

**程 序 设 计 专 题**

**大 程 序 报 告**



大程名称： 迷宫游戏

**指导老师： 张引**

**2022~2023春夏学期 2023 年 6 月 3 日**

**报告撰写注意事项**

1. 图文并茂。文字通顺，语言流畅，无错别字。
2. 书写格式规范，排版良好，内容完整。
3. 存在拼凑、剽窃等现象一律认定为抄袭；0分
4. 蓝色文字为说明，在最后提交的终稿版本，请删除这些文字。

**目 录**

[1 大程序简介 4](#_Toc65352773)

[1.1 选题背景及意义 4](#_Toc65352774)

[1.2 目标要求 4](#_Toc65352775)

[1.3 术语说明 4](#_Toc65352776)

[2 需求分析 4](#_Toc65352777)

[2.1 业务需求 4](#_Toc65352778)

[2.2 功能需求 4](#_Toc65352779)

[2.3 数据需求 5](#_Toc65352780)

[2.4 性能需求 5](#_Toc65352781)

[3 程序开发设计 5](#_Toc65352782)

[3.1 总体架构设计 5](#_Toc65352783)

[3.2 功能模块设计 5](#_Toc65352784)

[3.3 数据结构设计 5](#_Toc65352785)

[3.4 源代码文件组织设计 5](#_Toc65352786)

[3.5 函数设计描述 6](#_Toc65352787)

[4 部署运行和使用说明 6](#_Toc65352788)

[4.1 编译安装 6](#_Toc65352789)

[4.2 运行测试 6](#_Toc65352790)

[4.3 使用操作 7](#_Toc65352791)

[5 团队合作 7](#_Toc65352792)

[5.1 任务分工 7](#_Toc65352793)

[5.2 开发计划 7](#_Toc65352794)

[5.3 编码规范 7](#_Toc65352795)

[5.4 合作总结 7](#_Toc65352796)

[5.5 收获感言 8](#_Toc65352797)

[6 参考文献资料 8](#_Toc65352798)

迷宫游戏大程序设计项目

# 大程序简介

## 选题背景及意义

迷宫游戏是一种古老而受欢迎的游戏类型，玩家需要在复杂的迷宫中寻找出口。我们选择迷宫游戏有以下几点原因：

1. 挑战与解谜：迷宫游戏提供了一个充满挑战性和解谜性的环境，玩家需要运用逻辑推理、空间感知和问题解决能力来找到正确的路径。

2. 探索与发现：迷宫游戏中的迷宫通常充满了未知的区域和隐藏的秘密。迷宫游戏可以给玩家提供一个虚拟的世界，让他们可以自由地探索和发现新的地方、物品或故事情节。这种探索的感觉可以增加游戏的乐趣和长期的吸引力。

## 目标要求

1. 地图生成：

* 结构生成：迷宫地图可以生成28\*28的方形迷宫。
* 路径连通性：迷宫地图生成时确保从迷宫的起点到终点之间存在至少一条路径（可行解）。
* 随机性：迷宫地图生成时通常使用深度优先算法来创建不同的迷宫结构。这样可以确保每次生成的迷宫都是独特的，增加了游戏或谜题的复现性和挑战性。
* 难度调整：迷宫地图生成时具有调整迷宫难度的选项。通过调整生成算法的参数，可以生成简单、普通、困难的迷宫，满足不同玩家的需求。
* 可视化和输出：生成的迷宫以图形界面可视化的方式呈现给用户。
* 扩展功能：迷宫地图生成时有如陷阱、炸弹道具等玩法，还有计时器记录通关时间，增加游戏趣味性。

1. 迷宫求解：

* 搜索算法：迷宫求解使用深度优先搜索（DFS）算法来寻找从起点到终点的路径。
* 最优解：迷宫求解时寻找最短路径或最优路径。
* 障碍物处理：迷宫求解时可以检测并绕过障碍物，以找到有效的路径。
* 可视化和输出：迷宫求解时可以将解决方案以标记路径的方式可视化的方式呈现给用户，支持单步执行、自动执行。

1. 菜单和工具栏（快捷键）功能：
2. 开始界面

开始游戏：启动一个新的迷宫游戏。

设置：提供选择游戏难度级别的选项，如简单、普通、困难。

介绍：提供游戏玩法说明。

退出游戏：关闭游戏应用程序。

1. 游戏模式选择：

闯关模式、自定义模式、迷雾模式、[一往无前]模式

介绍：提供游戏玩法说明。

返回：返回开始界面。

1. 游戏界面：

* 文件菜单：

打开：打开本地迷宫文件开始游戏。

新建：以文件方式新建一个新的迷宫游戏。

保存：将当前游戏状态保存到文件中。

* 工具栏功能：

返回：返回游戏模式选择界面。

刷新：将游戏状态重置为起始状态。

寻路：可视化显示迷宫解的路径。

可视化寻路：可视化显示迷宫解的路径和求解过程。

提示：提示游戏下一步的路径。

* 介绍：提供游戏玩法说明。
* 炸弹：使用炸弹炸毁路径。

1. 高级玩法：

生命值：玩家具有生命值，若踩到陷阱生命值将降低。

陷阱：游戏中有“看不见”的陷阱阻塞在迷宫路径中。

炸弹：迷宫中可能会随机掉落炸弹，玩家可使用炸弹炸毁路径。

1. 游戏界面：

游戏背景：采用了图形库绘图，使游戏界面更加美观

## 术语说明

BFS:广度优先算法 [DFS:深度优先搜索算法](https://blog.csdn.net/qq_38670588/article/details/108267028" \t "https://www.bing.com/_blank)

# 需求分析

## 业务需求

1. 迷宫设计：本迷宫游戏精心设计迷宫的结构、模式、难度等，以提供有趣且具有挑战性的游戏体验。

2. 控制和交互：迷宫游戏提供适合玩家操作的控制方式，鼠标可用于点击按钮，键盘可用于控制移动物以及快捷键。游戏还应该具备良好的交互性，玩家可以与迷宫中的机关、道具等进行互动。

3. 难度和进度：玩家可自由调整迷宫游戏的难度，从简单到困难，让玩家感到挑战和成就感。同时，游戏应该提供适当的关卡系统，让玩家能够逐步解锁新的迷宫和挑战。

4. 图形显示：本迷宫游戏采用漫威蜘蛛侠主题的图形显示，手工打造各界面图形，吸引玩家的眼球，并与游戏的主题相匹配。

5. 玩法介绍：鉴于本游戏玩法较丰富，需要玩家自行探索，因而我们也在各界面提供了进入玩法介绍的通道。

## 功能需求

1. 显示

为实现实时刷新的图形显示，本程序将需要实时显示的内容放入特定的display函数中，并将display函数放入回调函数中，实现了高刷显示。

1. 回调

为满足图形显示的高刷新率以及良好的交互体验，本程序采取了鼠标回调、键盘回调、字符回调、计时器。

1. 按键

为实现用户的自由式游戏体验，本程序设计了众多按键便于用户的切换需求。用户每按下一个按键，按键所对应的函数便被执行，向用户呈现出期望界面或功能。

1. 文件

为便于用户打开或保存心仪的地图，本程序实现了打开对话框或保存对话框的弹出，使用户得以熟悉地操作与选择目标文件或保存路径，同时通过文件函数完成新地图信息的导入或当前地图信息的录入。

1. 介绍

为便于用户全面了解游戏玩法及其中创新点，且扩大用户受众面，本程序专门设计了玩法介绍界面（类似于说明书）。

1. 地图生成

本程序采用深度优先算法，保证了地图生成的连通性（存在至少一条通往终点的路径）、随机性（利用随机数，使得生成地图时设置元素的方向随机，避免了一路到底的问题）、稳定性（设置相应判断条件，防止挖穿地图或形成回路），此外本程序还创新地设计了随机陷阱、随机炸弹、随机终点，增加了迷宫的趣味性，同时将打印迷宫的函数加入至回调函数中，实现了地图的实时刷新。

1. 特色玩法

本程序设计了迷雾、炸弹、生命值等特色玩法。

1. 排名

迷雾模式中，为增加玩家的胜负欲，本程序加入了排名功能，用于记录用时。

1. 图形

为实现界面的美观，本程序根据现有图形库手工绘制了一系列图形。

1. 寻路

为实现自动求解迷宫出路，本程序同时采用了广度优先算法和深度优先算法，前者用于自定义模式寻找最短路，后者用于其他模式可视化寻路过程。

## 数据需求

1.地图的保存与打开

数据结构：无明显的数据结构。

数据格式：采用.txt文本格式存储地图的相关信息，不同信息之间以“空格”间隔，依次记录起点横坐标、起点纵坐标、终点横坐标、终点纵坐标、存储地图的二维数组的各个元素。

数据内容：对于读入的数据内容应为txt文件。

1. 排名的记录

数据结构：采用单向链表由完成迷宫耗时从短到长的顺序存储玩家昵称及用时。

数据格式：无明显的数据格式。

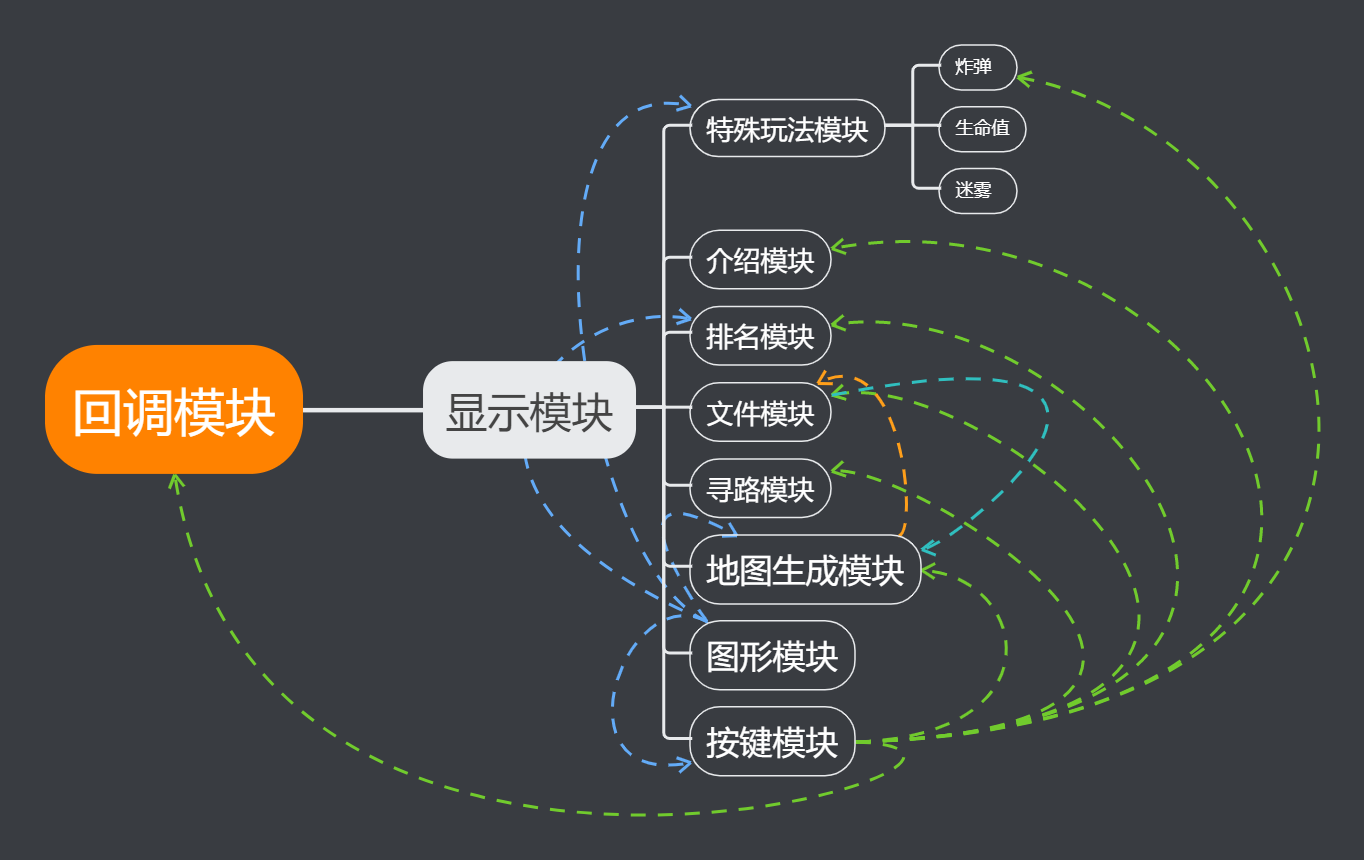
数据内容：链表的各个结点的数据内容包括数据域：玩家昵称（字符串）、耗费时长（整型数据）与指针域：指向下一个结点的结构体指针。

## 性能需求

满足各种回调函数和屏幕显示的高刷新率，无明显的运行时间速度要求。

# 程序开发设计

## 总体架构设计



## 3.2功能模块设计

**回调+显示模块：**进行界面设计，通过全局变量的控制调整信息的出现与否，根据玩家操作进行反馈，主要使用模拟算法，并通过回调模块实现实时刷新。

**图形+介绍模块：**借助简单图形库在特定位置进行绘图与打字。

**地图生成模块：**运用深度优先算法随机生成地图。

**寻路模块：**运用深度优先搜索（DFS）与广度优先搜索算法（BFS），借助递归和循环进行寻路。

**文件模块：**借助windows库自带函数，进行文件打开与保存操作。

**排名模块：**通过链表的数据结构实现完成迷宫的用时排名和记录。

**按键模块：**将每个按键链接至相应功能的函数，实现按键“响”则函数“应”。（各按键如下图所示）

1. 首页图形显示：

向玩家展示本迷宫的LOGO设计，按任意键进入迷宫游戏。

1. 主界面：

从上至下依次呈现按键“开始游戏”、“设置”、“介绍”、“退出游戏”：

2.1开始游戏：进入模式选择界面

2.2设置：进入难度选择界面

2.3介绍：进入玩法介绍界面

2.4退出游戏：关闭程序

1. 模式选择界面：

从上至下依次呈现按键“闯关模式”、“自定义模式”、“迷雾模式”、“一往无前模式”、“返回”：

3.1闯关模式：顾名思义该模式中玩家到达终点后游戏不会结束，可继续下一关。该模式中玩家具有生命值，若踩到陷阱生命值将减少，减少至0则游戏结束。值得注意的是，随机生成的陷阱起初并不可见，已踩过的陷阱可见，但陷阱的伤害仍然存在，因而玩家需要小心谨慎地斟酌每一步。地图中还会随机生成炸弹，捡到炸弹后可炸毁墙壁或陷阱为自己开路。

3.2自定义模式：该模式中地图将被清空，玩家可根据自己的想法编辑起点、终点、墙壁，创造出自己的地图。

3.3迷雾模式：该模式中玩家行走过程中地图会闪烁式呈现，除了需要凭借玩家记忆力短暂“失明地”移动外，还需小心在迷雾中反复撞到陷阱导致生命值骤降。该模式中具有独有的排名功能，用以记录在迷雾模式下完成迷宫的时间。

3.4一往无前模式：该模式中，玩家所走过的上一步会生成一堵墙，“一往无前”顾名思义就是阻止玩家走回头路反复试错，因此玩家需斟酌再三再移动。

1. 菜单：

各个模式中文件栏功能相同，工具栏功能部分不同，以下将分类介绍。

4.1文件栏菜单：从上至下依次呈现按键“打开”、“新建”、“保存”。

打开：弹出打开对话框，选择按所需数据格式保存了地图信息的txt文本文件即可在当前模式导入打开的地图，若所选文件不是txt文件或者存储的数据格式有误，则会提示“格式错误”并无法成功导入地图。

新建：进入自定义模式，并执行保存功能。

保存：弹出保存对话框，选择要保存的txt文件存放的目标地址，并给该文件命名，如果用户命名时未加上.txt后缀，我们设置了自动加上.txt后缀保证所有保存的地图信息都是txt文本文件。

4.2工具栏菜单：

4.2.1闯关模式：从上至下依次呈现按键“返回”、“刷新”、“寻路/清空寻路”、“可视化寻路/清空寻路”、“提示/关闭提示”。

返回：返回至模式选择界面；

刷新：生成新的地图并初始化所有状态；

寻路/清空寻路：可视化显示一条由玩家当前位置至终点的安全道路（寻路过程中无法进行其他操作），寻路结束后可清空寻路的路径；

可视化寻路：可视化显示迷宫解的路径和求解过程（寻路过程中无法进行其他操作），寻路结束后可清空寻路的路径；

提示/关闭提示：可视化显示通往终点的下一步的路径（开启提示时无法寻路，会提示警告“请先关闭提示功能后在进行寻路”，提示关闭后警告消失）。

4.2.2自定义模式：从上至下依次呈现按键“返回”、“刷新”、“寻找最短路径/清空寻路”、“编辑/退出编辑”。

返回：返回至模式选择界面；

刷新：清空当前地图；

寻找最短路径/清空寻路：可视化显示由玩家当前位置至终点的最短路径，寻路结束后可清空寻路的路径；

编辑/退出编辑：打开编辑功能后可设置起点、终点、墙壁，退出编辑功能后地图元素的设置按钮消失；

4.2.3迷雾模式：从上至下依次呈现按键“返回”、“刷新”、“开启/关闭迷雾”、“寻路/清空寻路”、“重新开始”。

返回：返回至模式选择界面；

刷新：生成新的地图并初始化所有状态；

开启/关闭迷雾：开启迷雾后地图开始闪烁，玩家可以开始移动，并开始计时，无法使用寻路功能；关闭迷雾后地图停止闪烁，玩家无法移动，计时暂停，但可使用寻路功能；

寻路/清空寻路：可视化显示一条由玩家当前位置至终点的安全道路（寻路过程中无法进行其他操作）（若在迷雾模式中使用此功能会提示警告“请先关闭迷雾”且无法寻路，关闭迷雾后警告消失），寻路结束后可清空寻路的路径；

重新开始：重置玩家位置至起点并初始化所有状态；

4.2.4一往无前模式：从上至下依次呈现按键“返回”、“刷新”、“寻路/清空寻路”、“可视化寻路/清空寻路”、“提示/关闭提示”、“重新开始”。

返回：返回至模式选择界面；

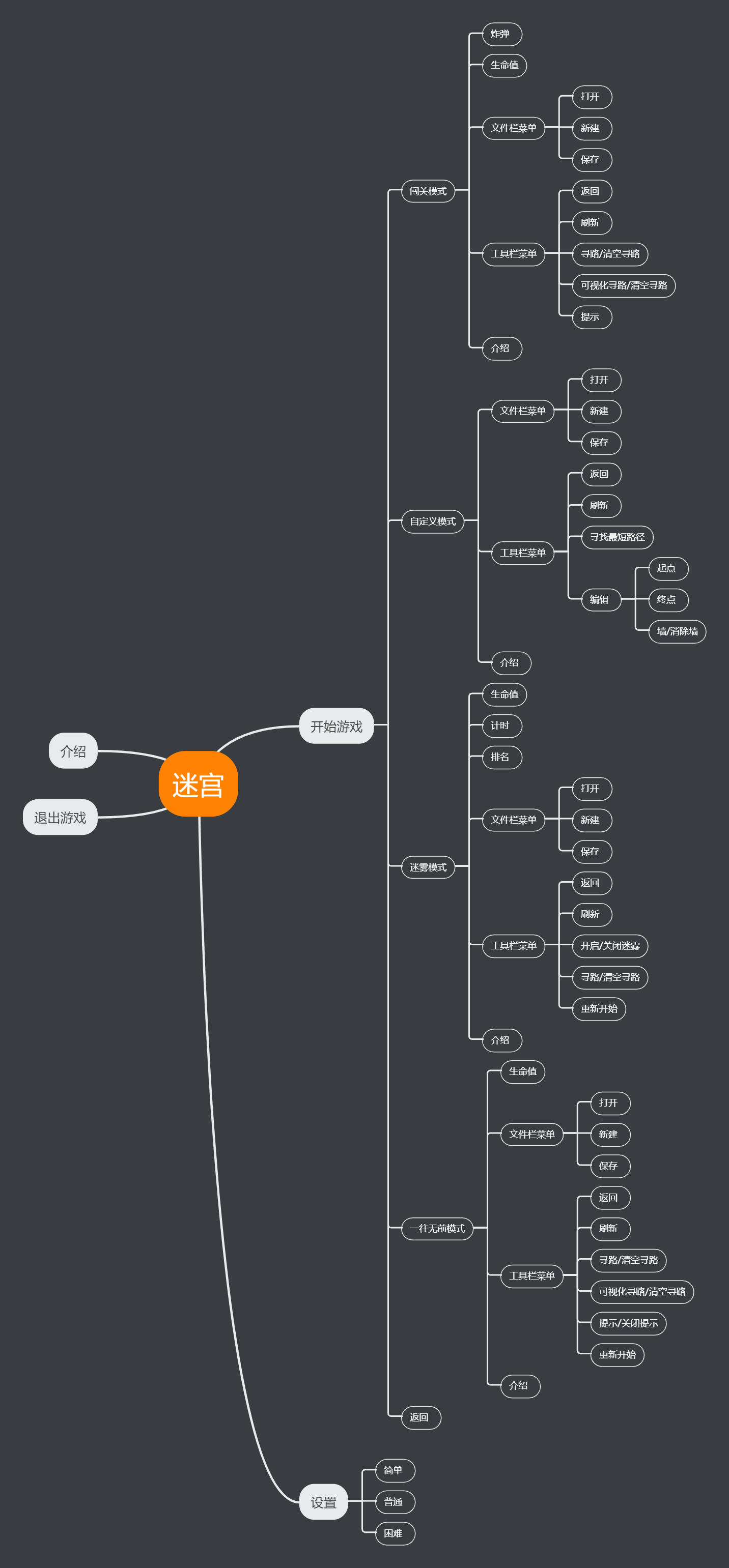
刷新：生成新的地图并初始化所有状态；

寻路/清空寻路：可视化显示一条由玩家当前位置至终点的安全道路（寻路过程中无法进行其他操作）（若无解会提示“无法找到出路”），寻路结束后可清空寻路的路径；

可视化寻路：可视化显示迷宫解的路径和求解过程（寻路过程中无法进行其他操作）（若无解会提示“无法找到出路”），寻路结束后可清空寻路的路径；

提示：可视化显示通往终点的下一步的路径（开启提示时无法寻路，会提示警告“请先关闭提示功能后在进行寻路”，提示关闭后警告消失）；

重新开始：重置玩家位置至起点并初始化所有状态。



## 3.3数据结构设计

**全局变量的定义**

int Map[L][L];//记录迷宫地图

int Bx=2,By=2;//移动物实时位置坐标

int visit[50][50]={0};//记录是否访问

int road[2000][2];//记录走过的路

int step=0;//行走顺序

int Ex,Ey;//出口坐标

int Difficulty=2;//迷宫复杂度

int All\_the\_way[L\*L][2];//记录寻路过程中所有点的坐标

int All\_step;//记录坐标编号

int mseconds500 = 5; //路径显示计时器时间间隔

int mseconds1000 = 1;//寻路过程显示时间间隔

int mseconds1500=1000;

int Time\_i=0;

int Time\_j=0;

int Timer1=0;//计时器1的状态

int Timer2=0;//计时器2的状态

int Timer3=0;//计时器3的状态

int Timer33=0;//计时器3的状态

int Time=0;//记录时长

char time1[10];//保存时长数据

int Level=1;//记录关卡

char level[10];//保存关卡数据

double life=5;//生命值

int boom=0;//炸弹数量

int All\_way\_visit[L][L];//每个点的状态 判断是否输出过

double Cx,Cy;int Cevent;//记录鼠标的实时坐标及按下情况

char Name[100];

int path1[L\*L],path2[L\*L];

int trace[2][2];//记录移动物的上一步和上上步

int trap[2];//记录移动物踩到陷阱的一步

int no\_way=0;//状态量。判断是否有路

int Get\_the\_end=0;//状态量，判断是否到达终点

int Button\_find\_way\_down=0;//状态量，判断"Find the way"是否被执行

int Button\_find\_all\_the\_way=0;//状态量，判断"Find all the way"是否被执行

int Button\_Stop=0;//状态量，判断"Button\_stop"是否被执行

int Button\_Start=0;//状态量，判断"Button\_start"是否被执行

int Button\_find\_the\_shortest\_way=0;//状态量判断“find the shortest way”是否按下

int Min=L\*L;

int Button\_DIYKey=0; //状态量，判断按键"Start"/"End"/"Wall"是否按下

int Is=1;//状态量，判断是否打印移动物体

int Enter=5;//状态量，判断进入了哪个界面

int Auto=1;//状态量，判断是否进入自动建图模式

int Go=0;//状态量，判断是否进入一往无前模式

int Item=0;//状态量，判断是否打开道具栏

int DIY=0;//状态量，判断是否打开了编辑菜单

int Tip=0;//状态量，判断是否给出警告

int Winer=0;//状态量，判断是否获胜

int winer=0;//状态量，判断是否输入完成

int link=0;//状态量，确认是否建立链表

int Get\_in=0;//状态量，确认是否输入链表

int rank1=0;//状态量，判断排名是否按下

int Hint=0;//状态量，判断提示是否打开

int lose=0;//状态量，记录游戏是否结束

int heat=0;//状态量，记录游戏是否胜利

char CurrentFileName[MAX\_PATH];//存入文件相关操作所需文件名

int min=0;//最小路

int help=1;//状态量，判断介绍是否打开

// 颜色库

char Color[][100]={"Blue","Black","Red","Green","Gray","Yellow","White","Dark Gray","Light Gray","Brown","Orange","Violet","Magenta","Cyan"};

struct Ranking{

char Name[100];

int score;

struct Ranking \*Next;

};排名的结构体

struct Ranking \*Head,\*tail;

**宏定义**

#define L 30

#define BLK 0

#define ACS 1

#define TRP 2

#define EXT 3

#define BOM 4

#define TRP2 5

#define TIMER\_BLINK500 1

#define TIMER\_BLINK1000 2

#define TIMER\_BLINK1500 3

#define PI 3.1415926

## 3.4函数设计描述

**文件：main.c**

1. void Main(void)

功能描述：确定窗口名称、确定窗口大小、初始化窗口、初始化键盘、鼠标、字符、计时器回调函数

参数描述：无

返回值描述：无

局部变量：无

特殊算法：无

模块对应关系：是所有模块的开始，其打开了窗口来显示其他模式

1. void display(void)

功能描述：主要负责各个模块间的显示功能，实现背景、按钮、地图等主要功能的显示

参数

重要变量（全局）：int Enter（主要作用是确定在什么界面，当数值为0时为主菜单界面，当数值为1时为副菜单的界面，当数值为2时进入自动或者手动建图或者一往无前模式，当数值为3时进入迷雾模式，当数值为4时进入难度调整界面，数值为5为游戏封面）

函数在模块中对应关系：该函数是整个项目最为重要的函数，该函数起到连接所有模块的作用，通过该函数完成模块之间的切换显示，以及所有功能的可视化显示。

参数描述：无

返回值描述：无

局部变量：无

特殊算法：无

1. void Display(void)

功能描述：此函数功能是类似于display（）函数的，但是这个函数主要负责迷雾模式的特殊显示，主要负责与定时器连接，定间隔显示迷雾，同时确保正确显示所记录的时间。最主要的是防止键盘以及鼠标回调对于显示的影响。

重要变量（全局） int Timer33 （该变量是用来确定计时器3的开始，来确定已经进如迷雾模式）

函数在模块中的对应关系：主要的负责迷雾模式的显示功能，包块其中显示计时器的时间，显示迷雾来遮挡地图，以及迷雾模式的时间排名的显示。

参数描述：无

返回值描述：无

算法描述：无

局部变量：无

**文件：button.c(原型定义) button.h（声明）**

1. void Button\_Main\_Menu(void)

功能描述：建立多个按钮，主要用于主界面，按钮分别为开始游戏（进入第二界面），设置（调整难度），介绍（进入介绍界面），以及退出游戏

参数描述：无

返回值描述：无

局部变量：无

函数算法描述：无

模块对应关系：主要负责按钮模块，通过按钮模块来实现与其他功能的链接，同时来控制游戏的正常开始结束。

1. void Button\_Second\_Menu(void)

功能描述：功能与上述函数功能相似，主要在第二界面建立多个按钮，通过按钮来确定需要的游戏模式分别为：闯关模式，自定义模式，迷雾模式，一往无前模式，以及介绍和返回。

参数描述：无

返回值描述：无

算法描述：无

模块对应关系：主要是按钮模块中，起到一个界面间的连接作用，通过按钮来连接各个不同的功能，起到一个中枢的作用。

1. void Button\_MenuList(void)

功能描述：该函数也是一个功能性的函数，主要功能是在第三界面中的闯关模式建立一个菜单，以及少量的按钮。主要来实现各功能如文件的保存读取，寻找路径，可视化寻路等主要功能。

参数描述：无

返回值描述：无

局部变量：char \*menuListTool[] 、char \*menuListFile[] 、int decision

局部变量功能描述：前两各字符形指针数值最主要是用于建立一个可视化菜单，并储存每个菜单上所显示的汉字。后一个int 性变量主要是用于记录玩家点击或者使用快捷键使用菜单什么功能，通过不同的数值来实现不同的功能。

模块对应关系：其作用也是一个中枢作用，但是不是界面间的联系，二十具体功能与界面的联系，是在一个迷宫图像上实现玩家控制使用功能的一个连接。

1. void Button\_Boom(double x,double y,int Event)

功能描述：该函数主要分为两个功能，第一个是建立一个按钮并告知玩家所拥有的炸弹的数量，第二个功能是使用炸弹，更具鼠标回调传如的数据确定在途中的方块，通过精准的定位来确定删除哪一个墙。

参数描述：double x,double y,int Event，三个参数均为鼠标回调传入，x、y主要是鼠标的实时坐标的转化，event主要是判断鼠标的行为。

返回值描述：无

重要局部变量：无

重要全局变量：int Item状态量，判断是否打开道具栏

函数算法描述：无

模块对应关系：主要也是按钮模块的函数，实现界面与功能的连接，通过该函数，使得玩家能使用道具，增加游戏趣味性。

1. void Button\_MenuList4(void)

功能描述：该函数也是一个功能性的函数，主要功能是在第三界面中的一往无前模式建立一个菜单，以及少量的按钮。主要来实现各功能如文件的保存读取，寻找路径，可视化寻路等主要功能。

参数描述：无

返回值描述：无

局部变脸：char \*menuListTool[] 、char \*menuListFile[] 、int decision

局部变量功能描述：前两各字符形指针数值最主要是用于建立一个可视化菜单，并储存每个菜单上所显示的汉字。后一个int 性变量主要是用于记录玩家点击或者使用快捷键使用菜单什么功能，通过不同的数值来实现不同的功能。

模块对应关系：其作用也是一个中枢作用，但是不是界面间的联系，而是具体功能与界面的联系，是在一个迷宫图像上实现玩家控制使用功能的一个连接。

1. void Button\_Find\_the\_shortest\_road(void)

功能描述：主要作用是判断该菜单按键是否按下，在不同的按键状态下实现不同的功能。

参数描述：无

返回值描述：无

局部变量：无

全局变量：int Button\_find\_the\_shortest\_way:状态量判断“find the shortest way”是否按下, int no\_way=0;//状态量。判断是否有路.

函数算法描述：无

模块对应关系：主要起到一个模块间的连接作用，将按键模块与寻路模块相连接，并完善过程中出现的问题。

1. void Button\_DIYCreate(void)

功能介绍：主要功能初始化自动建图模式，建立周围的墙体，建立内部空白的图像。

参数描述：无

返回值描述：无

局部变量：无

函数算法描述：无

模块对应关系：主要起到一个模块间的连接作用，将按键模块与自动建图模块相连接，并完善过程中出现的问题。

1. void Button\_DIY(double x,double y,int Event)

功能描述：主要作用为通过建立多个按钮来形成手动建图模式的操作流程，同时根据鼠标的回调函数判断需要设置墙体，起点终点的位置。

参数描述：double x,double y,int Event，三个参数均为鼠标回调传入，x、y主要是鼠标的实时坐标的转化，event主要是判断鼠标的行为。

局部变量：static int inEdit1=0,inEdit2=0;维持量，保持编辑状态可以一直执行

static int start=0,end=0;记录起点和终点数量，防止产生多个

返回值描述：无

函数算法描述：无

模块对应关系：主要起到板块间的连接作用，将按键模块与自动建图模块相连接。

1. void Button\_Refresh(void)

功能描述：主要作用为刷新地图，如果在自动模式，会刷新一个新的地图，如果是手动建图模式，就清空之前建立的地图，迷雾模式会刷新地图重新计时。

参数描述：无

返回值描述：无

局部变量：无

函数算法描述：无

模块对应关系：这个是一个全模块的函数，对于每一个模块的实现都起到了作用，其能刷新图像的显示，建立新的地图，重新开始游戏，刷新每一个使用过的变量。

1. void Button\_MenuList2(void)

功能描述：该函数也是一个功能性的函数，主要功能是在第三界面中的DIY模式建立一个菜单，以及少量的按钮。主要来实现各功能如文件的保存读取，寻找最短路径，以及手动建立图像等主要功能。

参数描述：无

返回值描述：无

局部变量：char \*menuListTool[] 、char \*menuListFile[] 、int selection

局部变量功能描述：前两个字符形指针数值最主要是用于建立一个可视化菜单，并储存每个菜单上所显示的汉字。后一个int 性变量主要是用于记录玩家点击或者使用快捷键使用菜单什么功能，通过不同的数值来实现不同的功能。

算法描述：无

模块对应关系：其作用也是一个中枢作用，但是不是界面间的联系，而是具体功能与界面的联系，是在一个迷宫图像上实现玩家控制使用功能的一个连接。

1. void Button\_MenuList3(void)

功能描述：该函数也是一个功能性的函数，主要功能是在第三界面中的迷雾模式建立一个菜单，以及少量的按钮。主要来实现各功能如文件的保存读取，开启迷雾，寻路等主要功能。

参数描述：无

返回值描述：无

局部变量：char \*menuListTool[] 、char \*menuListFile[] 、int decision

局部变量功能描述：前两个字符形指针数值最主要是用于建立一个可视化菜单，并储存每个菜单上所显示的汉字。后一个int 性变量主要是用于记录玩家点击或者使用快捷键使用菜单什么功能，通过不同的数值来实现不同的功能。

算法描述：无

模块对应关系：其作用也是一个中枢作用，但是不是界面间的联系，而是具体功能与界面的联系，是在一个迷宫图像上实现玩家控制使用功能的一个连接。

1. void Button\_stop(void)

功能描述：一个停止按键，主要是在迷雾模式中停止迷雾的重复以及计时。来暂停计时器。刷新一些重要的关于状态的全局变量确保下一次游戏能够正常进行。

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

模块对应关系：主要是用于按钮模式与回调模式的连接，通过使用按钮来停止计时器的运行。

1. void Button\_start(void)

功能描述：开始计时器，在迷雾迷雾模式中使用，可以理解成开始游戏的按钮。按下后将开始计时以及迷雾的刷新。

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

模块对应关系：主要是用于按钮模式与回调模式的连接，通过使用按钮来开始计时器的运行。

1. Button\_Difficulty（void）

功能描述：建立多个按钮来确认不同的难度。

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

重要全局变量：int Difficulty;迷宫复杂度

模块对应关系：主要还是作用在按钮模块，主要负责一个界面的功能的实现，在设置功能中，实现选择难度的功能。

**文件：model.c(原型定义) model.h（声明）**

1. void Fog\_model(void)

功能描述：主要是根据计时器来画出迷雾，更具计时器时间来判断是否显示。、

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

模块对应关系：主要也是一个跨模块连接作用的函数，连接了计时器与显示模块。

1. void Life(void)

功能描述：判断是都需要流失体力值，以及记录玩家的生命值，同时显示在游戏中。同时判断是否因为没有血量而死亡。

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

重要全局变量：double life;生命值

模块对应关系：主要也是一个跨模块连接作用的函数，连接了游戏功能与显示，满足了游戏的趣味性也保证了显示血量。

1. void Boom(void)

功能描述：在游戏中捡起炸弹后，把炸弹数量+1，同时把防炸弹的路变成一个普通的通路。

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

重要全局变量：int boom;/炸弹数量

模块对应关系：主要是在游戏的游玩板块中实现作用，起到记录以及刷新的作用。

**文件：callback.c(原型定义) callback.h（声明）**

1. void KeyboardEventProcess(int key, int event)

功能描述：键盘回调函数

参数描述：int key（按键）, int event（判断状态）

返回值描述：无

局部变量：无

算法描述：无

模块对应关系：主要是在回调模块。

1. void void CharEventProcess(char ch)

功能描述：字符回调函数

参数描述： char ch（判断按下的字符）

返回值描述：无

局部变量：无

算法描述：无

模块对应关系：主要是在回调模块。

1. void TimerEventProcess(int timerID)

功能描述：计时器回调函数

参数描述： int timerID（判断计时器编号）

返回值描述：无

局部变量：无

算法描述：无

模块对应关系：主要是在回调模块。

1. void MouseEventProcess(int x, int y, int button, int event)

功能描述：鼠标回调函数

参数描述：int x, int y（判断鼠标的位置）, int button（判断使用的是哪一个按键）, int event（判断状态）

返回值描述：无

局部变量：无

算法描述：无

模块对应关系：主要是在回调模块。

**文件：file.c(原型定义) file.h（声明）**

1. void Open(void)

功能描述：主要作用是打开一个文件，同时把文件中的主要的信息内容转化为程序中所需要的变量数值。

参数描述：无

算法描述：无

返回值描述：无

重要局部变量：typedef struct{

int bx;int by;

int ex;int ey;

int map[L][L];

}Data;

Data \*mp=(Data \*)malloc(sizeof(Data));

定义了一个结构体与一个结构体指针，来暂时性的储存读取的数据。

模块对应：主要对应文件模块，实现对于文本文件的读取并转化为程序所需要的内容。

1. void Open\_File(char \*path)

功能描述：主要负责使玩家选择需要打开的文件并记录下文件的名称。

参数描述：char \*path（记录提供的文件地址）

算法描述：无

重要局部变量：无

返回值描述：无

模块对应：主要对应文件模块，实现对于文本文件的选择功能。

1. void Save(void)

功能描述：主要作用是新建一个文件，同时把程序中所需要的变量数值内容转化为文件中的主要的信息，并储存。

算法描述：无

参数描述：无

返回值描述：无

重要局部变量：无

模块对应：对应文件模块，使得将程序内的数据储存在文本文件中。、

1. void Save\_File(char \*file)

功能描述：能实现玩家自选路径的保存。

算法描述：无

返回值描述：无

参数描述：char \* file（记录提供的文件地址）

重要局部变量：无

模块对应：主要对应文件模块，能实现玩家自选路径的保存。

**文件：help.c(原型定义) help.h（声明）**

1. void Help(void)

功能描述：再点开介绍功能后该函数打开性的界面，来显示对游戏的简单介绍。

参数描述：无

返回值描述：无

局部变量定义：无

算法描述：无

模块对应关系：主要对应显示以及游戏介绍模块。

**文件：map.c(原型定义) map.h（声明）**

1. void Maze(void)

功能描述：主要作用是建立一个随机生成的地图并储存每一个点的信息，同时随机生成陷阱等增加趣味性的元素。

参数描述：无

返回值描述：无

局部变量定义：int \*\*map=(int\*\*)malloc(L\*sizeof(int\*));义二级指针，便于各个函数对地图进行编辑

重要全局变量：int Map[L][L];记录迷宫地图

算法描述：无

模块对应关系：主要是负责地图生成的模块来建立一个随机生成地图。

1. void Print\_Maze(void)

功能描述：根据所记录的迷宫来实时显示出迷宫的全貌。

参数描述：无

返回值描述：无

局部变量定义：无

算法描述：无

重要全局变量：int Map[L][L];记录迷宫地图

模块对应关系：连接功能主要是将地图模块与显示模块相连接，根据所记录的迷宫来实时显示出迷宫的全貌。

1. void Digmap(int \*\*map,int x,int y)

功能描述：根据深度优先算法，来建立一个随机生成的地图，其中保证了不会形成环路，确保所形成的地图一定有解，同时在地图的右上随机生成一个终点。

返回值描述： int \*\*map（地图的二维数组，将建立好的地图记录在这个数组中并返回）

参数描述：int \*\*map（地图的二维数组，将建立好的地图记录在这个数组中并返回）,int x,int y（是地图开始挖的起点）

重要局部变量：使上下左右四个方向随机化（与其在direction数组中的位置错位），使得挖路方向随机化 int direction[4][2]={{1,0},{-1,0},{0,1},{0,-1}};

算法描述：深度优先算法

模块对应关系：主要是地图建立模块，随机生成一个复杂的地图。

1. void Createmap(int \*\*map)

功能描述：在上述函数所建立的地图中，随机生成陷阱，终点炸弹等增加趣味性的道具等。

返回值描述：int \*\*map（地图的二维数组，将建立好的地图记录在这个数组中并返回）

参数描述：int \*\*map（地图的二维数组，将建立好的地图记录在这个数组中并返回）

重要局部变量：无

算法描述：深度优先算法

模块对应关系：主要是地图建立模块，随机生成陷阱，终点炸弹等增加趣味性的道具等。

**文件：move.c(原型定义) move.h（声明）**

1. void MoveBlock(int key,int event)

功能描述：主要功能为确保移动物在地图中的正常移动，无法穿墙，判断陷阱，炸弹，终点等。

参数描述：int key（判断方向键）,int event（判断是否按下方向建）

返回值描述：无

重要局部变量：无

函数算法描述：无

模块对应：主要是对应物块移动，确保游戏的正常进行。保证基本功能的实现。

文件：rank.c(原型定义) rank.h（声明）

1. void Build\_a\_leaderboard(void)

功能描述：初始化链表，先初始化前五个数据都为empty和10000秒完成游戏。

参数描述：无

返回值描述：无

重要局部变量：struct Ranking \*p=(struct Ranking \*)malloc(sizeof(struct Ranking));（主要是一个指针，指向正在编辑的链表）

函数算法描述：无

模块关系：主要是排名模块，初始化链表，先初始化前五个数据都为empty和10000秒完成游戏。

1. void Add\_rank(char \*name,int grade)

功能描述：向链表中添加新的数据，通过于遍历链表，来对于时间比大小，找到需要添加的位置，向其中添加新的节点。

参数描述：char \*name（向链表中传入的名称）,int grade（向链表中传入的成绩）

返回值描述：无

重要局部变量：struct Ranking \*p1（主要是一个指针，指向正在编辑的节点）,\*p2（主要是一个指针，指向正在编辑的上一个节点）,\*p（新建的节点地址）;

算法描述：无

模块关系：主要是排名模块，向链表中添加新的数据，通过于遍历链表，来对于时间比大小，找到需要添加的位置，向其中添加新的节点。

1. void Button\_rank(void)

功能描述：主要是建立一个按钮，来显示排名。

参数描述：无

返回值描述：无

算法描述：无

重要局部变量：无

模块关系：主要是排名模块，也有连接功能来连接排名模块与显示模块。

**文件：rank.c(原型定义) rank.h（声明）**

1. void sp(double px,double py,double l);

功能描述：画人物。

参数描述：double px,double py（位置）,double l（大小）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawWall(int px,int py,double size);

功能描述：画墙。

参数描述：int px,int py（位置）, double size（大小）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void Homepage();

功能描述：画界面。

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void Dun();

功能描述：画首界面。

参数描述：无

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawNet(int x,int y,double size);

功能描述：画蜘蛛网。

参数描述：int x,int y（位置）, double size（大小）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void forward(double distance);

功能描述：画蜘蛛网。

参数描述：double distance（距离）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void block(int x,int y,int color,double size);

功能描述：画出边长为小方块 根据数组坐标转化 不带边框。

参数描述：int x,int y（位置）, double size（大小），int color（颜色）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void Block(int x,int y,int color,double size);

功能描述：画出边长为小方块 根据数组坐标转化 带边框。

参数描述：int x,int y（位置）, double size（大小），int color（颜色）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawBoom(int x,int y,double size);

功能描述：画炸弹

参数描述：int x,int y（位置）, double size（大小）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawEnd (int x,int y,double size);

功能描述：画终点

参数描述：int x,int y（位置）, double size（大小）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawWin (int x,int y,double size);

功能描述：画win

参数描述：int x,int y（位置）, double size（大小）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawFail (int x,int y,double size);

功能描述：画fail

参数描述：int x,int y（位置）, double size（大小）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawBackground();

功能描述：画背景

参数面试：无

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

1. void DrawRankFrame(double x,double y);

功能描述：画排名

参数描述：double x,double y（位置）

返回值描述：无

局部变量：无

算法描述：无

模块连接：绘图模块。

**文件：way.c(原型定义) way.h（声明）**

1. void Find\_the\_road(int x,int y)

功能描述：通过dfs算法来寻找一个通路，并将其显示出来。

返回值描述：无

参数描述：int x,int y（开始寻路的位置）

重要局部变量：int Nx,Ny（记录走到的位置）;int direction[4][2]={{1,0},{-1,0},{0,-1},{0,1}};（方向）

算法描述：dfs算法来寻找路径

重要全局变量：int road[2000][2];记录走过的路径，int All\_the\_way[L\*L][2];//记录寻路过程中所有点的坐标

模块连接：主要是寻路模块，通过dfs算法来寻找一个通路，并将其显示出来。

1. void Print\_way1(void)

功能描述：时刻打印出寻找的路径。

返回值描述：无

参数描述：无

重要局部变量：无

算法描述：无

模块连接：主要是寻路模块，时刻打印出寻找的路径。

1. void Print\_way2(void)

功能描述：时刻打印出寻找的可视化寻路路径。

返回值描述：无

参数描述：无

重要局部变量：无

算法描述：无

模块连接：主要是寻路模块，时刻打印出寻找的路径。

1. void Find\_the\_shortest\_road(int bx,int by)

功能描述：通过bfs算法来寻找一个通路，并将其显示出来。

返回值描述：无

参数描述：int bx,int by开始寻路的位置

重要局部变量：int Nx,Ny（记录走到的位置）;int direction[4][2]={{1,0},{-1,0},{0,-1},{0,1}};（方向）

算法描述：dfs算法来寻找路径

重要全局变量：struct {

int x;

int y;

int pre;

}path[10000];结构是将每一步记录在结构体中，同时记录上一步的步数

int i=0,j=1,Px,Py,k;

int Visit[L][L]={0};判断是否走过这个点

int dir[4][2]={{1,0},{-1,0},{0,-1},{0,1}};方向

模块连接：主要是寻路模块，通过bfs算法来寻找一个通路，并将其显示出来同时还能判断是否有路。

1. Button\_Find\_the\_road(void)

功能介绍：按钮的功能当按下后执行逐步显示路径或者清除已经显示的路径。

参数描述：无

返回值描述：无

重要局部变量描述：无

算法描述：无

模块连接：主要起到显示寻路以及按钮和计时器4个模块的连接，保证游戏正常进行。

1. Button\_Find\_all\_the\_road (void)

功能介绍：按钮的功能当按下后执行逐步显示可视化寻路路径或者清除已经显示的路径。

参数描述：无

返回值描述：无

重要局部变量描述：无

算法描述：无

模块连接：主要起到显示寻路以及按钮和计时器4个模块的连接，保证游戏正常进行。

1. void Button\_Hint(void)

功能介绍：按钮功能，再按下后能够给玩家一个提示。

参数描述：无

返回值描述：无

重要局部变量描述：无

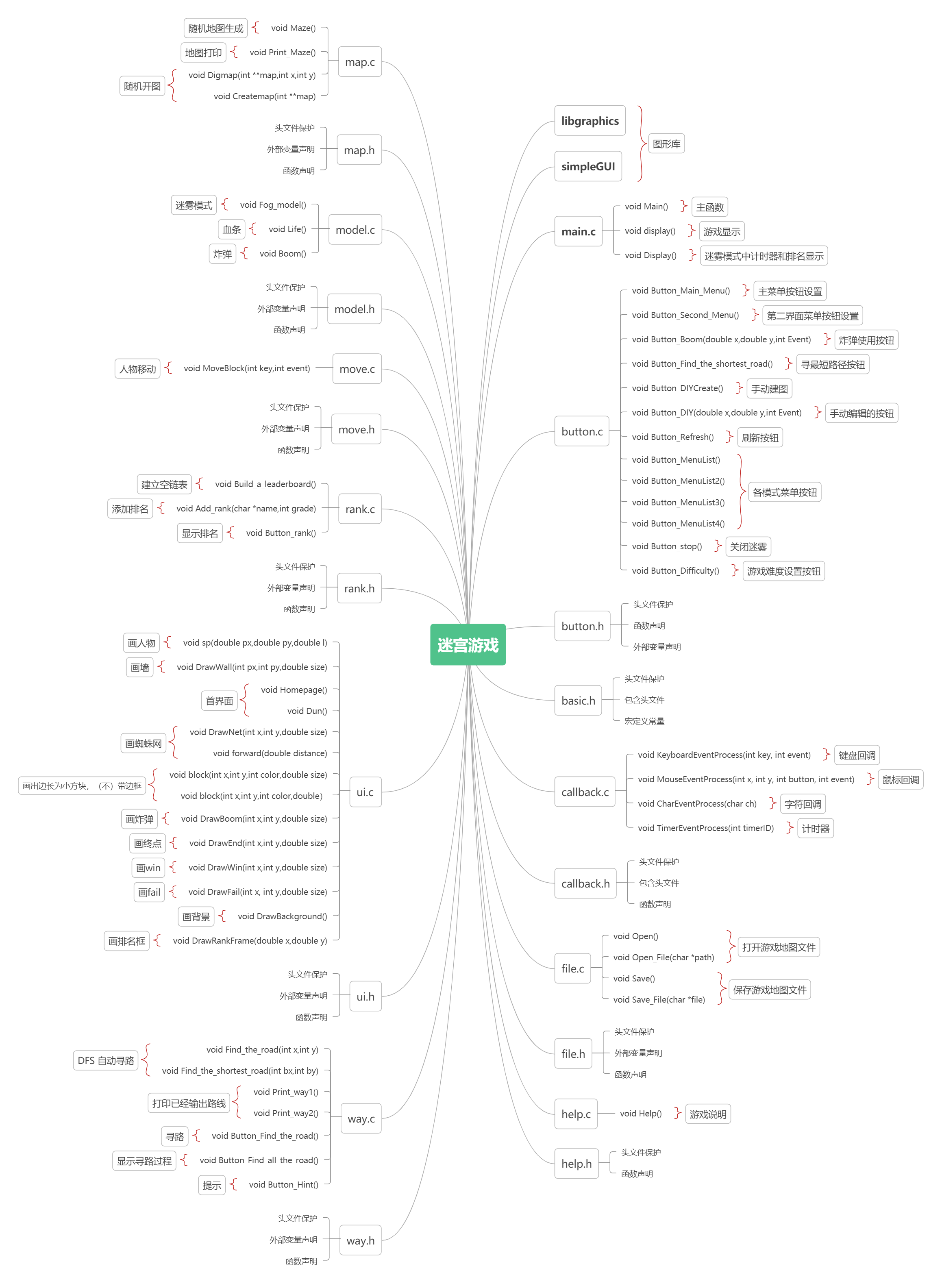
算法描述：无

模块连接：寻路显示等模块的连接作用。

## 3.5源代码文件组织设计

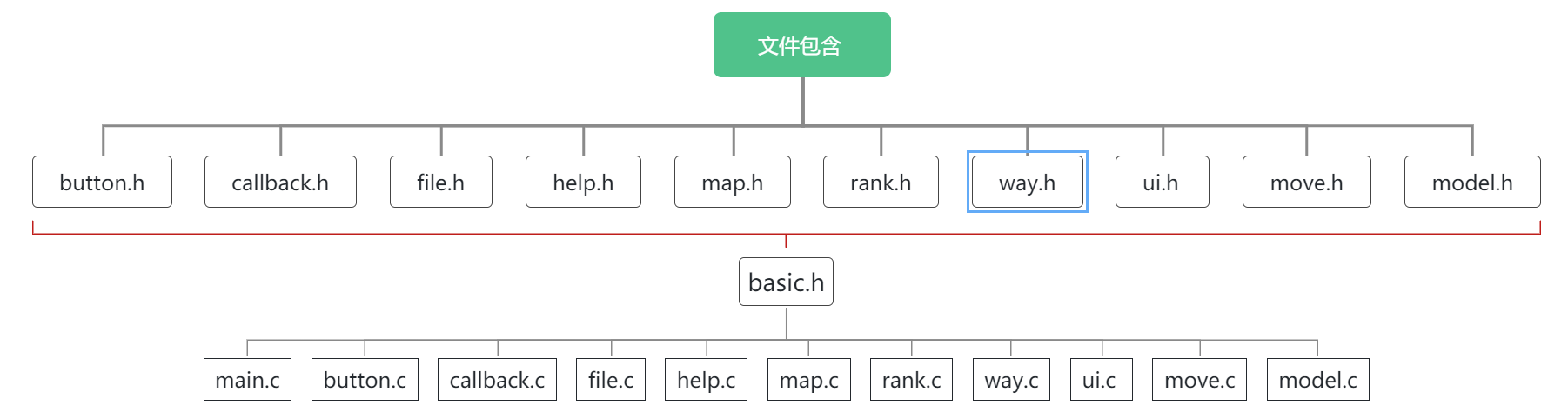
<文件目录结构>

1）文件函数结构



2）多文件构成机制

文件包含：



头文件保护：所有.h 文件都使用了 define 保护。

以basic.h(为例)

#ifndef BASIC\_H

#define BASIC\_H

……

#endif

1. main.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include "basic.h"

#include "ui.h"

#include "help.h"

#include "file.h"

#include "map.h"

#include "rank.h"

#include "way.h"

#include "move.h"

#include "model.h"

使用了这些外部函数：

int textbox(int id, double x, double y, double w, double h, char textbuf[], int buflen);

void Add\_rank(char \*name,int grade);

void Block(int x,int y,int color,double size);

void Boom();

void Button\_Difficulty();

void Button\_DIY(double x,double y,int Event);

void Button\_Hint();

void Button\_Main\_Menu()//主菜单按钮设置

void Button\_MenuList();

void Button\_MenuList2();

void Button\_MenuList3();

void Button\_MenuList4();

void Button\_rank();

void Button\_Refresh();

void Button\_Second\_Menu()//第二界面菜单按钮设置

void CharEventProcess(char ch);

void DisplayClear(void);

void DrawBackground();

void DrawRankFrame(double x,double y);

void DrawTextString(string text);

void DrawWin(int x,int y,double size);

void Fog\_model();//迷雾模式

void Help();

void Homepage();

void InitGraphics(void);

void KeyboardEventProcess(int key, int event);

void Life();

void MouseEventProcess(int x, int y, int button, int event);

void MovePen(double x, double y);

void Print\_Maze();

void Print\_way1();

void Print\_way2();

void registerCharEvent(CharEventCallback callback);

void registerKeyboardEvent(KeyboardEventCallback callback);

void registerMouseEvent(MouseEventCallback callback);

void registerTimerEvent(TimerEventCallback callback);

void SetPenColor(string color);

void SetPointSize(int size);

void setTextBoxColors(char \*frame, char\*label, char \*hotFrame, char \*hotLabel, int fillflag);

void SetWindowSize(double width, double height);

void SetWindowTitle(string title);

void sp(double px,double py,double l);

void TimerEventProcess(int timerID);

1. button.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include "button.h"

#include "basic.h"

使用了这些外部变量：

extern char CurrentFileName[MAX\_PATH];//存入文件相关操作所需文件名

extern char Name[100];

extern double Cx,Cy;

extern double life;//生命值

extern int Auto;//状态量，判断是否进入自动建图模式

extern int boom;//炸弹数量

extern int Button\_DIYKey; //状态量，判断按键"Start"/"End"/"Wall"是否按下

extern int Button\_find\_the\_shortest\_way;//状态量判断“find the shortest way”是否按下

extern int Button\_Start;//状态量，判断"Button\_start"是否被执行

extern int Button\_Stop;//状态量，判断"Button\_stop"是否被执行

extern int Cevent;//记录鼠标的实时坐标及按下情况

extern int Difficulty;//迷宫复杂度

extern int DIY;//状态量，判断是否打开了编辑菜单

extern int Enter;//状态量，判断进入了哪个界面

extern int Ex,Ey;//出口坐标

extern int Get\_in;//状态量，确认是否输入链表

extern int Go;//状态量，判断是否进入一往无前模式

extern int heat;//状态量，记录游戏是否胜利

extern int help;//状态量，判断介绍是否打开

extern int Hint;//状态量，判断提示是否打开

extern int Is;//状态量，判断是否打印移动物体

extern int Item;//状态量，判断是否打开道具栏

extern int Level;//记录关卡

extern int lose;//状态量，记录游戏是否结束

extern int mseconds1500;

extern int rank1;//状态量，判断排名是否按下

extern int Time;//记录时长

extern int Timer3;//计时器3的状态

extern int Timer33;//计时器3的状态

extern int Tip;//状态量，判断是否给出警告

extern int trace[2][2];//记录移动物的上一步和上上步

extern int Winer;//状态量，判断是否获胜

extern int winer;//状态量，判断是否输入完成

int Map[L][L];//记录迷宫地图

使用了这些外部函数：

int button(int id, double x, double y, double w, double h, char \*label);//显示一个按钮

void block(int x,int y,int color,double size);

void Button\_Find\_all\_the\_road();

void Button\_Find\_the\_road();

void CharEventProcess(char ch);//字符回调

void display(void);//显示

void DisplayClear(void);//清屏

void DrawTextString(string text);//画字符串

void KeyboardEventProcess(int key, int event);//键盘回调

void Maze()

void MouseEventProcess(int x, int y, int button, int event);//鼠标回调

void MovePen(double x, double y);

void setButtonColors(char \*frame, char\*label, char \*hotFrame, char \*hotLabel, int fillflag);//设置按钮颜色

void SetEraseMode(bool mode);

void setMenuColors(char \*frame, char\*label, char \*hotFrame, char \*hotLabel, int fillflag);//设置菜单颜色

void sp(double px,double py,double l);

void TimerEventProcess(int timerID);//计时器

1. callback.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include "basic.h"

#include "callback.h"

使用了这些外部变量：

int Timer1=0;//计时器1的状态

int Timer2=0;//计时器2的状态

int Timer3=0;//计时器3的状态

int Timer33=0;//计时器3的状态

int All\_the\_way[L\*L][2];//记录寻路过程中所有点的坐标

int All\_way\_visit[L][L];//每个点的状态 判断是否输出过

int Time\_i=0;

int Time\_j=0;

使用了这些外部函数：

double ScaleXInches(int x);//x coordinate from pixels to inches

double ScaleYInches(int y);//y coordinate from pixels to inches

void Block(int x,int y,int color,double size);

void block(int x,int y,int color,double size);

void display(void);//显示

void DisplayClear(void);//清屏

void MoveBlock(int key,int event);//控制主要移动物

void SetEraseMode(bool mode);

1. file.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include "basic.h"

#include"file.h"

使用了这些外部变量：

extern char CurrentFileName[MAX\_PATH];//存入文件相关操作所需文件名

extern int Bx,By;//移动物实时位置坐标

extern int Ex,Ey;//出口坐标

extern int Map[L][L];//记录迷宫地图

使用了这些外部函数：

void DrawTextString(string text);

void MovePen(double x, double y);

1. help.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include"basic.h"

#include"help.h"

使用了这些外部函数：

void DefineColor(string name,double red, double green, double blue);

void DrawLine(double dx, double dy);

void DrawTextString(string text);

void EndFilledRegion(void);

void MovePen(double x, double y);

void SetPenColor(string color);

void StartFilledRegion(double grayScale);

1. map.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include"basic.h"

#include"map.h"

#include"ui.h"

使用了这些外部变量：

extern double life;//生命值

extern int Auto;//状态量，判断是否自动开图

extern int Bx,By;//移动物实时位置坐标

extern int Difficulty;//难度系数

extern int Enter;//状态量，判断目前的界面

extern int Ex,Ey;//出口坐标

extern int Go;//状态量，判断一往无前模式是否打开

extern int Map[L][L];//记录迷宫地图

extern int trace[2][2];//倒数两步

extern int trap[2];//踩到陷阱的前一步

使用了这些外部函数：

int RandomInteger(int low, int high);

void DrawLine(double dx, double dy);

void DrawWall(int px,int py,double size);

void EndFilledRegion(void);

void Maze();

void MoveBlock(int key,int event);//控制主要移动物

void MovePen(double x, double y);

void Print\_Maze();//打印地图

void sp(double px,double py,double l);

void StartFilledRegion(double grayScale);

1. model.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include "basic.h"

#include "model.h"

使用了这些外部变量：

extern int Timer3;

extern int Bx,By;//移动物实时位置坐标

extern int Ex,Ey;//出口坐标

extern int Map[L][L];//记录迷宫地图

extern int heat;//状态量，记录游戏是否胜利

extern int Level;//记录关卡

extern int lose;//状态量，记录游戏是否结束

extern int boom;//炸弹数量

extern double life;

使用了这些外部函数：

void DrawFail(int x, int y,double size);

void DrawLine(double dx, double dy);

void drawRectangle(double x, double y, double w, double h, int fillflag);

void EndFilledRegion(void);

void MovePen(double x, double y);

void SetPenColor(string color);

void StartFilledRegion(double grayScale);

1. move.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include "basic.h"

#include "move.h"

使用了这些外部变量：

extern int trap[2];

extern int trace[2][2];

extern int Bx,By;//移动物实时位置坐标

extern int Ex,Ey;//出口坐标

extern int Map[L][L];//记录迷宫地图

int Go=0;//状态量，判断是否进入一往无前模式

使用了这些外部函数：

void DrawWall(int px,int py,double size);

1. rank.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include"basic.h"

#include"rank.h"

#include"ui.h"

使用了这些外部变量：

struct Ranking{

char Name[100];

int score;

struct Ranking \*Next;

};

extern struct Ranking \*Head,\*tail;

extern int link;//状态量，确认是否建立链表

extern int rank1;//状态量，判断排名是否按下

使用了这些外部函数：

int button(int id, double x, double y, double w, double h, char \*label);

void DrawRankFrame(double x,double y);

void SetEraseMode(bool mode)

void DrawTextString(string text);

void MovePen(double x, double y);

1. ui.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include"basic.h"

#include"ui.h"

使用了这些外部函数：

void DrawArc(double r, double start, double sweep);

void DrawLine(double dx, double dy);

void EndFilledRegion(void);

void MovePen(double x, double y);

void SetPenColor(string color);

void SetPenSize(int size);

void StartFilledRegion(double grayScale);

1. way.c

包含了必要的系统库、libgraphics 和 simpleGUI 以及

#include "basic.h"

#include "way.h"

使用了这些外部变量：

extern int All\_step;//记录坐标编号

extern int All\_the\_way[L\*L][2];//记录寻路过程中所有点的坐标

extern int All\_way\_visit[L][L];//每个点的状态 判断是否输出过

extern int Button\_find\_all\_the\_way;//状态量，判断"Find all the way"是否被执行

extern int Button\_find\_way\_down;//状态量，判断"Find the way"是否被执行

extern int Bx,By;//移动物实时位置坐标

extern int Get\_the\_end;//状态量，判断是否到达终点

extern int Map[L][L];//记录迷宫地图

extern int min;//最小路

extern int mseconds1000;//寻路过程显示时间间隔

extern int mseconds500; //路径显示计时器时间间隔

extern int no\_way;//状态量。判断是否有路

extern int path1[L\*L],path2[L\*L];

extern int road[2000][2];//记录

extern int step;//行走顺序

extern int Time\_i;

extern int Time\_j;

extern int Timer1;//计时器1的状态

extern int Timer2;//计时器2的状态

extern int visit[50][50];//记录是否访问

使用了这些外部函数：

void Block(int x,int y,int color,double size);//画出边长为小方块 根据数组坐标转化 带边框

void DrawTextString(string text);

void MovePen(double x, double y);

void SetEraseMode(bool mode);

void startTimer(int id,int timeinterval);

# 部署运行和使用说明

## 编译安装

**编译环境：**

Dev C++ 5.5.3 / 5.11

**安装步骤：**

1.将压缩包解压，保证其中所有文件仍处于一个文件夹中，不要私自移动

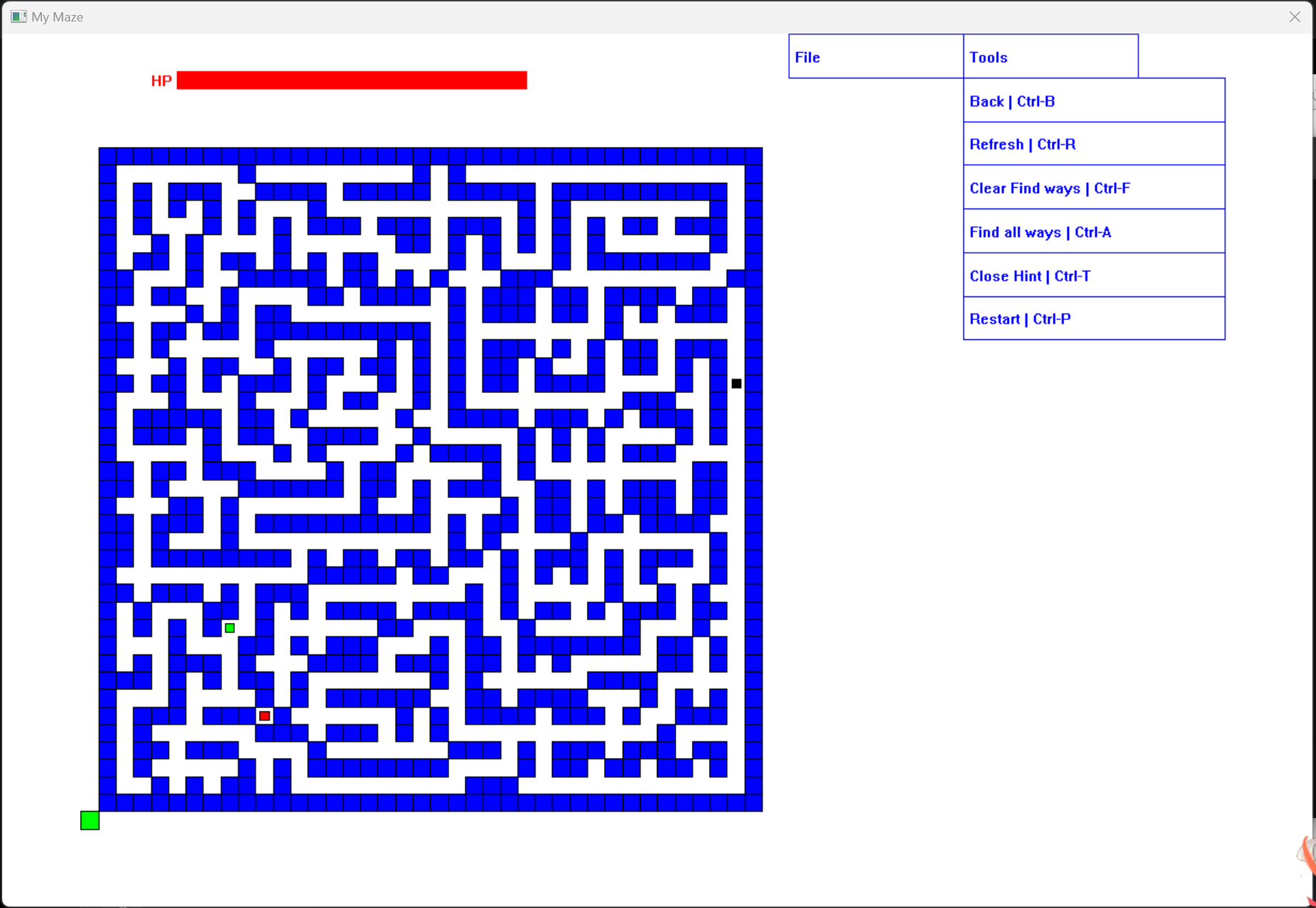
2.鼠标左键双击，打开文本编辑器.dev 工程文件

3.将libgraphics和simpleGUI图形库在包含路径中

4.打开文件后，点击运行中的编译运行按钮，编译并运行程序

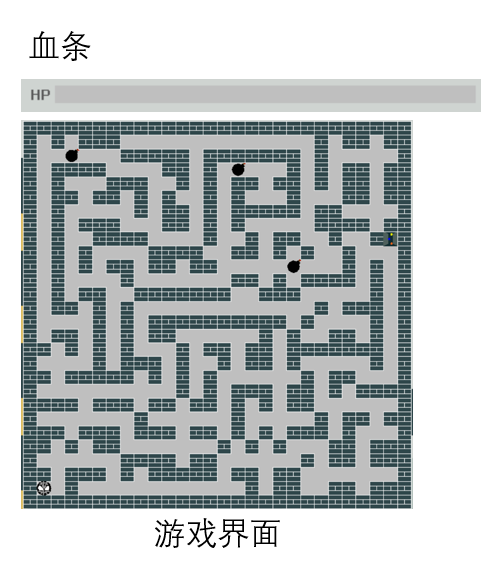
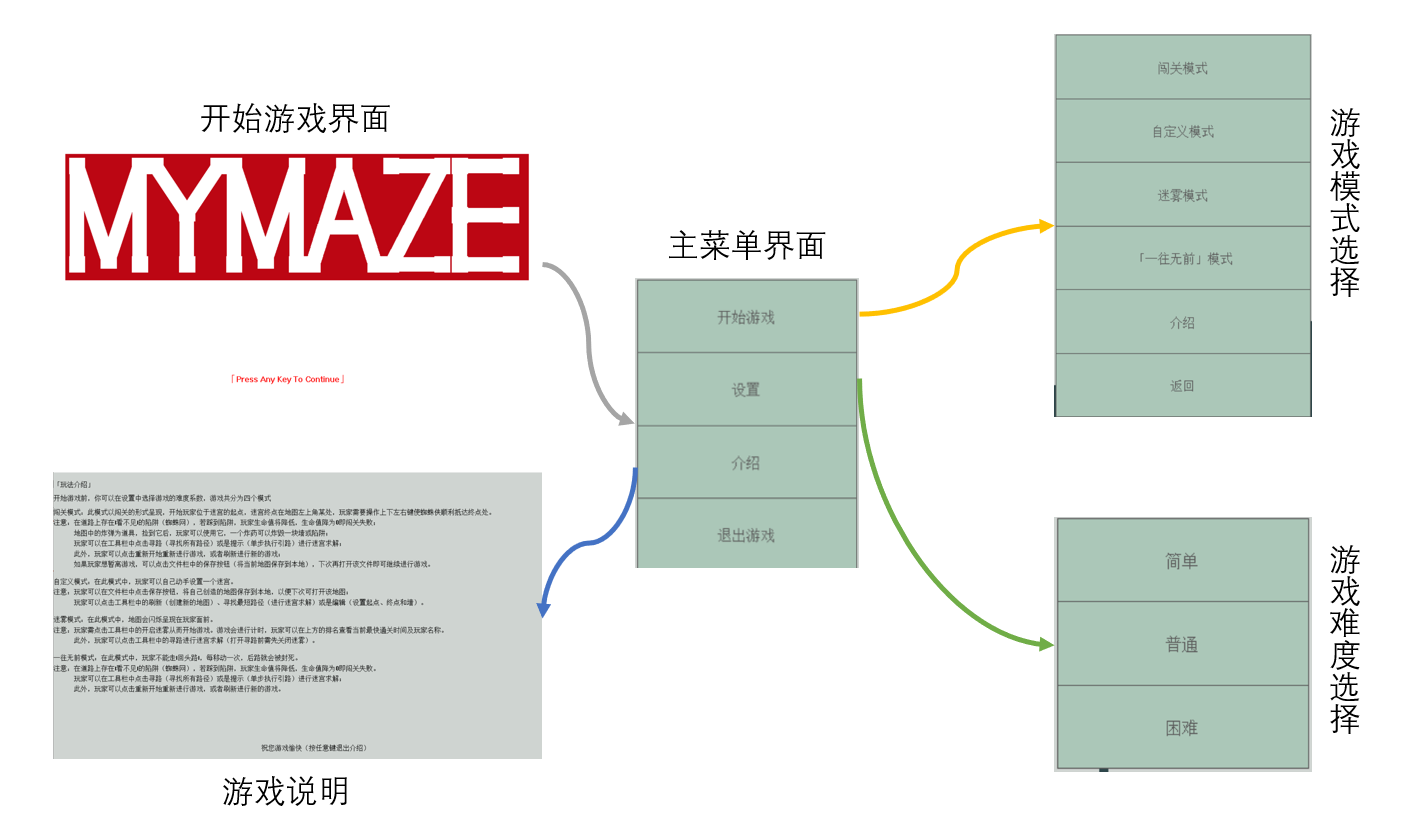
5.迷宫游戏即可正常运行

## 运行测试



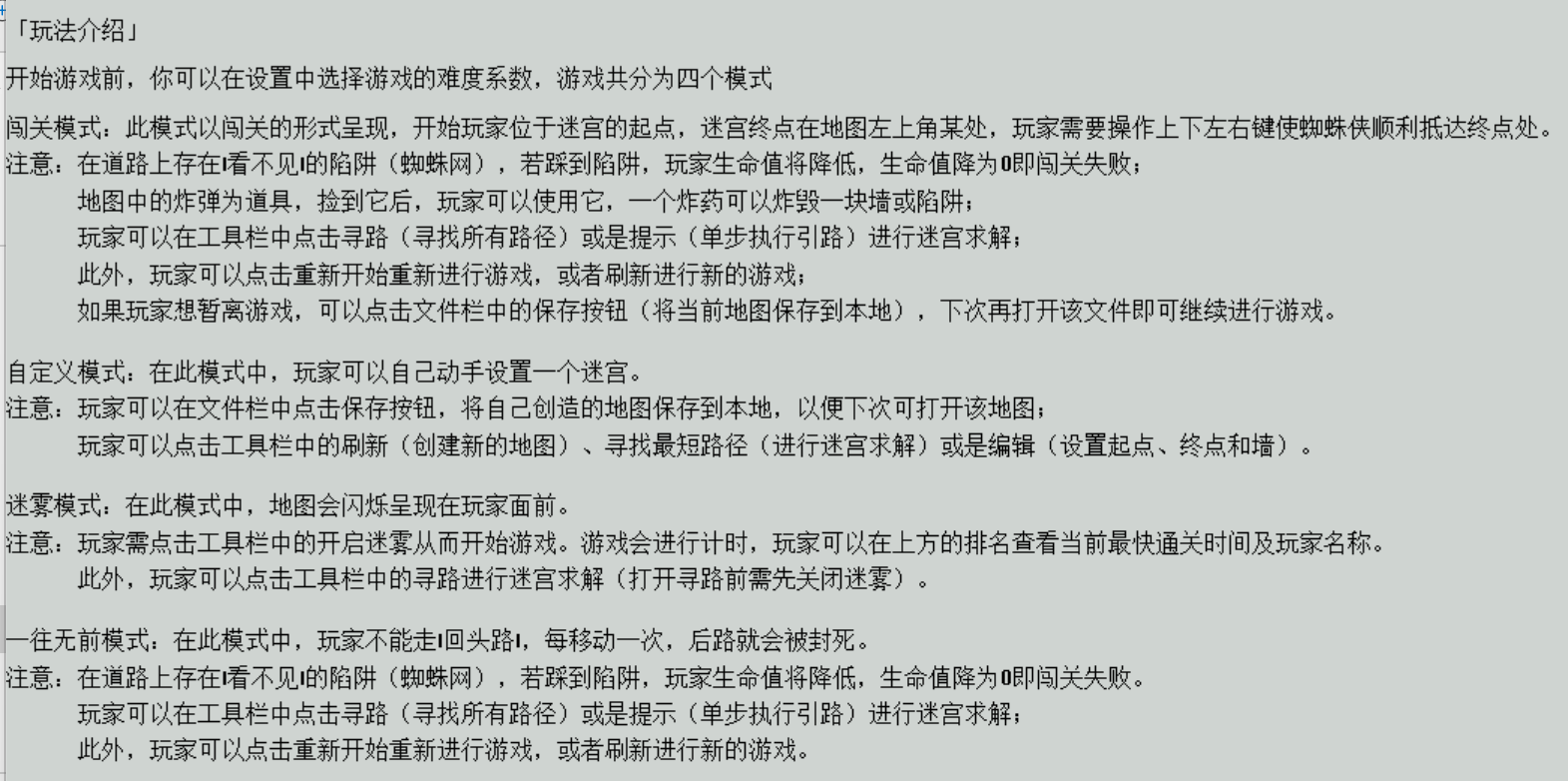
试玩时发现[一往无前]模式中，无路可走的时候没有被判定为失败

## 用户使用手册



操作方法：上下左右键控制蜘蛛侠移动

胜利条件：到达终点且生命值大于零



# 团队合作

## 开发计划

|  |  |  |
| --- | --- | --- |
| 时间 | 人员 | 内容 |
| 第一周 | 所有小组成员 | 进行大程选题要求的研读分析，确定了迷宫游戏 |
| 第二周 | 学生1、学生2 | 根据各功能模块构造函数进行编程 |
| 第三周 | 所有小组成员 | 优化游戏界面和玩法，进行多文件的组织 |
| 第四周 | 所有小组成员 | 对本次开发进行总结，并进行大报告的撰写 |

## 编码规范

1. 变量命名

变量命名采用下划线命名法。

变量示例：

int no\_way=0;//状态量。判断是否有路

int Get\_the\_end=0;//状态量，判断是否到达终点

1. .h 文件格式

条件编译示例：

#ifndef DISPLAY\_H

#define DISPLAY\_H

/\*

.h 文件主体

\*/

#endif

1. 预处理指令

引用示例：

#include "graphics.h"

1. 函数声明/函数定义

左大括号在函数名下一行；

函数示例：

void Button\_Main\_Menu()

{

……

}

函数命名采用下划线命名和帕斯卡命名法。

1. 空格的使用

（1）关键字之后留空格，以便于辨析关键字。

（2）函数名后不留空格，紧跟左括号“(”，以与关键字区别。

（3）赋值操作符、比较操作符、算术操作符、逻辑操作符、位操作符，如“=”、

“+=”、“>=”、“<=”、“+”、“\*”、“%”、“&&”、“|”、“<<” 、

“^”等二元操作符的前后加空格。

1. 缩进的使用

在组员编写的代码中统一采用 Tab 作为缩进方式。在 libgraphics 和 imgui 中由于其缩进格式不明晰，同时存在 Tab 和空格，因此不加以要求。

## 任务分工

学生1：BFS以及DFS寻路，实现可视化寻路，迷雾模式实现，排名系统实现，绘制背景以及人物，完成人物移动等基础功能。

学生2：各按键与菜单的相应功能函数；各界面布局；自定义模式与一往无前模式的开发；首页及终点的绘画。

学生3：游戏说明；游戏中墙、蜘蛛网、胜利和失败、排名框的绘制

## 个人遇到的难点与解决方案

### 5.4.1 学生1

1） 寻路算法的选择

由于地图是通过深度优先算法所生成的，地图上的可行走路线也较为有限，所以我起初选择了DFS来寻找所需要的路径。这个算法在实现随机地图的情况下没有什么大的问题。但是当在自定义模式下，当玩家只放置少量的墙时，就会有近似于无穷多条路能够到达，为了方便显示，于是修改了DFS算法，希望去找到最短的路径，但是由于这个算法在较大地图中的复杂度实在是太高了，会使得程序运行卡顿，但是他在随机地图上面的可视化寻路的效果很好。于是我便选择不同地图采取不同的寻路方式，在随机生成的地图中仍然保留DFS来更好的可视化寻路的过程。对于自定义地图则采取了BFS寻路方式来找路径。这样便很好的解决了这个问题。

2）迷雾模式中回调函数

迷雾模式的根本原理是在计时器运行的同时能够完成正常的走迷宫功能，同时根据定时器的时间的不同，显示不同的地图状态：每间隔一秒切换迷雾与地图，同时在旁边时刻有一个定时器。当正常写完以后发现，在游戏运行是我鼠标以及键盘的移动都会干扰地图和时间的显示，这个是由于只有一个display函数，所有的显示功能都在其中，在上一个函数还走完时，鼠标或者键盘的移动引发了回调函数，使得再一次进入dispaly中去，于是便影响了该函数的完整功能。于是为了解决这个问题我又创建了一个新的显示函数，这个函数主要是负责对于计时器方面的显示：迷雾与计时。于是两个显示函数相互独立，没有影响，这样便能正常游戏了。

### 5.4.2 学生2

1） 问题：自定义模式中，点击“编辑”按钮后理应可以在地图区域开始编辑，实现“点哪亮哪”，但实际在地图区域无论如何点击均无法出现相应元素。

解决方案：由于simplegui库中button函数的设定，如果将button作为if语句的判断条件，只会在按下的一瞬间执行一次if内的语句，从而if语句中放入后续编辑的需要鼠标回调参数的相关操作便变得无效了，因为按下按键的那一瞬间鼠标的位置不可能在地图中，从而无法产生相应元素。为解决此问题，我引入了新的变量作为状态量，用来判断编辑功能是否打开，并将原先以button作为判断条件的if语句改为以新变量作为判断条件，而button作为另一个if语句的判断条件用来改变新变量的值，也即编辑状态。这样便可以保证编辑操作可一直保持。

2）问题：在各个模式中点击“返回”键，理应界面呈现为模式选择界面，但原先模式的地图或按键等仍残留，动动鼠标或者按下键盘才能消失，即使是在“返回”键中手动加入DisplayClear()与display()，不等鼠标回调或键盘回调的响应，但依旧不见效。

解决方案：在display()中各个模式的相关残留显示前加上对界面状态量的判断。因为在按下“返回”键的瞬间，虽然界面状态量已经被改变，但由于键盘回调display()会再次被执行，因而会再最后一次显示一遍原先界面的同时也显示当前新界面。

### 5.4.3 学生3

1) StartFilledRegion和EndFilledRegion函数判断看上去封闭图形时总是报 错图形未封闭

DrawLine函数中的参数不能用math.h中函数所得的小数结果，应该直接用坐标系中实际移动的x，y作为参数带入进行绘图

2）游戏说明只能在点击相应button并不移动鼠标的情况下显示

将说明从Button\_MenuList函数移动到display函数中并设置相应参数判断是否显示

## 合作总结

2023年5月13日：开会进行讨论并分配任务

2023年5月20日：讨论完成进度

2023年5月23日：初步完成功能实现

2023年5月26日：实现并优化各模式玩法

2023年5月28日：优化游戏界面

2023年6月3日：开始撰写报告

总结：通过有效沟通、合理规划，共同合作，我们成功地完成了这个C语言大程序，同时也学到了很多宝贵的经验和教训。我们相信这次经历将对我们未来的项目合作和个人发展产生积极的影响。

## 收获感言

团队：

体会和心得：

* 团队合作的重要性：团队合作是成功项目的关键。我们学会了相互协作、共享知识和经验，互相支持和鼓励。通过有效的沟通和协调，我们能够解决问题、克服困难并实现共同目标。
* 规划和分工：在项目开始之前，我们制定了详细的计划和时间表。我们明确了每个成员的职责和任务，并根据每个人的技能和兴趣进行了合理的分工。这有助于提高效率并确保项目按时完成。
* 代码设计和结构：良好的代码设计和结构对于项目的可维护性和扩展性至关重要。我们学会了使用模块化的方法来组织代码，使用合适的数据结构和算法来解决问题。这使得代码更加清晰、易读和易于维护。
* 测试和调试：在开发过程中，我们重视测试和调试。我们试玩了多次游戏，通过不断调试和修复错误，我们保证了程序的正确性和稳定性。

经验和教训：

* 提前规划和需求分析：在项目开始之前，我们应该充分规划和分析项目的需求。这有助于明确目标、确定功能和制定合理的开发计划。如果需求不清晰或者变化频繁，会导致项目进度延误和功能不完善。
* 注重代码的可读性和可维护性：在开发过程中，我们应该注重编写清晰、易读和易于维护的代码。良好的命名规范、适当的注释和模块化的设计有助于代码的理解和维护。这样可以减少错误和提高开发效率。
* 持续学习和提升技能：在项目开发中，我们意识到编程领域的知识和技能是不断发展和更新的。我们应该保持学习的态度，及时了解新技术和工具，并应用到项目中。持续学习和自我提升可以提高我们的编程能力和解决问题的能力。
* 有效的沟通和协调：良好的沟通和协调是团队合作的关键。我们应该及时交流进展、遇到的问题和需求变化，确保每个成员都了解整个项目的情况。有效的沟通可以减少误解和冲突，提高团队的效率和合作质量。

通过这次迷宫游戏项目的开发，我们不仅实现了一个有趣的游戏，还获得了宝贵的经验和教训。这些经验将指导我们在未来的项目中更好地合作、编写高质量的代码和实现成功的项目。

个人：

周歆仪：窗体底端

学生3：

作为小组成员参与编写迷宫游戏，我获得了很多宝贵的经验和收获。下面是我的收获感言：

1.团队合作：在项目中，我与小组成员一起讨论、规划和分配任务，确保项目按时完成。通过这个过程，我学会了如何有效地与团队合作，倾听和尊重他人的意见，以及在面对挑战时如何协作解决问题。

2.编程技能提升：在这个大程序的编写过程中，我不仅巩固了自己的C语言基础知识，还学到了许多新的编程技巧和设计模式。通过与小组成员的交流和分享，我了解了不同的编程思路和方法，提高了自己的编程能力。

3.项目管理：在编写迷宫游戏的过程中，我学会了如何进行项目管理。我学会了制定计划、设定里程碑和分解任务，以及如何有效地管理时间和资源。这些技能对我未来的工作和学习中都非常有用。

4.解决问题的能力：在编写大程序时，我们遇到了一些问题和挑战。有时候代码会出现bug，有时候我们需要找到更好的解决方案。通过与小组成员的合作和研究，我们成功地解决了这些问题，并最终完成了迷宫游戏。

总的来说，参与编写迷宫游戏是一个非常有价值的经历。我不仅学到了很多技术知识，还培养了团队合作、项目管理、问题解决和自律性等重要的软技能。这些收获将对我未来的学习和职业发展产生积极的影响。

学生2：

设计并成功完成迷宫游戏的过程是一个充满收获的旅程。以下是我在成功设计迷宫游戏后的一些收获感言：

1. 创造力和想象力：迷宫游戏的设计激发了我的创造力和想象力。从设计迷宫结构到制作机关和道具，我不断挑战自己的创造力，为玩家创造了一个独特而令人着迷的游戏世界。

2. 用户体验：我十分注重玩家的体验，致力于设计一个令人满意和引人入胜的游戏。通过不断自行试玩和优化，我学会了如何提供流畅的游戏操作、直观的界面以及恰到好处的难度，以确保玩家获得最佳的游戏体验。

3. 团队协作：迷宫游戏的设计需要团队成员之间的紧密合作。我学会了有效地沟通和分工合作，充分发挥每个人的优势。团队协作的力量让我们得以高效克服挑战、解决问题，并取得了出色的成果。

4. 迭代和改进：在设计迷宫游戏的过程中，我们经历了多次迭代和改进。通过反复测试和收集内测用户反馈，我们不断调整和改进游戏的内容和机制。这个过程让我们明白了不断改进和适应的重要性，以确保游戏的质量和可玩性。

5. 成就感和满足感：最终完成并发布迷宫游戏给我们带来了巨大的成就感和满足感。看到玩家乐在其中、挑战谜题并享受游戏的过程，是我最大的动力和奖励。这种成就感激励着我继续追求更好的软件设计开发。

设计迷宫游戏是一个富有挑战性和创造性的过程，我通过这个项目获得了宝贵的经验和技能，巩固了对C语言程序设计的掌握。这次经历不仅提高了我的团队合作和设计能力，也为我未来在软件工程专业打下了坚实的基础。

学生1：

编写迷宫游戏是一个有趣而具有挑战性的任务，我在这个过程中学到了很多。首先，设计迷宫的关卡需要考虑到玩家的体验。迷宫应该有足够的复杂性，但也不能过于困难，以免让玩家感到沮丧。平衡关卡的难度是一个重要的考虑因素。

其次，编写迷宫游戏需要有良好的逻辑思维和规划能力。我发现在设计迷宫的时候，先在纸上绘制草图或者使用流程图可以帮助我更好地组织思路。这样可以避免出现错误或者漏洞。

此外，测试是不可或缺的一步。在编写迷宫游戏时，我经常会自己尝试通关，以确保关卡的可玩性和逻辑正确性。此外，让其他人尝试玩游戏并收集反馈也是很有帮助的。这样可以发现可能存在的问题并进行改进。

编写迷宫游戏也需要耐心和坚持。有时候可能会遇到一些技术难题或者设计上的困惑，但是要相信自己的能力，并坚持解决问题。与此同时，不断学习和改进也是非常重要的。

综合来说，我对自己在编写迷宫游戏方面的表现感到满意。我能够设计出有趣且具有挑战性的关卡，同时注重细节和玩家体验。通过这个项目，我不仅提高了自己的编程技能，还锻炼了自己的逻辑思维和问题解决能力。不过，我也意识到还有很多可以改进的地方，比如更加注重用户反馈和进行更全面的测试。我会持续学习和提升自己，为编写更好的迷宫游戏而努力。