****

**《程序设计课程实践》综合项目实验报告**

# 项目名称：迷宫游戏

**团队成员：（19151633 应宇杰 19151132 于跃）**

完成时间 2021 年 5 月

# 迷宫游戏项目实验报告

## 团队成员组成及分工

|  |  |  |
| --- | --- | --- |
| **学号** | **姓名** | **详细任务分工** |
| 19151132 | 应宇杰 | 自动走迷宫 |
| 19151132 | 于跃 | 跳转，主动走迷宫 |
|  |  |  |

## 2．开发背景

## 迷宫（希腊语：λαβύρινθος，拉丁转写：labyrinthos）指的是充满复杂通道，很难找到从其内部到达入口或从入口到达中心的道路，道路复杂难辨，人进去不容易出来的建筑物。

使用c语言开发一个走迷宫的程序。迷宫的宽和高、迷宫矩阵、

迷宫内的怪兽、迷宫的入口和出口从文件读入。可以实现手动或自

动两种模式走完迷宫。走迷宫时，规定时间内未走出迷宫或碰到怪

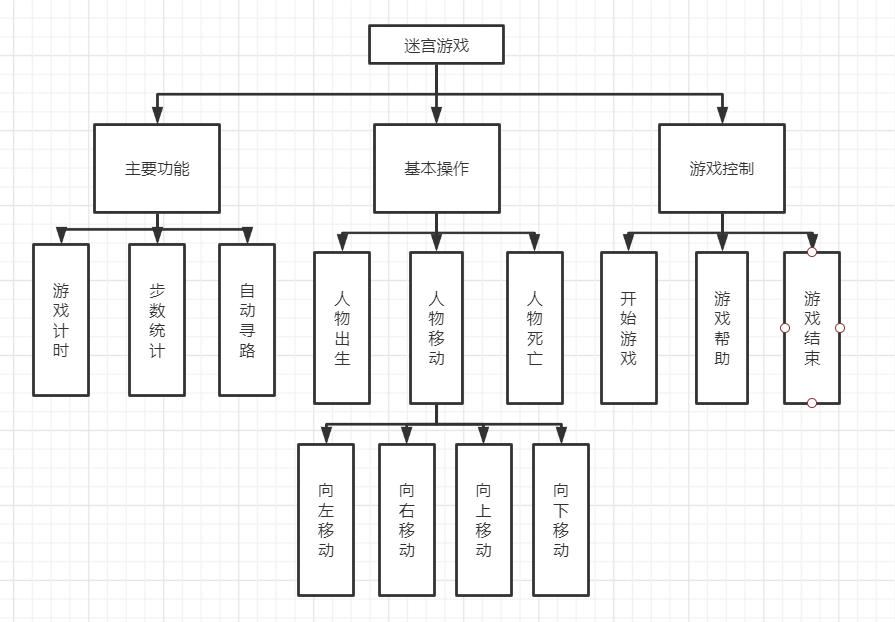
兽，游戏结束；在规定时间内走出迷宫，井没有碰到怪兽，游戏通

关。可以考虑提升游戏难度或者游戏等级设计，也可以自行设计更

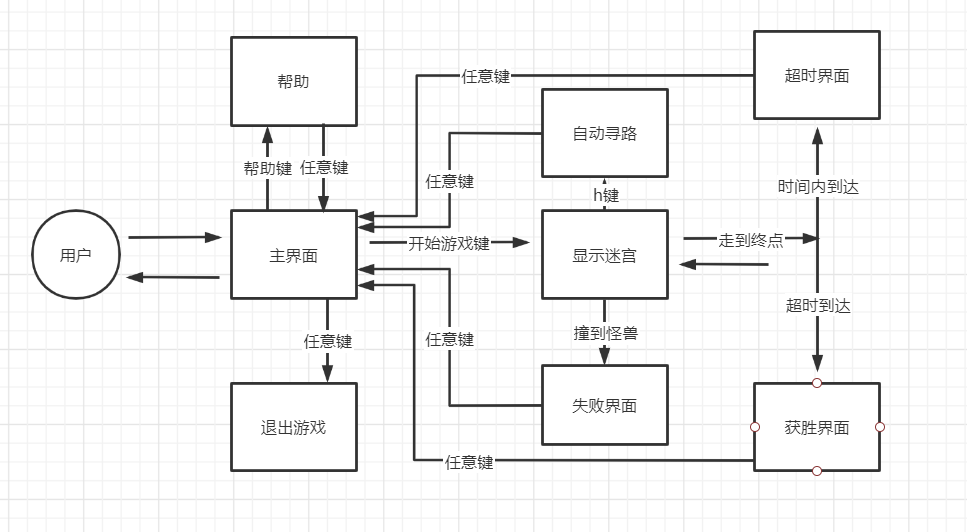
具有创新性的游戏功能。

## 3.系统功能设计

### 3.1系统功能模块设计



### 3.2 系统业务流程设计



## 4.项目创建

### 4.1系统开发环境要求

本项目的开发及运行环境要求：

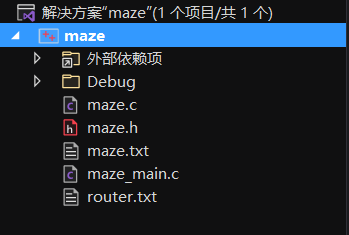
操作系统： win10

开发工具：visual studio

开发语言：c语言

### 4.2 项目创建过程

创建空项目后创建如下源文件



## 5.预处理模块设计

文档完成人：于跃

### 5.1 文件引用

#include <stdio.h>

#include <Windows.h> //命令行定位光标，清屏

#include<conio.h> //getch函数

#include<time.h> //clock函数计时

#include<stdlib.h>

#include<string.h>

### 5.2宏定义

无

### 5.3定义全局变量

typedef struct { //人物当前位置

int i;

int j;//数组下标

} Point;

FILE\* fp1, \* fp2;

Point point; ////人物所在位置结构体

int m = 0;//二维数组第一维

int n = 0;//二维数组第二维

errno\_t err; //返回fopen\_s结果（就是int）

int \*\* maze; //迷宫二维数组

FILE\* fp,\*fpp; //文件指针

int stepcount = 0;//步数

clock\_t starttime, endtime;//时间

int totaltime; //总体用时

int map[20][20];//代表迷宫大小

int a, b, t, flag;//di，dj代表结束位置

int si, sj, di, dj; //si，sj代表开始位置

int vis[20][20]; //代表是否走过

int dir[4][2] = { -1,0,0,-1,1,0,0,1 };//方向：上左下右

### 5.4函数声明

（1）void GotoXY(int x, int y); //光标定位

输入参数：光标位置

（2）void Hide(); //隐藏光标

（3）void InitMaze(); //初始化及刷新迷宫

（4）int Move(); //人物移动判定

返回参数：返回是否结束移动的参数

（5）void Win(); //胜利界面

（6）void Lose(); //失败界面

（7）int Menu(); //主页面

返回参数：根据用户选择菜单选项决定游戏的执行

（8）void Help(); //帮助页面

（9）void PrintRoad(); //打印自动寻路路径

（10）void dfs(int x, int y); //深度优先遍历

输入参数：人物开始坐标

（11）void findroad(); //自动寻路算法

（12）void printtime(); //打印寻路路径

文档完成人：于跃

## 6.系统界面设计

利用GotoXY函数定位光标位置，利用printf语句打印内容，分行打印迷宫墙壁，道路，怪兽，终点等。

## 7.游戏主窗体设计

命令行窗口

## 8.游戏逻辑设计

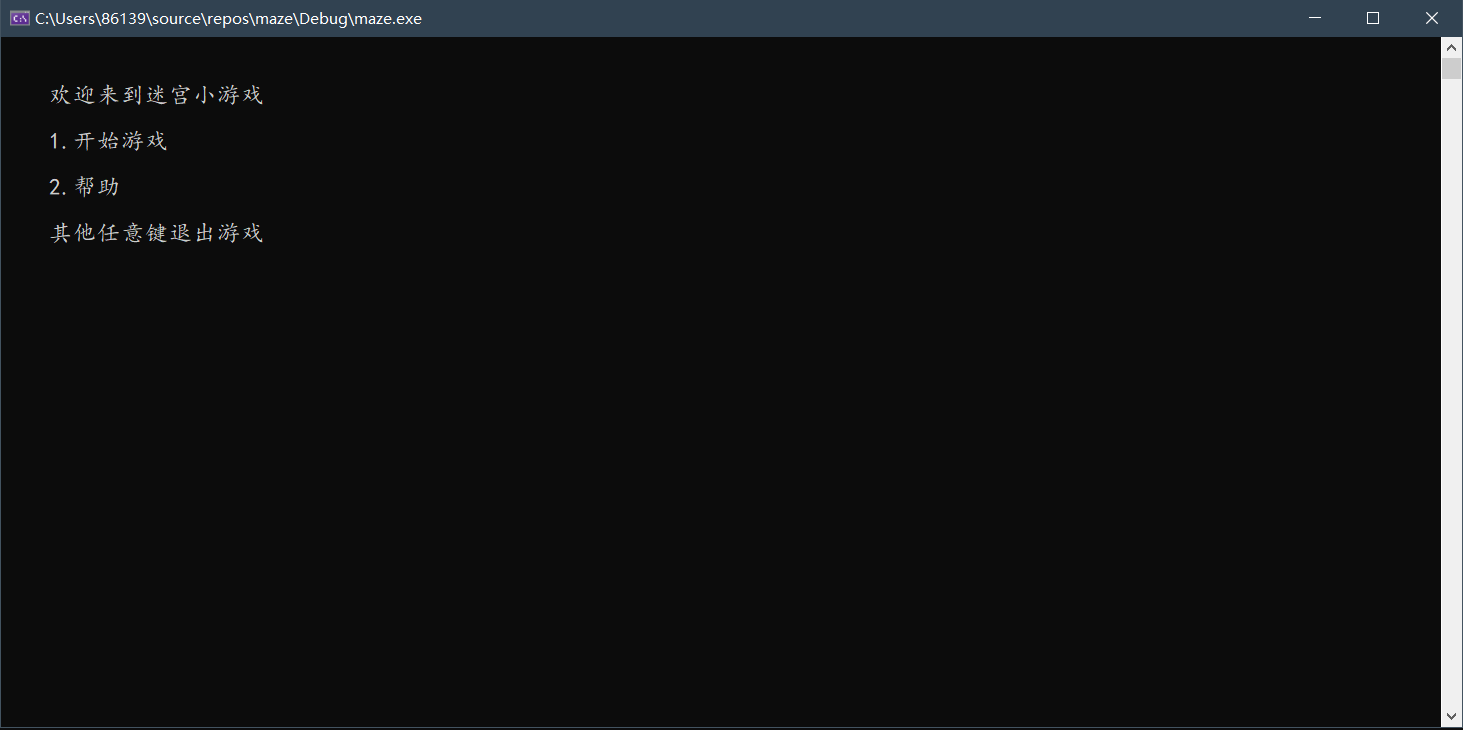
移动改变迷宫二维数组内容，每一步移动及时刷新页面重新打印当前迷宫及任务位置，通过switch case函数读入键盘输入，判断下一步位置，跳转结束页面

## 开始游戏

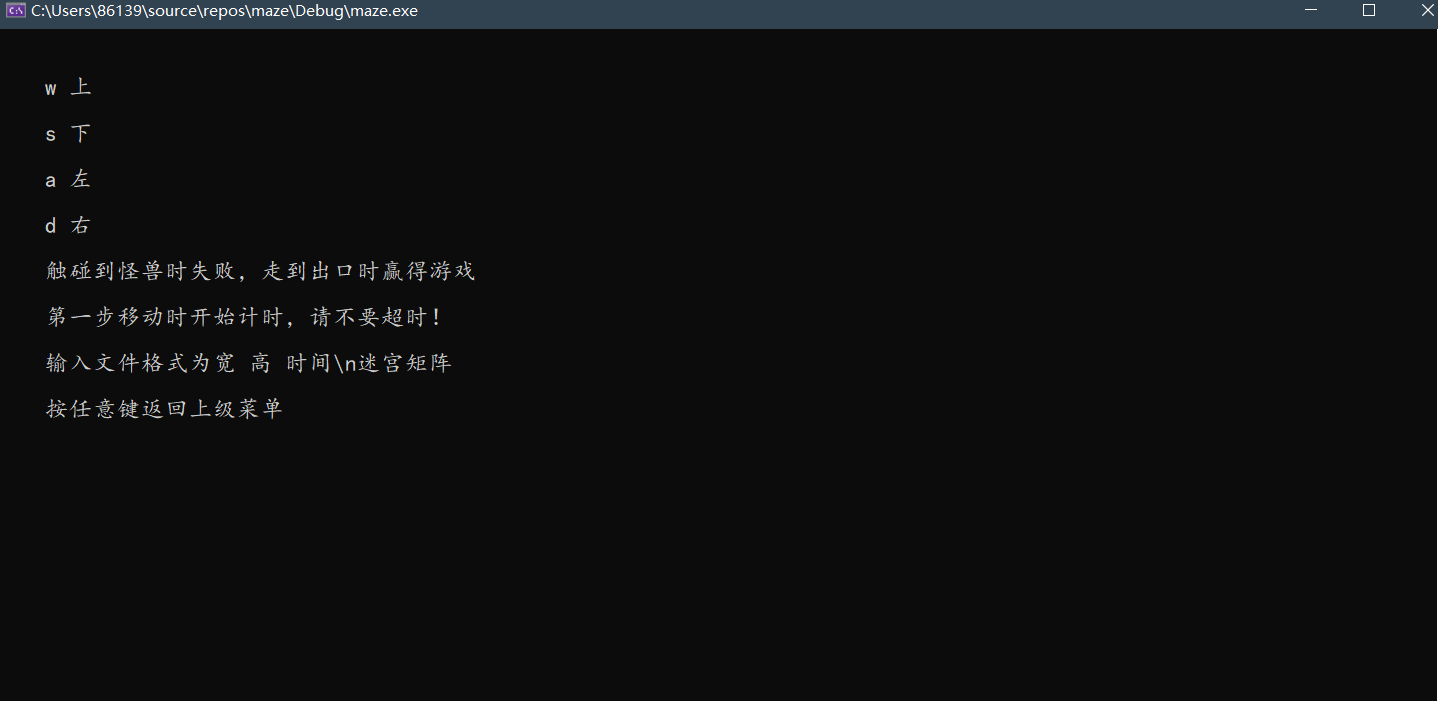
详见10

## 项目运行效果

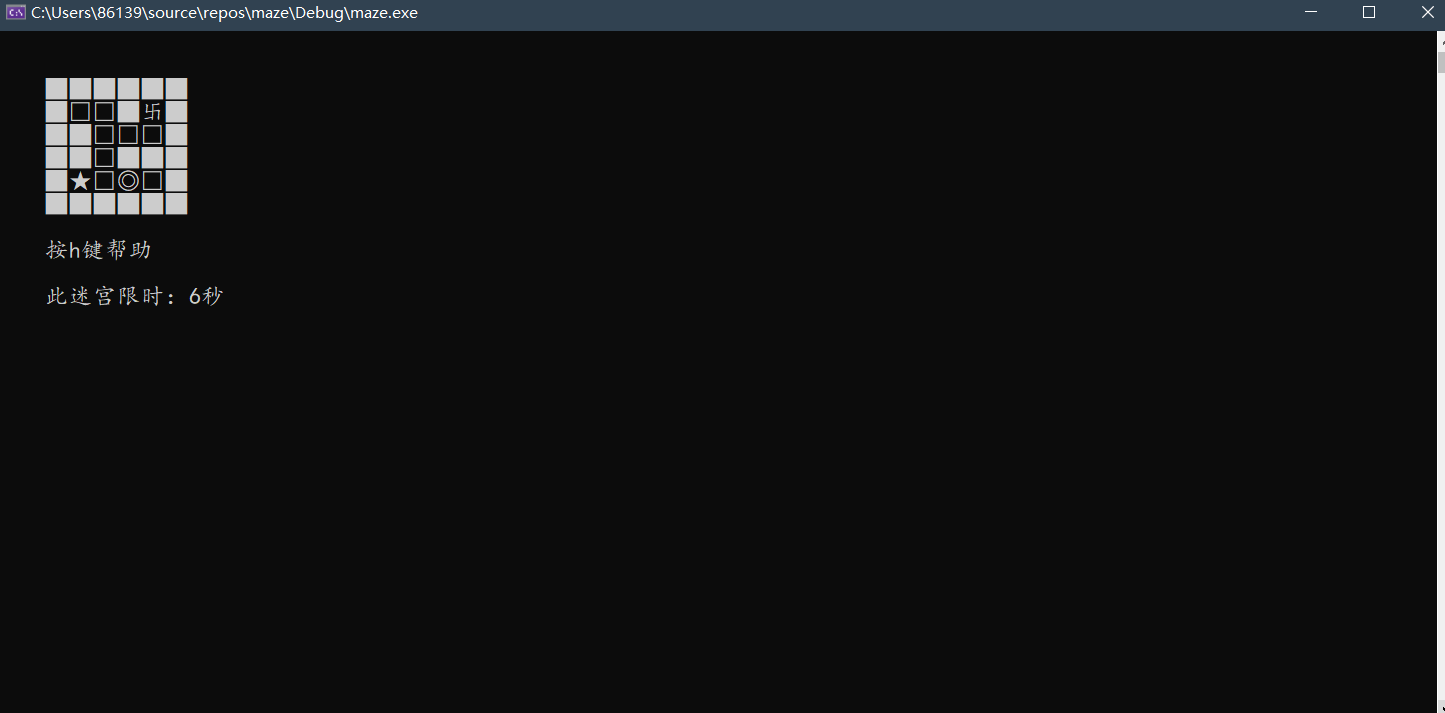
主页面



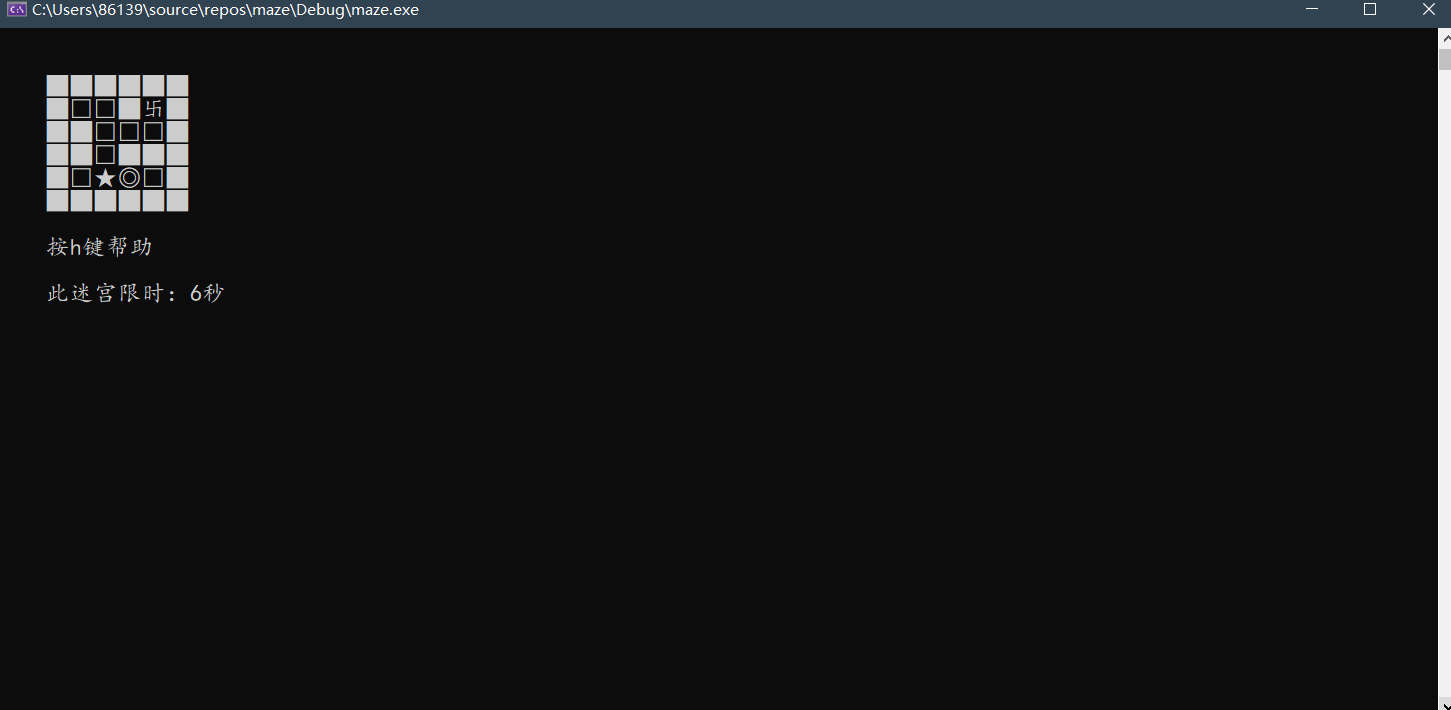
帮助页面



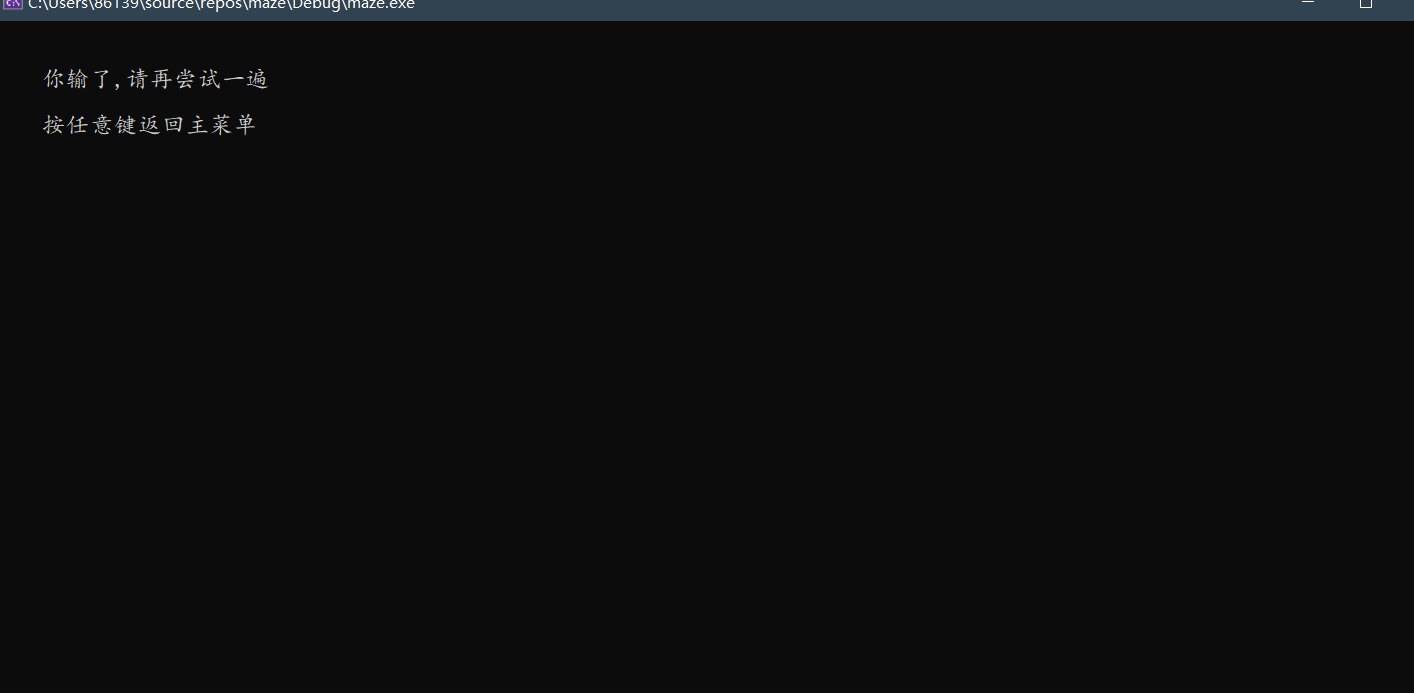
游戏界面

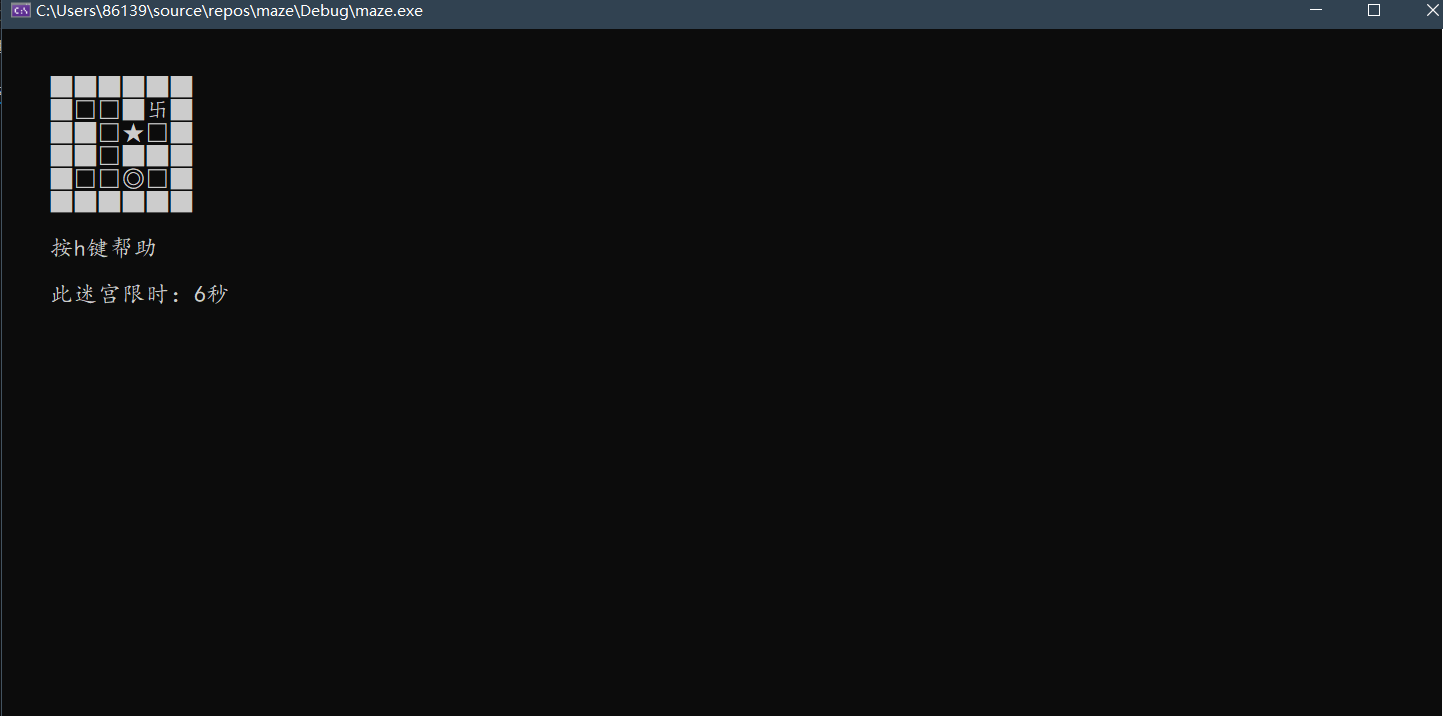


人物移动

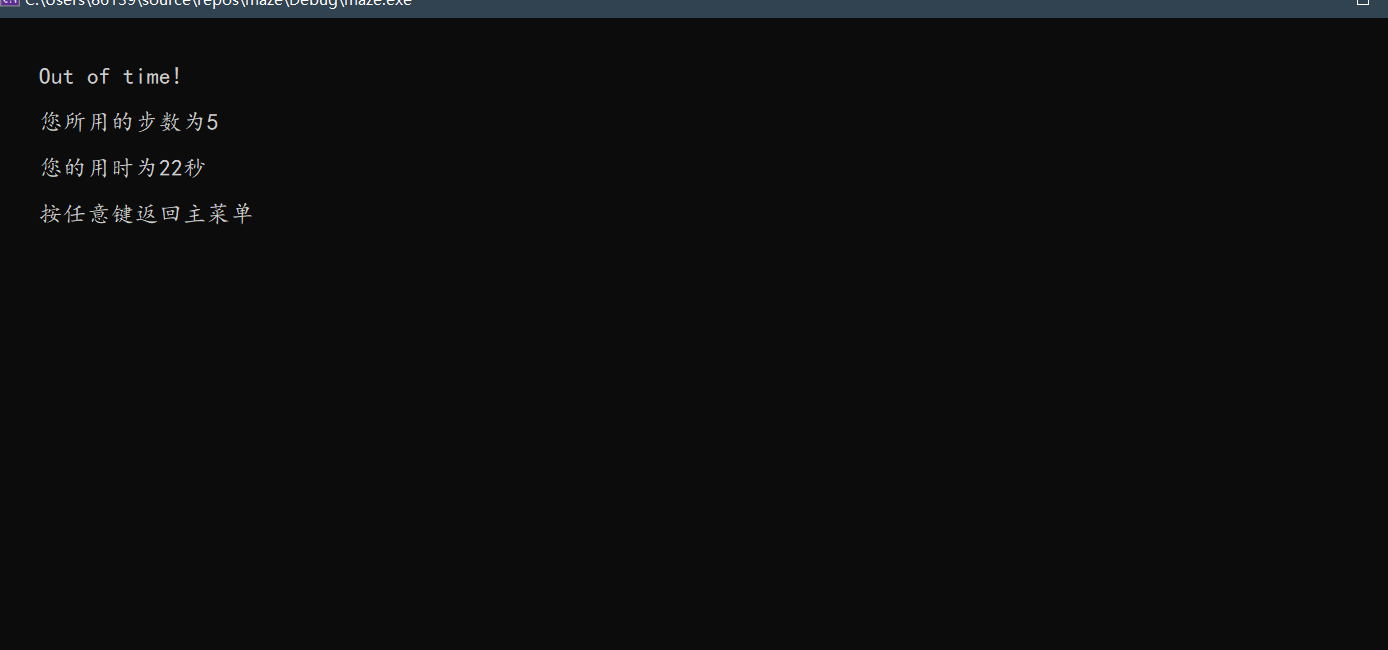


碰到怪兽

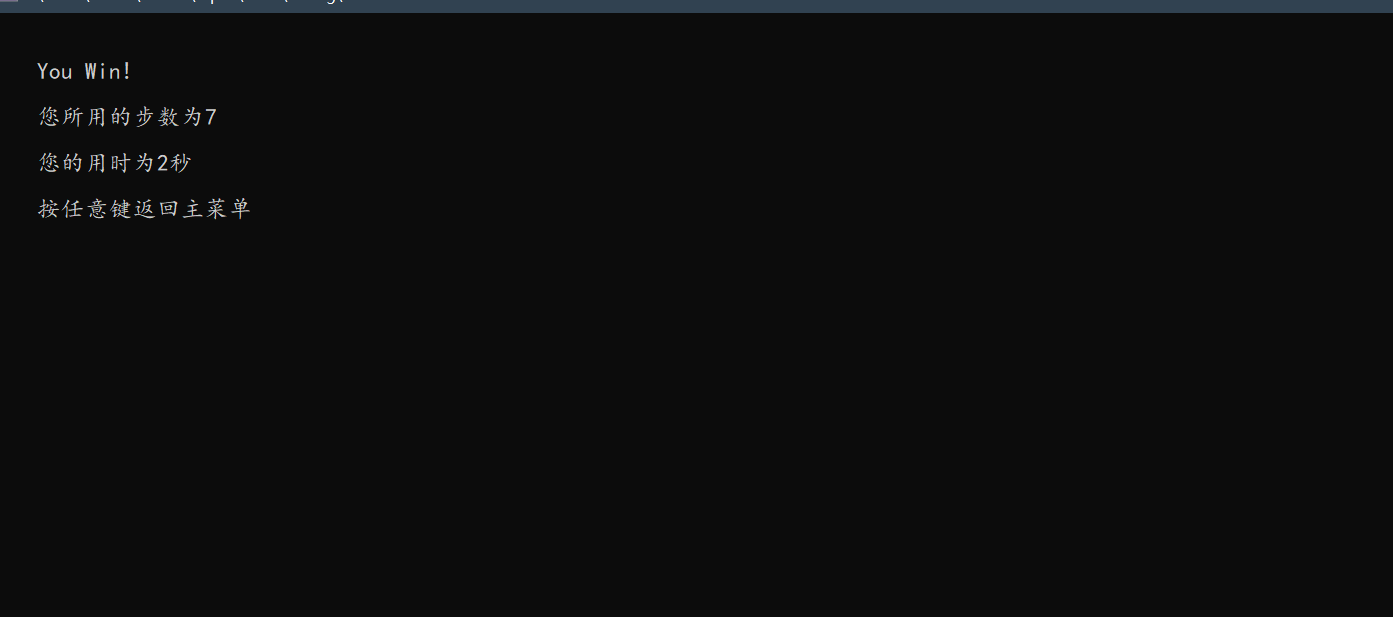




超时到达终点



规定时间内到达终点



文档完成人：于跃

## 11.项目创新点

## 1、在自动走迷宫中用了dfs算法来寻找最佳路线，并且通过图形化界面显示处理

2、用windows.h库中的goto函数来实现定位，并且用漂亮的图形将起点、怪兽、终点展示出来，在帮助文档中详细写了游戏规则。

文档完成人：应宇杰

## 12.收获和建议

组长应宇杰：

在上程序设计课程实践中，我学习了基础的算法，比如经典的查找和排序算法，为我后来的学习打下了坚实的基础，这次我们小组选择了迷宫游戏这个小组作业，在开始任务时一起讨论了任务进度和任务安排，于跃同学承担做迷宫游戏的图形化界面的任务，应宇杰同学实现用深度优先搜索算法实现自动走迷宫的任务，在实现算法的过程中，dfs算法使用错误，把dfs所有走过的路都都当做正确路线显示出来，在多重递归中难以找到错误原因，最后查阅百度资料和自己一步一步dubug终于找到错误实验了正确功能。

在现阶段的产品策划和研发过程中，团队协助必不可少，团队协作能够提高开发效率和产品质量，人员分工精细人能够各司其职沉淀经验，做出高质量的项目，但在日常生活中，任务的细分安排，进度安排混乱，人员任务安排不明确，出现错误互相推卸责任，无法高质量完成产品，所以需要一个任务人员安排的小插件，填好参数最好可以可视化，让人方便的看到成果。

成员于跃：

自己一个人写项目和团队分工写项目是完全不同的体验，自己写项目能对自己的代码有更深的印象，可以及时修改代码中debug出的各种错误，可以稍微不那么注意命名的规范，但团队合作写项目，每个人各司其职，在写好自己代码的同时，还要理解别人写的模块的代码，更要注意自己的命名规范，以及注释的易于理解，让别人更好地理解自己的代码。这只是程序设计实践这门课的大作业，只由我们两个人完成，但显示生产场景中往往是由数十人，甚至于几百个人分工合作共同完成某一项目，对代码标准度，沟通能力的要求都要成倍提高，尤其是整合代码的工作，一个小小的bug可能因为代码的不规范很难找到突破点而浪费掉更多本不应该浪费的时间。通过本次项目我认识到自己还有很大的进步空间。

## 附：源代码清单（每个部分写明是哪位同学完成）

maze.c（共同完成）

#include "maze.h"

FILE\* fp1, \* fp2;

Point point; ////人物所在位置结构体

int m = 0;//二维数组第一维

int n = 0;//二维数组第二维

errno\_t err; //返回fopen\_s结果（就是int）

int \*\* maze; //迷宫二维数组

FILE\* fp,\*fpp; //文件指针

int stepcount = 0;//步数

clock\_t starttime, endtime;//时间

int totaltime; //总体用时

int map[20][20];//代表迷宫大小

int a, b, t, flag;//di，dj代表结束位置

int si, sj, di, dj; //si，sj代表开始位置

int vis[20][20]; //代表是否走过

int dir[4][2] = { -1,0,0,-1,1,0,0,1 };//方向：上左下右

int Menu() {

GotoXY(4, 2); //定位光标位置

printf("欢迎来到迷宫小游戏");

GotoXY(4, 4);

printf("1.开始游戏");

GotoXY(4, 6);

printf("2.帮助");

GotoXY(4, 8);

printf("其他任意键退出游戏");

Hide(); //隐藏光标

char ch;

int result = 0;

maze = NULL;

fp = NULL;

starttime = NULL;

endtime = NULL;

stepcount = 0;

ch = \_getch(); //接收用户输入的菜单选项

switch (ch) { //根据选项设置返回结果值

case '1': result = 1; break;

case '2': result = 2; break;

case '3': result = 3; break;

default:

if (maze != NULL) {

for (int i = 0; i < m; i++)

free(maze[i]);

free(maze);

}

if (fp != NULL) {

fclose(fp);

}

break;

}

system("cls"); //调用系统命令cls完成清屏操作

return result;

}

void GotoXY(int x, int y) {

HANDLE hout;

COORD cor;

hout = GetStdHandle(STD\_OUTPUT\_HANDLE);

cor.X = x;

cor.Y = y;

SetConsoleCursorPosition(hout, cor);

}

/\*隐藏光标\*/

void Hide() {

HANDLE hout = GetStdHandle(STD\_OUTPUT\_HANDLE);

CONSOLE\_CURSOR\_INFO cor\_info = { 1, 0 };

SetConsoleCursorInfo(hout, &cor\_info);

}

void Help() {

GotoXY(4, 2);

printf("w 上");

GotoXY(4, 4);

printf("s 下");

GotoXY(4, 6);

printf("a 左");

GotoXY(4, 8);

printf("d 右");

GotoXY(4, 10);

printf("触碰到怪兽时失败，走到出口时赢得游戏");

GotoXY(4, 12);

printf("第一步移动时开始计时，请不要超时！");

GotoXY(4, 14);

printf("输入文件格式为宽 高 时间\\n迷宫矩阵");

GotoXY(4, 16);

printf("按任意键返回上级菜单");

Hide(); //隐藏光标

char ch = \_getch();

system("cls");

}

void InitMaze() {

system("cls");

Hide();

int y = 2;

GotoXY(4, y);

if (fp == NULL) {

err = fopen\_s(&fp,"maze.txt", "r");

fscanf\_s(fp, "%d%d%d", &m,&n,&t);

}

if (maze == NULL) {

maze = (int\*\*)malloc(sizeof(int\*) \* m);

for (int i = 0; i < m; i++)

maze[i] = (int\*)malloc(sizeof(int) \* n);

for (int i = 0; i < m; i++)

{

for (int j = 0; j < n; j++)

{

fscanf\_s(fp, "%d ", &maze[i][j]);

}

}

}

for (int i = 0; i < m; i++) {

for (int j = 0; j < n; j++) {

if (maze[i][j] == 1) {

printf("█");//wall

}

else if (maze[i][j] == 0) {

printf("□"); //road

}

else if (maze[i][j] == 2) {

printf("◎"); //monster

}

else if (maze[i][j] == 3) {

printf("★"); //begin

point.i = i;

point.j = j;

}

else if (maze[i][j] == 4) {

printf("卐"); //end

}

}

GotoXY(4, ++y);

}

GotoXY(4, ++y);

printf("按h键帮助");

++y;

GotoXY(4, ++y);

printf("此迷宫限时：%d秒",t);

fclose(fp);

}

int Move() {

char ch = \_getch();

switch (ch) {

case'w':

case'W':

if (maze[point.i - 1][point.j] != 1) {

if (maze[point.i - 1][point.j] == 2) {

Lose();

return 0;

}

if (maze[point.i - 1][point.j] == 4) {

endtime = clock(NULL);

Win();

return 0;

}

maze[point.i][point.j] = 0;

point.i--;

maze[point.i][point.j] = 3;

}

break;

case'a':

case'A':

if (maze[point.i][point.j - 1] != 1) {

if (maze[point.i][point.j - 1] == 2) {

Lose();

return 0;

}

if (maze[point.i][point.j - 1] == 4) {

endtime = clock(NULL);

Win();

return 0;

}

maze[point.i][point.j] = 0;

point.j--;

maze[point.i][point.j] = 3;

}

break;

case's':

case'S':

if (maze[point.i + 1][point.j] != 1) {

if (maze[point.i + 1][point.j] == 2) {

Lose();

return 0;

}

if (maze[point.i + 1][point.j] == 4) {

endtime = clock(NULL);

Win();

return 0;

}

maze[point.i][point.j] = 0;

point.i++;

maze[point.i][point.j] = 3;

}

break;

case'd':

case'D':

if (maze[point.i][point.j + 1] != 1) {

if (maze[point.i][point.j + 1] == 2) {

Lose();

return 0;

}

if (maze[point.i][point.j + 1] == 4) {

endtime = clock(NULL);

Win();

return 0;

}

maze[point.i][point.j] = 0;

point.j++;

maze[point.i][point.j] = 3;

}

break;

case'h':

case'H':

PrintRoad();

return 0;

default:

system("cls");

return 0;

}

if (starttime == NULL) starttime = clock();

stepcount++;

InitMaze();

return 1;

}

void Win() {

system("cls");

GotoXY(4, 2);

printf("You Win!");

GotoXY(4, 4);

printf("您所用的步数为%d", stepcount);

GotoXY(4, 6);

totaltime = (int)(endtime - starttime) / CLOCKS\_PER\_SEC;

printf("您的用时为%d秒",totaltime);

if (totaltime > t) {

GotoXY(4, 2);

printf("Out of time!");

}

GotoXY(4, 8);

printf("按任意键返回主菜单");

Hide(); //隐藏光标

char ch = \_getch();

system("cls");

}

void Lose() {

system("cls");

GotoXY(4, 2);

printf("你输了,请再尝试一遍");

GotoXY(4, 4);

printf("按任意键返回主菜单");

Hide(); //隐藏光标

char ch = \_getch();

system("cls");

}

void PrintRoad() {

system("cls");

Hide();

int y = 2;

GotoXY(4, y);

int helper;

err = fopen\_s(&fpp, "router.txt", "r+");

for (int i = 0; i < m; i++) {

for (int j = 0; j < n; j++) {

if(j == n-1) fscanf\_s(fpp, "%d\n", &helper);

else

fscanf\_s(fpp, "%d ", &helper);

if (maze[i][j] == 1) {

printf("█");//wall

}

else if (helper == 2 || maze[i][j] == 3) {

printf("★"); //begin

}

else if (maze[i][j] == 0) {

printf("□"); //road

}

else if (maze[i][j] == 2) {

printf("◎"); //monster

}

else if (maze[i][j] == 4) {

printf("卐"); //end

}

}

GotoXY(4, ++y);

}

fclose(fpp);

char ch = \_getch();

system("cls");

}

void dfs(int x, int y)

{

if (x == di && y == dj) {

flag = 1;

vis[x][y] = 4;

}

if (flag == 1) return;

if (x <= 0 || x > a || y <= 0 || y > b) return;

for (int i = 0; i < 4; i++)

{

int xx = x + dir[i][0];

int yy = y + dir[i][1];

if (vis[xx][yy] == 0)

{

vis[xx][yy] = 2;

dfs(xx, yy);

if (flag == 0)

vis[xx][yy] = 0;

else {

break;

}

}

}

}

//迷宫数字代表的意思：1是围墙,2是怪兽,3是起点,4是终点

void findroad()

{

err = fopen\_s(&fp1,"maze.txt", "r");

fscanf\_s(fp1, "%d%d%d\n", &a, &b);

if (a == 0 || b == 0) return 0;

int i = 1, j = 1;

int sum = 0;

flag = 0;

memset(vis, 0, sizeof(vis));

while (i <= a)

{

while (j <= b)

{

fscanf\_s(fp1, "%d", &map[i][j]);

if (map[i][j] == '\n') j--;

if (map[i][j] == 3) { //如果遇到开始符号

si = i; sj = j;

vis[si][sj] = 1;

}

else if (map[i][j] == 1 || map[i][j] == 2) //如果遇到围墙或者怪兽

{

vis[i][j] = 1;

sum++;

}

else if (map[i][j] == 4) { //如果遇到结束符号

di = i; dj = j;

}

j++;

}

i++;

j = 1;

}

fclose(fp1);

dfs(si, sj);

vis[si][sj] = 2;

if (flag == 1) {

err = fopen\_s(&fp2,"router.txt", "w+");

for (i = 1; i <= a; i++)

for (int j = 1; j <= b; j++) {

if (j == b) {

fprintf(fp2, "%d\n", vis[i][j]);

}

else

fprintf(fp2, "%d ", vis[i][j]);

}

fclose(fp2);

}

}

Maze\_main.c:（于跃）

#include "maze.h"

int main() {

int end = 1, result;

findroad();

while (end) {

result = Menu(); //显示主菜单，并根据用户选择菜单选项决定游戏的执行

switch (result) {

case 1: //选择1表示，开始游戏

InitMaze(); //初始化

while (Move());

break;

case 2: //选择2表示，显示帮助信息

Help();

break;

case 0: //选择0表示，表示结束程序

end = 0;

break;

}

}

return 0;

}:

maze.h:（于跃）

#include <stdio.h>

#include <Windows.h>

#include<conio.h>

#include<time.h>

#include<stdlib.h>

#include<string.h>

typedef struct { //人物当前位置

int i;

int j;//数组下标

} Point;

void GotoXY(int x, int y); //光标定位函数

void Hide(); //隐藏光标函数

void InitMaze();

int Move();

void Win();

void Lose();

int Menu();

void Help();

void PrintRoad();

void dfs(int x, int y);

void findroad();

void printtime();