**数据结构课程设计报告**

**设计题目：**老鼠走迷宫

**学生姓名：**SandyZheng

**专业：**

**班级：**

**学号：**

**指导教师：**

**完成日期：**2022.7.7

**目录**

[I](#_Toc13263) **[需求和规格说明](#_Toc13263)** [2](#_Toc13263)

[II](#_Toc8484) **[设计](#_Toc8484)** [3](#_Toc8484)

[i 设计思想 3](#_Toc25848)

[ii 设计表示 3](#_Toc9204)

[iii 核心代码 4](#_Toc32199)

[III](#_Toc18788) **[用户手册 7](#_Toc18788)**

[IV](#_Toc17147) **[调试及测试 8](#_Toc17147)**

[i 游戏流程展示 8](#_Toc18910)

[ii 仍需的改动 1](#_Toc32622)4

[V](#_Toc20610) **[感想 1](#_Toc20610)5**

# 需求和规格说明

**问题描述：**给定一个迷宫地图，迷宫中央有一只老鼠，迷宫的右下方有一个

粮仓。游戏的任务是使用键盘上的方向键操纵老鼠在规定的时间内走到粮仓处。

**编程任务：**

1. 老鼠形象可辨认，可用键盘操纵老鼠上下左右移动；
2. 迷宫的墙足够结实，老鼠不能穿墙而过；
3. 正确检测结果，若老鼠在规定时间内走到粮仓处，提示成功，否则提示失败；
4. 添加编辑迷宫功能，可修改当前迷宫，修改内容：墙变路、路变墙；
5. 找出走出迷宫的所有路径，以及最短路径；
6. 利用序列化功能实现迷宫地图文件的存盘和读出等功能。

注：为了方便设计迷宫，笔者将老鼠的出发点设为迷宫左上角。

# 设计

1. **设计思想**

本题考察点：

1. 用二维数组表示迷宫：1代表墙，0代表路；老鼠不能穿墙即遇到值为1的方块无法移动。
2. 题目要求寻找路径于是联想到深度优先搜索（DFS），又由于需保存最短路径并输出，故考虑用ArrayList保存所有路径方便后续筛选。
3. 倒计时的实现：考虑将IDEA自带的Timer类进行微调；
4. 存盘与读出：迷宫会以txt文件的形式存储，将二维数组转换为String类型，通过fstream类的函数进行读写操作。
5. 界面制作与交互设计：需制作功能清晰的面板，便于玩家进行各种游戏操作。
6. **设计表示**

**表1 类**

|  |  |  |
| --- | --- | --- |
| **类名** | **中文解释** | **功能** |
| Maze | 迷宫容器 | 容纳EnterPanel |
| EnterPanel | 进入游戏面板（含主函数） | 选择游戏模式（新游戏/读取存档/编辑模式/退出） |
| ChooseEditPanel | 编辑选项面板 | 选择想要编辑的地图 |
| EditPanel | 编辑面板 | 编辑地图 |
| MainPanel | 主面板（游戏面板） | 操控老鼠移动通关/存档/寻找路径 |
| MyMapPanel | 自创地图面板 | 读取并显示自制地图 |

**表2 MainPanel里的函数**

|  |  |  |
| --- | --- | --- |
| **数据类型** | **函数/类名称** | **作用** |
| 无 | MainPanel() | 构造函数，初始化主面板 |
| void | addButton() | 添加按钮及监听事件 |
| void | dfs(int[][] maze, int i, int j, int n, int m, ArrayList<Integer> step) | 深度优先搜索，寻找所有路径 |
| void | replay() | 本关失败选择复活后重来 |
| void | next() | 本关成功后进入下一关 |
| void | loadImage() | 载入程序所需要的所有图像 |
| void | drawMap(Graphics g), drawRole(Graphics g), drawCheese(Graphics g), drawShortCut(Graphics g) | 在指定位置画出图像 |
| void | paintComponent(Graphics g) | 在JPanel基础上调用上述四个方法绘制图像 |
| void | keyPressed(KeyEvent e) | 处理角色移动事件（例如碰到墙不能移动） |
| boolean | isAllow() | 判定是否允许移动的发生，被move()函数调用 |
| void | move(int event) | 判断移动事件，关联isAllow()方法 |
| 无（内部类） | TimeListener | 定时器（倒计时30s） |

**表3 EditPanel和MyMapPanel里的函数**

|  |  |  |
| --- | --- | --- |
| **数据类型** | **函数名称** | **作用** |
| int[][] | getSparseArrayFromFile() | 从txt中读出二维数组 |
| void | intDataToFileOut(int[][] sparseArray) | 将二维数组存进txt |
| void | mousePressed(MouseEvent e) | 实现鼠标点击处墙变路、路变墙 |

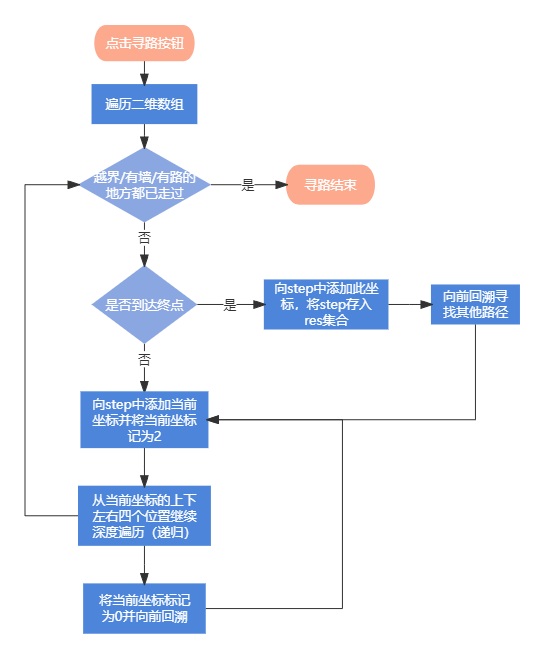
注：进入面板（EnterPanel）与编辑选项面板（ChooseEditPanel）所含函数与上述面板基本重复，故不再列举。

1. **核心代码**

该程序的核心算法主要放在函数dfs(int[][] maze, int i, int j, int n, int m, ArrayList<Integer> step)、isAllow(int x, int y)中，其中dfs（即找路函数）最为重要。

1. 深度优先搜索——dfs(int[][] maze, int i, int j, int n, int m, ArrayList<Integer> step)：
2. 第一步：规定结束（返回）条件：越界（i<=0 || i>=n || j<=m || j>=m）、有墙（maze[i][j]==1）、已经走过（maze[i][j]==2）。
3. 第二步（重点）：从起始点遍历二维数组：
4. 若已到达终点，则先向存储路径的存储整型的集合 step中添加终点坐标，并将step存入存储ArrayList的集合res中；再从step中删除这个坐标，回溯到终点的前一个坐标，按未到达终点的情况进行处理，即寻找是否还有其他路径；
5. 若未达到终点，则先向存储路径的存储整型的集合 step中添加终点坐标，并将当前坐标标记为2（已走过）；接着继续从当前坐标的上下左右四个位置开始深度遍历（即递归）；递归结束之后将当前坐标标记为0（未走过），删除step中存储的此坐标，回溯到此坐标的前一个坐标，寻找是否还有其他路径。
6. 第三步：从res集合中通过比较筛选出最短路径并输出（红色）；接着输出res集合中的所有其他路径（蓝色）。此处筛选最短路径的方法为：依次比较res集合中每个ArrayList的长度，用index定位长度最小的ArrayList，即res(index)为最终的最短路径。

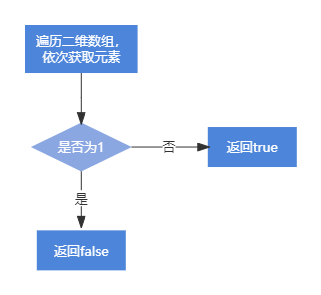
流程图如下：



1. 判断老鼠是否能移动——isAllow(int x, int y)：
2. 第一步：遍历二维数组，获取每个元素的值；
3. 第二步：若值为1，则说明此处是墙，老鼠不可走到这个位置，返回false；若值为0，则说明此处是路，老鼠可通过，返回true。

此方法需配合move()函数使用，其中isAllow(int x, int y)控制老鼠是否能走到当前位置，如果返回值为true，则move()控制老鼠坐标变化（x±1或y±1）；如果返回值为false，则move()控制老鼠坐标不变，即停在原地，不可穿墙而过。

流程图如下：



注：move()函数功能：

1. 获取键盘事件，LEFT/RIGHT/UP/DOWN分别对应上下左右键；
2. 以LEFT为例，若isAllow(x-1, y)为真，则x=x-1；若isAllow(x-1, y)为假，则不改变x值。

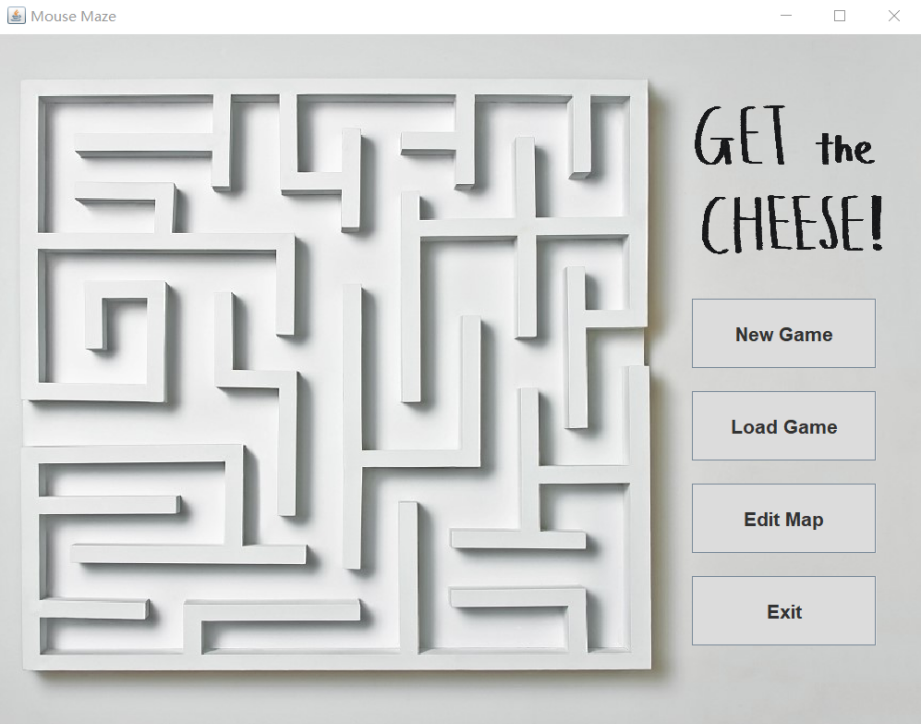
# 用户手册

程序运行后，显示进入游戏界面。此时有四个选项：新游戏（New Game）、读取存档（Load Game）、编辑地图（Edit Map）、退出（Exit）。

1. 点击New Game，游戏加载第一关，面板上方的倒计时开始运行。
2. 若老鼠在规定时间内吃到奶酪，将显示弹窗告知玩家已通关，是否进入下一关。若玩家选择“是”，地图将刷新；若玩家选择“否”，将返回进入游戏的面板。
3. 若老鼠未在规定时间内吃到奶酪，将显示弹窗告知玩家通关失败，是否复活。若玩家选择“是”，则老鼠回到起点，倒计时重新开始；若选择“否”，则返回进入游戏的面板。
4. 游戏中，玩家可查看最短路径和所有路径，只需点击“Routes(shortest red)”按钮即可。红色为最短路径，蓝色为平民路径。值得一提的是，在查看路径之后，倒计时停止，玩家即使操控老鼠到达终点也不会有提示通关，因为查看路径相当于作弊。此时若想进入下一关，只需点击“→”按钮即可。
5. 点击Load Game，将有弹窗提示玩家输入存盘名字。输入正确存盘名之后，程序将读取并显示玩家自制的地图，并开启倒计时。
6. 点击Edit Map，将有四个选择——Map 1，Map 2，Map3，Blank Map（空白地图），玩家需选择一个地图进行编辑。除了起点、终点与四周的墙之外，地图可通过点击完成墙变路，路变墙操作。编辑结束后，点击“Save you map!”按钮，将有弹窗提示玩家输入存盘名字，按照指示输入并点击确认即存盘成功。
7. 点击Exit，退出游戏。
8. 所有界面都有return按钮，方便玩家返回上一界面。

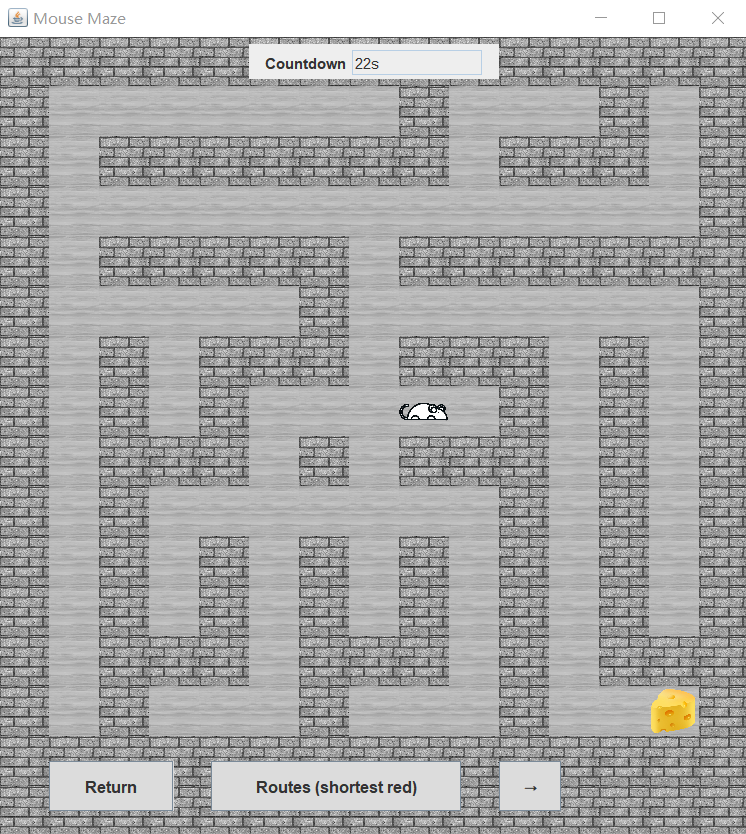
# 调试及测试

1. **游戏流程展示：**



**图1 “进入游戏”界面**

选择New Game后进入第一关，倒计时开始。



**图2 游戏中界面（第一关）**

****

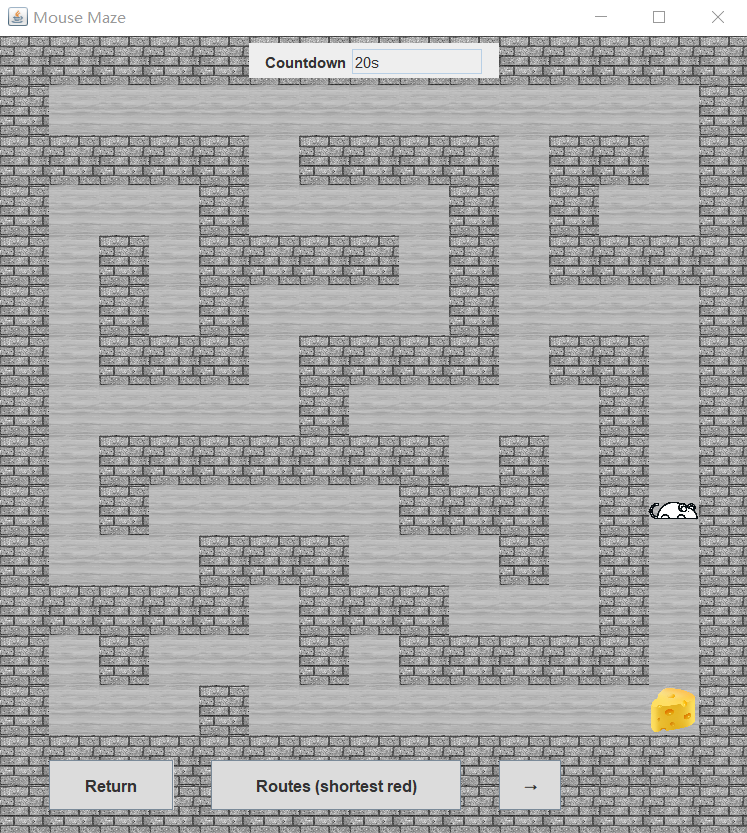
**图3 游戏成功**

若在规定时间内到达终点，则可进入第二关。

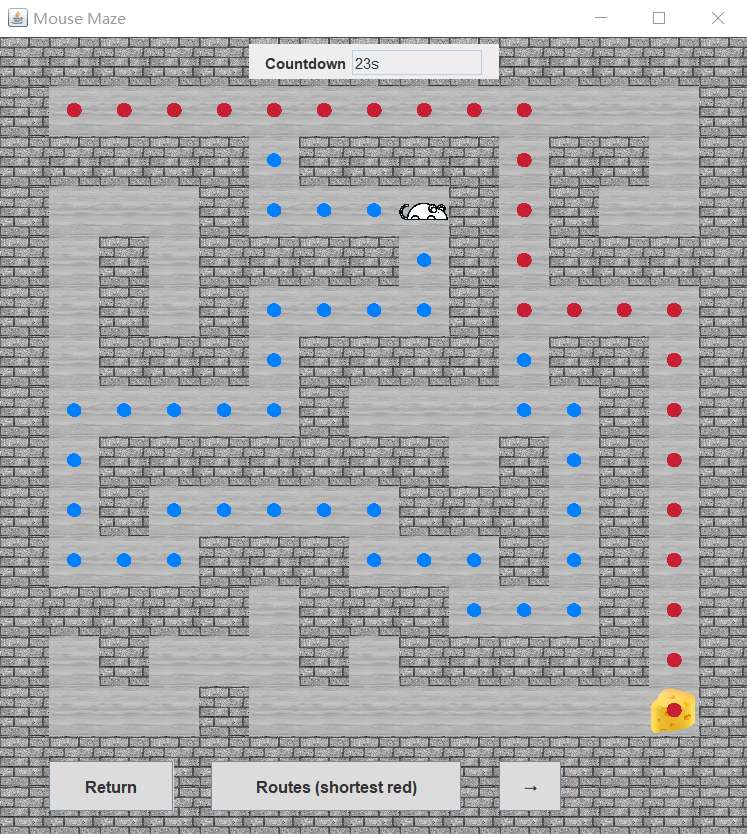
****

**图4 游戏失败**

若未在规定时间内到达终点，小老鼠将停止移动，弹窗提示是否复活。

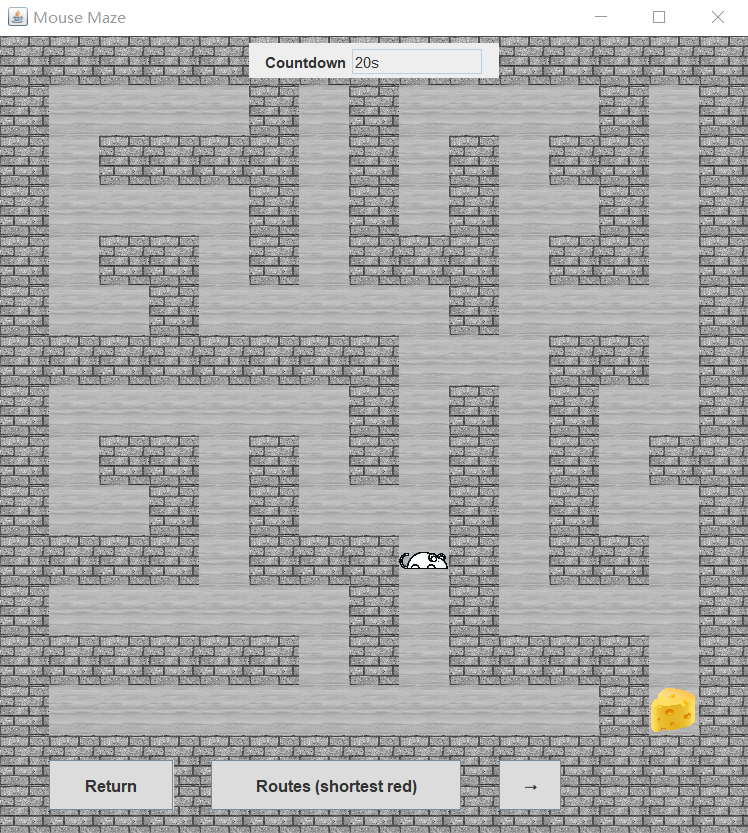


**图5 第二关**



**图6 寻找路径（第二关两条路）**

点击寻路按钮，红色为最短路径，蓝色为其他路径。

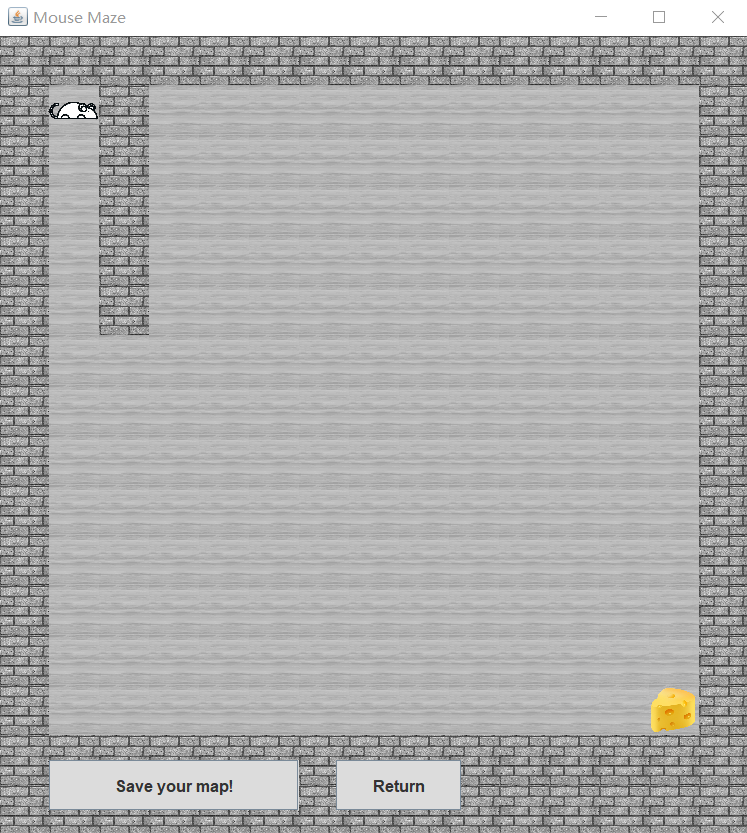


**图7 第三关**



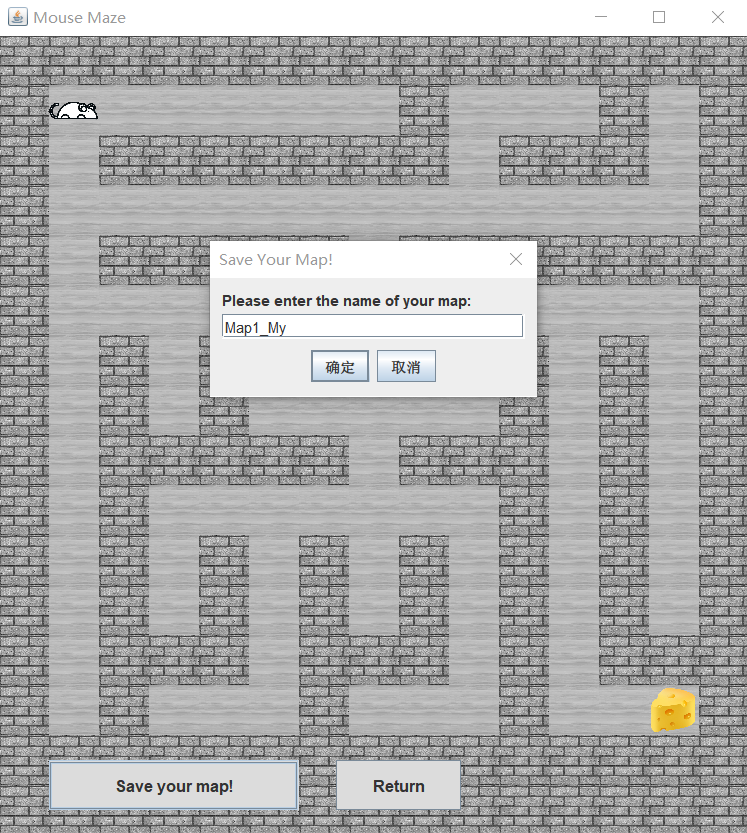
**图8 选择地图编辑**

有四个地图可供编辑：第一关、第二关、第三关、空白地图。



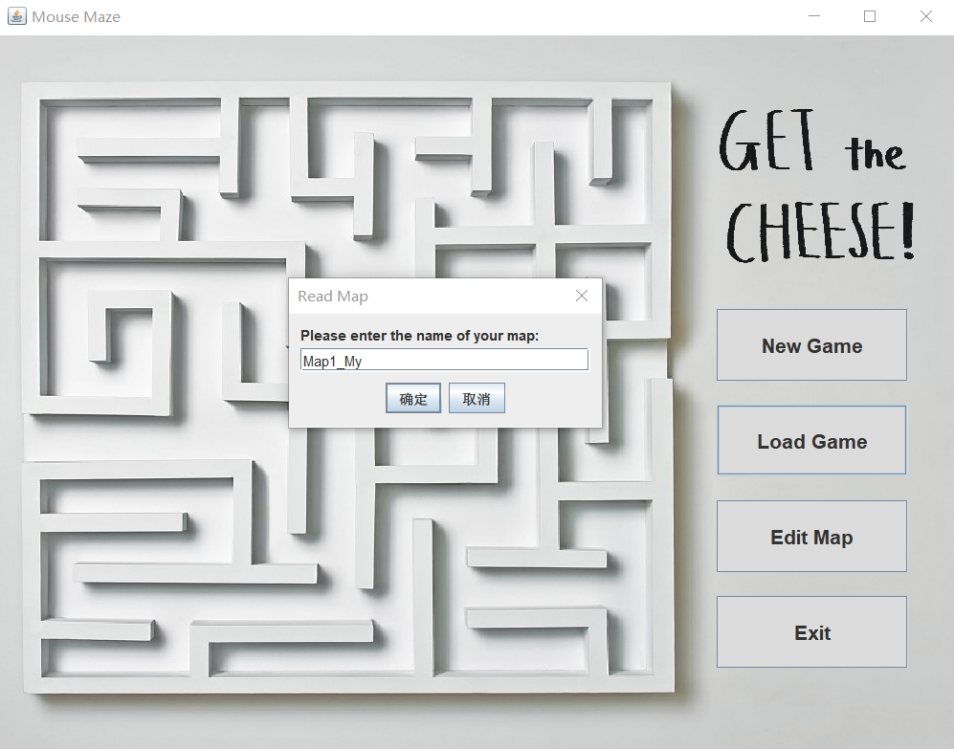
**图9 编辑地图（空白）**

通过单击方块可实现墙变路、路变墙。其中起点、终点、四周墙壁不可改变。



**图10 存盘（存入txt文件）**

点击Save your map!后输入存档名，地图便自动存入相应txt中。



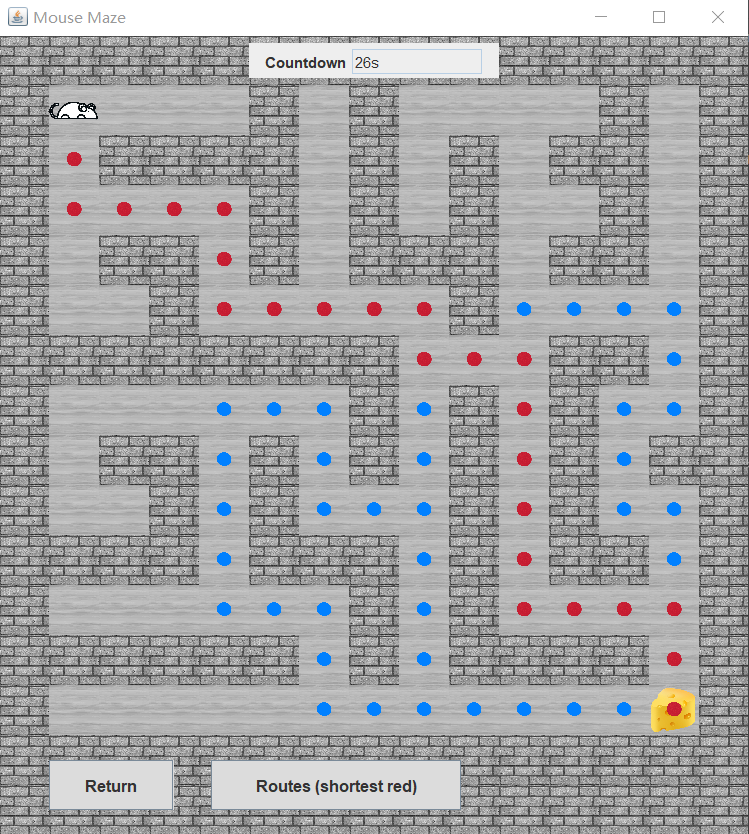
**图11 读取存盘**

点击Load Game输入存档名，程序自动读取txt文件中的二维数组。

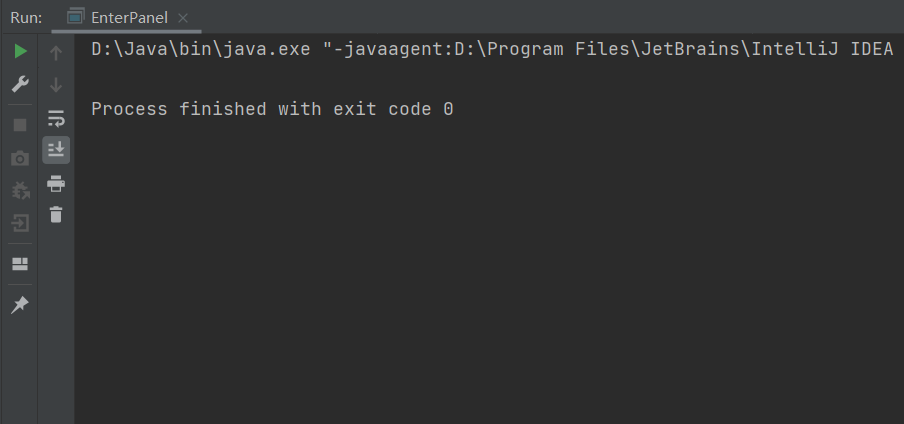
****

**图12 自制地图通关**

倒计时重新启动，玩家可体验自制地图。

****

**图13 自制地图寻路**

****

**图14 退出游戏（Exit）**

1. **仍需的改动：**
2. 若用户输入存盘名字有误，找不到存盘，应当给予弹窗警告或提示。目前只能程序内报错，弹窗交互尚存在问题；
3. 若从起点到终点无路径，应给予弹窗警告，目前只能程序内报错。即缺少判断有无路径的函数；
4. 由于时间有限，笔者暂时只设计了三关地图，且无进度存档，游戏略显单薄。若有时间进一步完善，将考虑添加进度存档功能&增加关卡数&提升关卡难度；
5. 地图是自己设计的而无法自动生成。

# 感想

本题考察的知识较为综合，囊括了界面交互、java插件的使用、深度优先搜索、io流应用等，费时费力，且仍有一些界面的小bug笔者至今也未修复好。尽管如此，笔者在一个月的反复实践后学到了许多非常有用的知识，对界面的操作已经十分熟悉，还真正在游戏中应用了dfs，使知识走出书本，以另一种更优化的方式呈现在自己面前。

笔者将本题难点归于两类。

第一类是界面的重复化与精细化劳动。设置背景布局、事件监听、界面跳转在这一个月里都是家常便饭。笔者总共写了十多个按钮的监听事件，碰到了数不胜数的很小但不得不解决的bug，例如界面跳转之后键盘监听失效、背景没有刷新、鼠标定位不符合预期等。然而笔者也在一次次的查阅与修改中学会了各式各样的解决方案，比如requestFocus()、pack()、repaint()等，都帮助笔者不算顺利但成功地解决了许多界面的小问题。如果没有这许多细节的处理，此游戏或许只能困于“每次展示功能都必须重新运行一次代码”的窘境。

第二类是核心代码的设计。总代码量约1500行，但核心代码实际上只有200行左右，而这区区13%的比例正是实现课程设计任务要求的精髓。本题的dfs和课本上的区别很大，相同的原理却需要重新设计思路。笔者一边借鉴网上的经验一边自己修改，前前后后改了四五次才设计出符合自己迷宫的dfs。还有控制老鼠不穿墙而过的代码，在IDEA里面不过两行就解决，简洁而管用。所有核心代码的大前提就是将迷宫数字化，即1代表墙，0代表没走过的路，2代表走过的路。

纵然笔者在制作过程中反复优化玩家体验感，添加了许多便捷的交互功能，但总有考虑不周之处，均已写入改进方案中。验收的时候助教也给出了一些好建议。若今后有时间，笔者希望能尽量呈现更好的游戏内容与更佳的界面交互。

短时间内匆匆将游戏收尾，笔者允许自己容忍一些不够好的地方，同时也为自己的界面制作感到自豪。背景和素材图片是前往视觉中国官网筛选而得，标题文字是自己在ipad上手写并上传的，因此不论怎样说，这都是一个80%原创的游戏，也是笔者做的第一个游戏。笔者确实在不停地进行游戏优化与自我提升。总而言之，这一个月写课设代码虽然非常辛苦，但收获颇丰，倒也值得。

感谢小老鼠和奶酪陪我熬过的不知道多少个夜晚。