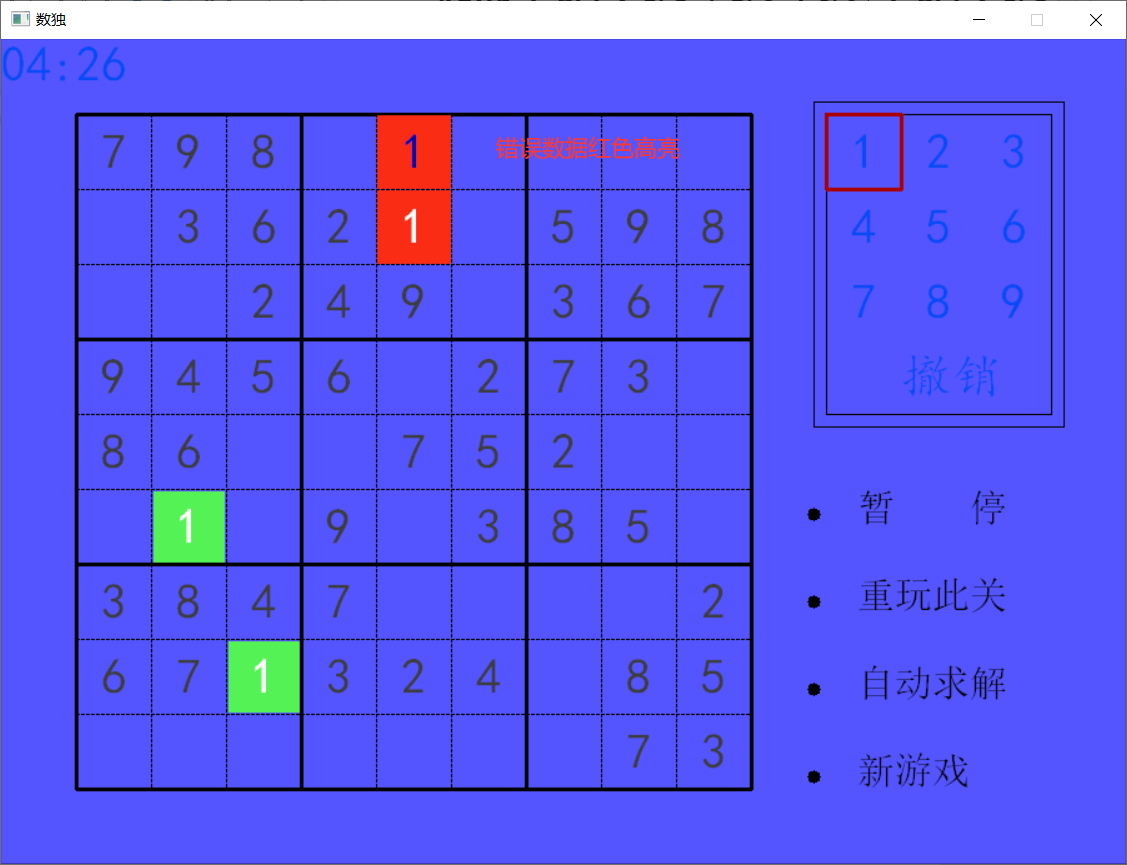
1.静态页面



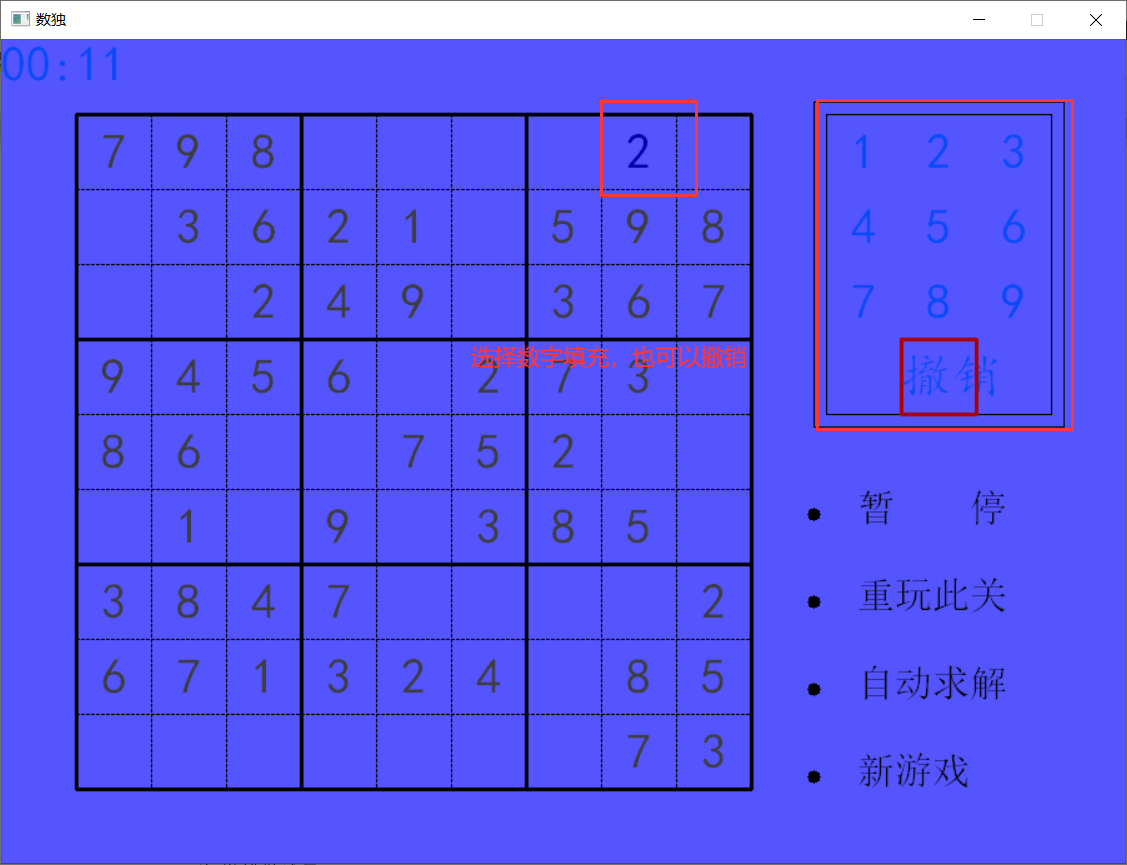
2.高亮测试



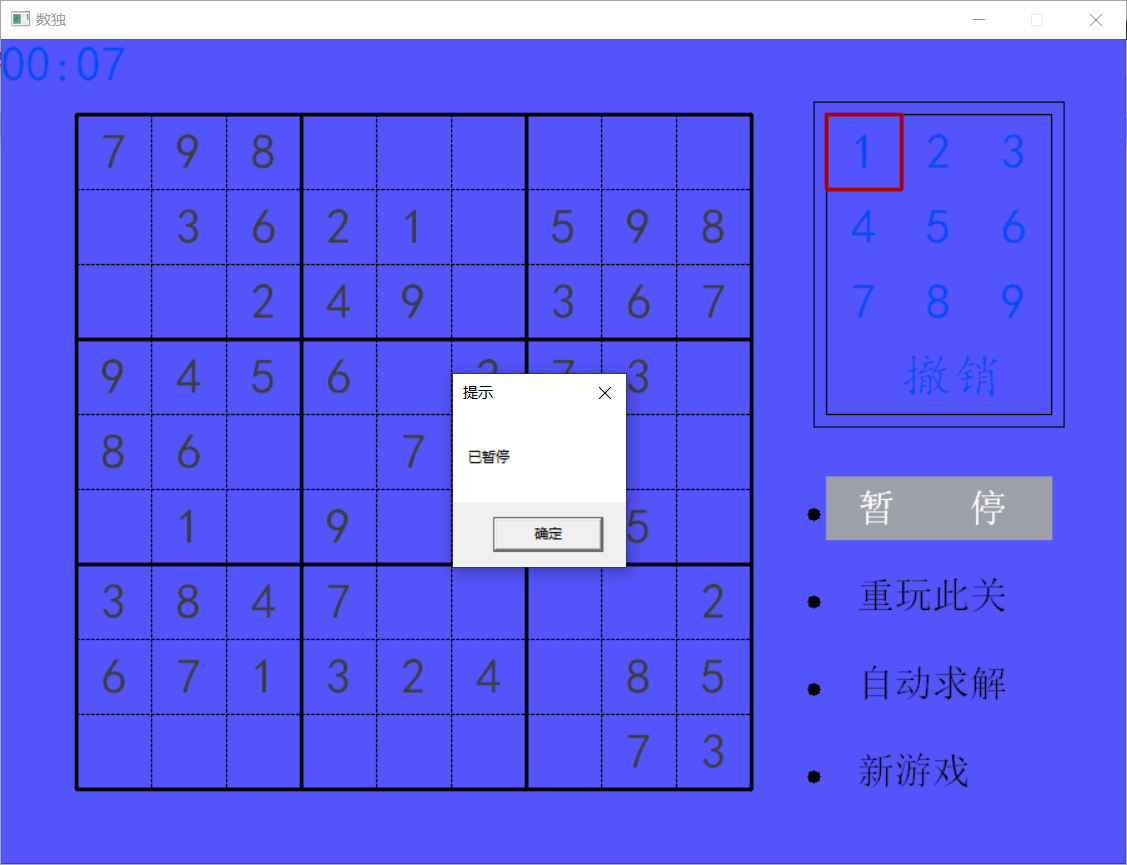




3.游戏测试



4.暂停



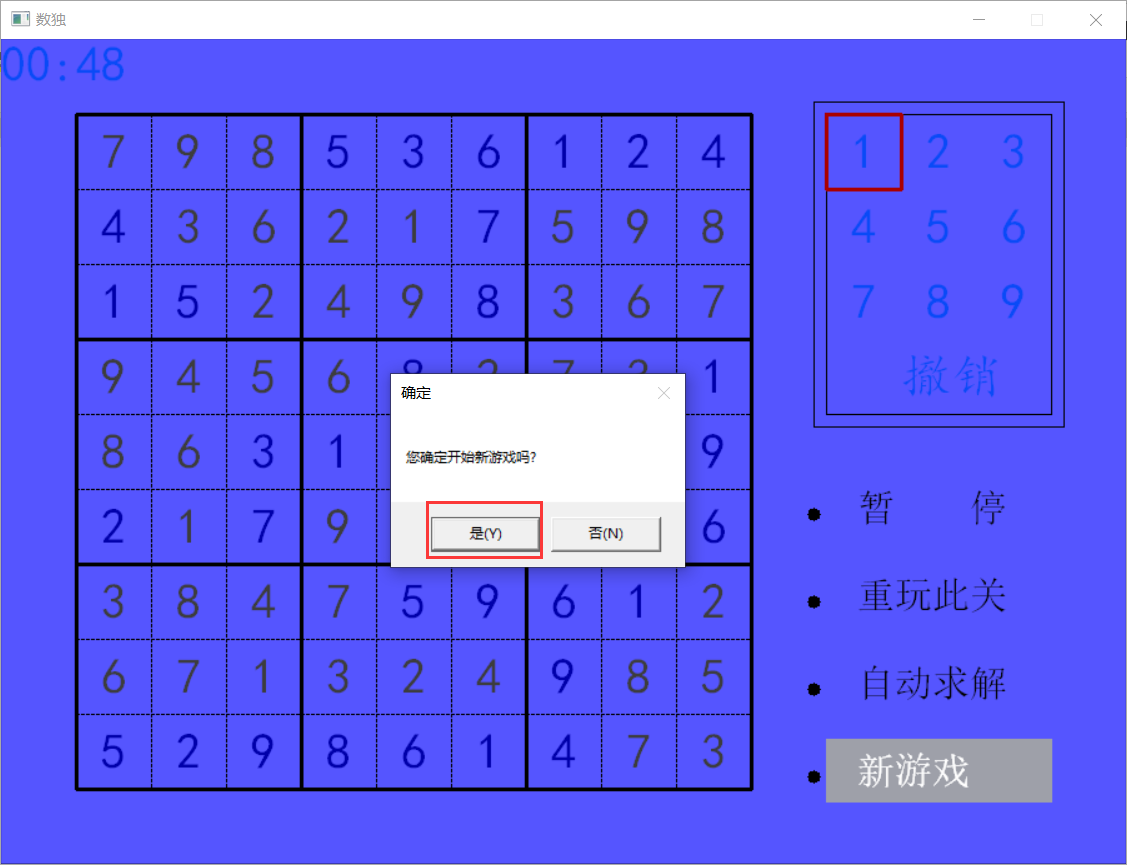
5.重玩此关



6.自动求解



7.新游戏



1. **胜利页面**



**三、实习小结**

通过本次是实习对回溯算法有了一定的掌握。可视化方面主要是使用easyX，所以对该图形库有了进一步的认识，学会了如何控制鼠标事件、计时、页面布置等ui知识；

**四、课程学习总结（个人收获与教学建议）**

学习完本课程之后，对算法有了进一步认识，了解到算法主要分为递归、分治、动态规划、贪心、回溯分支、随机化等算法。虽然没有对所有的算法都有清晰的理解掌握，但是还是大部分都了解到各算法的思路和含义作用。

-------------------------------

成绩评定：

教师签名：

批改日期：