# 1 项目描述

## 1.1 项目目的

​ 本次的操作系统课程设计旨在让我们通过学习并实现一个最基本的的操作系统，增加对进程管理、内存管理、文件管理等基本组成部分的理解，并了解操作系统是如何发展的出来的，同时也帮助我们熟悉Linux的环境。

## 1.2 开发环境

\* \*\*操作系统：\*\*Ubuntu 64bits\*(使用 VMware虚拟机)\*

\* \*\*操作系统模拟器：\*\*Bochs开源模拟器

\* \*\*代码编辑器：\*\*Visual Studio Code

\* \*\*代码语言：\*\*C语言、汇编语言

## 1.3 项目选题及达成指标

项目选题：完成《Orange's：一个操作系统的实现》项目要求

达成指标：\*\*

\* \*\*B级：\*\*本系统对文件系统进行重新实现，新增代码量达到相关模块代码的一半。

\* \*\*C级：\*\*在参考源码上实现了系统级应用，如磁盘、工具台、进程管理等，通过调用较多系统API实现对系统的检测和控制。

\* \*\*D级：\*\*在参考源码上实现了用户级应用，包括五个小游戏、日历、计算器。

# 2 设计方案

## 2.1 项目概述

​ 本系统以《Orange's：一个操作系统的实现》一书中的源代码为基础，实现了一个简单的操作系统，同时通过调用部分系统API实现了贪吃蛇、扫雷等小游戏，以及日历、计算器等工具以及一个开机动画。

## 2.2 系统功能及使用说明

\* 在`SummerOS`打开终端

\* 输入`bochs`->`6`->`c`

\* 进入`Summer`操作系统

### 2.2.1 开机动画

### 2.2.2 帮助界面/主界面

​ 主界面主要是提供所有能接收到的指令。输入不同的命令可以进入相应功能界面。

​ 回到帮助界面/主界面的方法是在命令行中输入“help”后按下回车。

### 2.2.3 清屏功能

​ 当需要清理屏幕时，输入指令clear按下回车即可。

### 2.2.4 进程管理

runProcessManager() 进程管理

​ 用户输入指令\*\*“process”\*\*，进入进程管理帮助界面。

\*\*实现方法\*\*

进程管理系统为对计算机内的全部进程的调度，展示，修改等控制功能。主要包含的操作有：ps，block，start，quit，help以及clear。该界面包含了一个命令提示帮助界面。

#### 2.2.4.1 ps

\*\*描述：\*\*

用户进入进程管理系统后，输入指令\*\*“ps”，\*\*系统打印出全部进程以及运行状态、优先级等全部信息。

\*\*实现方法\*\*

调⽤show\_all\_Process()⽅法，访问proc\_table[]，依次输出其中进程的各种信息。

#### 2.2.4.2 block

\*\*描述：\*\*

用户进入进程管理系统后，输入指令\*\*“block x”，\*\*系统将关闭第x号进程。

\*\*实现方法\*\*

调⽤block\_a\_Process()⽅法，根据输入的x的值寻找到proc\_table[]的相应的位置，修改进程的优先级和运行状态。

#### 2.2.4.3 start

\*\*描述：\*\*

用户进入进程管理系统后，输入指令\*\*“startx”，\*\*系统将打开第x号进程。

\*\*实现方法\*\*

调⽤start\_a\_Process()⽅法，根据输入的x的值寻找到proc\_table[]的相应的位置，修改进程的优先级和运行状态。

#### 2.2.4.4 quit

\*\*描述：\*\*

用户进入进程管理系统后，输入指令\*\*“quit”，\*\*系统退出进程管理系统。

#### 2.2.4.5 clear

\*\*描述：\*\*

用户进入进程管理系统后，输入指令\*\*“clear”，\*\*后，系统会自动清理屏幕。

\*\*实现方法\*\*

#### 2.2.4.6 help

\*\*描述：\*\*

用户进入进程管理系统后，输入指令\*\*“help”，\*\*后，系统会自动清理屏幕并回到命令帮助界面。

### 2.2.5 文件管理

runFileManage():文件系统

本文件系统模拟Linux的部分简单文件操作，设置初始用户为home，实现了可长期保存的多级目录下的文件系统。主要包含操作有：查看该目录下所有目录及文件名（ls），创建普通文件（touch），创建目录文件（mkdir），删除文件（rm），读文件（rd），写文件（wt），进入多级目录（cd）。

用户通过在1号控制台中输入\*\*“file”\*\*指令进入文件管理系统。

下面是文件系统包含命令帮助界面：

#### 2.2.5.1 quit

\*\*描述\*\*

用户进入文件系统后，输入指令\*\*“ls”，\*\*系统打印出当前⽂件⽬录下的全部⽂件及⽬录。

\*\*实现方法\*\*

调⽤ls()⽅法，找到当前⽬录下的全部⽂件名。

![图片](https://uploader.shimo.im/f/6cl5VcIFbOLchPuZ.png!thumbnail)

#### 2.2.5.2 mkdir

\*\*描述\*\*

用户进入文件系统后，输入指令\*\*“mkdir + filename”，\*\*系统即在当前目录下创建一个新的目录文件并命名为filename。若创建成功，系统提示“DIR created”的消息，若创建失败则系统提示“Failed to create a new directory”的消息。用户可在该子目录下进行等同于根目录的操作。

\*\*实现方法\*\*

根据传入的type类型（I\_DIRECTORY），通过调⽤open()⽅法创建对应类型的新目录，根据⽅法返回值判断是否创建成功并给⽤户发出反馈消息。

![图片](https://uploader.shimo.im/f/YJc0xJvhyCH6o0ZN.png!thumbnail)

#### 2.2.5.3 touch

\*\*描述\*\*

⽤户输⼊“\*\*touch + filename\*\*”指令，在当前⽬录下创建⼀个新的⽂件并命名，若创建成功则系 统提示“file created successfully！”的消息，若创建失败则系统提示“failed to create a new file...”的消息

\*\*实现方法\*\*

根据传入的type类型（I\_REGULAR），通过调⽤open()⽅法创建对应类型的新⽂件，根据⽅法返回值判断是否创建成功并给⽤户发出反馈消息。

![图片](https://uploader.shimo.im/f/NBSDovSOk0yiQtWY.png!thumbnail)

#### 2.2.5.4 remove（rm）

\*\*描述\*\*

用户进入文件系统某一目录后，可以输入\*\*“rm”\*\*在当前目录下进行文件删除操作。若删除普通文件，则输入操作\*\*“rm filename”\*\*，系统执行删除操作并返回是否删除成功标志；若删除目录文件，则输入操作\*\*“rm -r dir\_name”\*\*，系统执行删除目录操作并返回是否成功删除标志。

\*\*实现方法\*\*

通过调用unlink()API实现文件删除，通过递归调用dir\_unlink()实现目录文件的级联删除。

![图片](https://uploader.shimo.im/f/2XlNuRKNpr3vK3k5.png!thumbnail)

#### 2.2.5.5 write（wt）

\*\*描述\*\*

⽤户输⼊\*\*“wt+filename”\*\*指令，系统在定位到⽬标⽂件后显示⽂件路径，之后⽤户输⼊要写⼊⽂件的内容并回⻋，系统显示写入byte数，完成写⽂件操作。

\*\*实现方法\*\*

调用open()⽅法定位⽂件，然后调⽤write()⽅法对⽂件进⾏写操作。

![图片](https://uploader.shimo.im/f/yCbIAkaIMIrMONra.png!thumbnail)

#### 2.2.5.6 read（rd）

\*\*描述\*\*

⽤户输⼊\*\*“rd+filename”\*\*指令，查看该文件内容，若文件已写入内容，则显示文件内容；若文件已创建但为空（未写入内容），则打印空行；若文件未查找到或文件打开失败，则提示用户\*\*“fail to open！”\*\*。

\*\*实现方法\*\*

调用open()⽅法定位⽂件，然后调⽤read()⽅法对⽂件进⾏读操作。

![图片](https://uploader.shimo.im/f/WT9LqHNBvTujOMsr.png!thumbnail)

#### 2.2.5.7 cd

\*\*描述\*\*

⽤户输⼊\*\*“cd+directory”\*\*指令，系统定位到该相对路径所在的目录并检索其是否存在，如果存在则将cur\_dir变量后追加用户所输入的相对路径，在此后进行操作时，控制台屏幕上显示的当前目录为cd后进入的目录，以后操作中涉及的目录均以此为基础进行定位。

\*\*实现方法\*\*

调用open()方法确认要进入的目录是否存在，如果存在则有strcat()函数更新cur\_dir。

![图片](https://uploader.shimo.im/f/yyr6VpjkvHz97k2N.png!thumbnail)

### 2.2.6 游戏及工具

#### 2.2.6.1 游戏界面

用户输入指令\*\*"game"\*\*，进入游戏界面，控制台将显示所有游戏简介及相应指令。

![图片](https://uploader.shimo.im/f/LZrBx9J6SgkyAJlL.png!thumbnail)

#### 2.2.6.2 sudoku（数独）

\*\*操作描述\*\*

\* 用户输入指令\*\*“sudoku”\*\*，进入数独游戏界面，控制台将显示游戏的操作说明，用户按下回车键即开始游戏

![图片](https://uploader.shimo.im/f/eDWk552vFO9kCmdm.png!thumbnail)

\* 用户可以通过指令在数独棋盘中输入或删除数字，每次操作后将打印新的棋盘

![图片](https://uploader.shimo.im/f/wFOChTYMeEsNsvfH.png!thumbnail)

\* 用户成功完成数独时，将显示游戏获胜信息

\*\*功能实现\*\*

\* 采用深度优先搜索，调用getmap()随机生成一个数独棋盘，并随机取一定数量单元格作为初始给定的单元格

\* 调用sudoku\_main()接收用户的指令进行游戏

\* 每次用户完成操作后，调用``printmap()``打印新棋盘

\* 若当前已被填满，则调用``check\_win()``判断用户是否获胜

#### 2.2.6.3 bwchess（黑白棋）

\*\*操作描述\*\*

\* 用户输入指令\*\*“bwchess”\*\*，进入黑白棋游戏界面，控制台将显示游戏的操作说明，用户按下回车键即开始游戏

![图片](https://uploader.shimo.im/f/mF2xIXgQEGdViueu.png!thumbnail)

\* 系统默认玩家持黑子，AI持白子，玩家可以选择输入1或0来选择先手或后手

![图片](https://uploader.shimo.im/f/6QljzWHh4koHw2IR.png!thumbnail)

\* 用户可以通过指令输入将要落子的行列进行落子，每次操作后将打印新的棋盘

![图片](https://uploader.shimo.im/f/lk1Uycr3rIC6en8z.png!thumbnail)

\* 当棋盘下满或双方无法钧无法落子时，系统将判定输赢并计算分数，显示游戏结算信息

\*\*功能实现\*\*

\* 调用Exa()函数计算落子的步数用于最后计算分数

\* 调用Calsore()函数通过计算最后棋子的个数来计算玩家最后的得分

\* 调用show()函数来每次打印棋盘

\* 调用bwchess\_main()函数来接受用户指令进行游戏

#### 2.2.6.4 tic-tac-toe（井字棋）

\*\*操作描述\*\*

\* 用户输入指令\*\*“tic”\*\*，进入井字棋游戏界面，控制台将显示井字棋的操作说明，用户按下回车键即开始游戏

![图片](https://uploader.shimo.im/f/g237H2hX2BiYxJsB.png!thumbnail)

\* 游戏支持两位玩家进行对战，两位玩家依次输入数字1-9（表示落子的位置）进行落子（若输入则可退出游戏小程序）

![图片](https://uploader.shimo.im/f/cuZ8Hjhihkhd3gIH.png!thumbnail)

\* 若其中一位玩家的三颗棋子连成一线，则该玩家获胜；若棋盘已满而无人获胜，则平局，将在屏幕显示最终游戏结果

![图片](https://uploader.shimo.im/f/NSqMS2EmqeZynFcx.png!thumbnail)

\*\*功能实现\*\*

\* 调用``ticTacToe\_main()``输出井字棋棋盘，接收用户输入的指令执行相应操作，并判断游戏最终结果

#### 2.2.6.5 mine（扫雷）

\*\*操作描述\*\*

\* 用户输入指令\*\*“mine”\*\*，进入扫雷游戏界面，控制台将显示游戏选项，用户输入`1`即开始游戏，输入`0`退出游戏

![图片](https://uploader.shimo.im/f/u00f9wMXxvCwU4M3.png!thumbnail)

\* 用户可以通过指令在扫雷棋盘中翻开指定块，每次操作后将打印新的棋盘

![图片](https://uploader.shimo.im/f/GeQew6LfGXcHl5tw.png!thumbnail)

\* 用户成功排除所有雷（10个）时，将显示游戏获胜信息

![图片](https://uploader.shimo.im/f/961I4xYQcFlyqr3R.png!thumbnail)

\* 用户“踩”到雷则游戏失败

![图片](https://uploader.shimo.im/f/2PAeKCZnqOs3i1xi.png!thumbnail)

\*\*功能实现\*\*

\* 调用InitBoard()生成两个棋盘，一个用于显示给用户，一个用于记录哪里埋雷

\* 调用mine\_main()接收用户的指令进行游戏

#### 2.2.6.5 carrycraft（推箱子)

\*\*操作描述\*\*

\* 用户输入指令\*\*“carrycraft”\*\*，进入推箱子游戏界面，控制台将显示游戏的操作说明，用户按下回车键即开始游戏

![图片](https://uploader.shimo.im/f/LiRe8HGehVoqgBq9.png!thumbnail)

\* 玩家通过wasd输入来实现控制人物进行上下左右的行走

![图片](https://uploader.shimo.im/f/fQMMwd60RQEbh7kZ.png!thumbnail)

\* 状态图根据玩家的指令修改相应信息，每次操作后将打印新的地图

![图片](https://uploader.shimo.im/f/55me2i1K6rvRXvou.png!thumbnail)

\* 当所有的箱子都放置到目标地点后或者任务无法移动，系统将判定输赢，显示游戏结算信息

\*\*功能实现\*\*

\* 调用drawmain()来打印地图信息

\* 调用win()函数来判断玩家是否胜利

\* 调用carry\_main()函数来接受用户指令进行游戏

#### 2.2.6.6 calendar（日历）

\*\*操作描述\*\*

\* 用户输入指令\*\*“calend”\*\*，进入日历应用界面，控制台将显示日历的操作说明

![图片](https://uploader.shimo.im/f/zooEtMyugZ8vrK4S.png!thumbnail)

\* 用户输入要查询的年月日

![图片](https://uploader.shimo.im/f/jL8iqJ0hxF96BoZh.png!thumbnail)

\* 系统输出查询的日期具体是星期几，并且输出该月份的日历

![图片](https://uploader.shimo.im/f/n65R9Dc6KqxhBLVf.png!thumbnail)

\*\*功能实现\*\*

\* 调用weekday()函数来判断输入日期是星期几

\* 调用display\_month来打印该月的日历

\* 调用calendar\_main()函数来接受用户指令进行查询

#### 2.2.6.7 calculator（计算器）

\*\*操作描述\*\*

\* 用户输入指令\*\*“calcul”\*\*，进入计算器应用界面，控制台将显示计算器的操作说明

# ![图片](https://uploader.shimo.im/f/Ca2S8gETxxa8nbNE.png!thumbnail)

\* 输入要进行的操作

![图片](https://uploader.shimo.im/f/Mo7JVO8pEXWYUsb6.png!thumbnail)

\* 输入要计算的表达式或数字

![图片](https://uploader.shimo.im/f/2zomRbHEStbdQxn7.png!thumbnail)

\*\*功能实现\*\*

\* 调用get\_option(int \*fd\_stdin)函数让用户选择要进行的操作

\* 用户输入要计算的数字

\* 调用print\_result()函数打印计算结果

#### 2.2.6.8 Snake（贪吃蛇）

\*\*操作描述\*\*

\* 用户输入指令\*\*“Snake”\*\*，进入贪吃蛇的游戏界面。首先是一个简介与欢迎界面，为用户提供了基本的玩法说明，本界面输入任意键进入游戏界面。

\*游戏界面可以通过指令的输入控制贪吃蛇的行走，例如“w 3”将控制贪吃蛇向上行走三格。

\*\*功能实现\*\*

\* 调用snake\_game(int fd\_stdin)函数进入贪吃蛇游戏中

\* 首先调用welcome()和rule()函数进入简介与欢迎界面，输入任意键可以退出。

\* 系统自动初始化地图和蛇以及蛇并打印出来开始接受用户的指令。用户可以通过输入相应的指令执行行走的操作。

\* 通过一个while()循环不断地接受指令直到游戏结束。

# 3 项目核心代码

## 3.1 进程管理

#define NULL ((void\*)0)

//进程管理主函数

void runProcessManage(int fd\_stdin)

{

clear();

char readbuffer[12];

while (1)

{

Process\_Welcome();

printf("SummerOS process-manager: $ ");

int end = read(fd\_stdin, readbuffer, 12);

readbuffer[end] = 0;

int i = 0, j = 0;

//获得命令指令

char com[8] = { 0 };

while (readbuffer[i] != ' ' && readbuffer[i] != 0)

{

com[i] = readbuffer[i];

i++;

}

i++;

//获取命令目标

char number[2] = { 0 };

while (readbuffer[i] != ' ' && readbuffer[i] != 0)

{

number[j] = readbuffer[i];

i++;

j++;

}

//结束进程;

if (strcmp(com, "block") == 0)

{

clear();

block\_a\_Process(number);

continue;

}

//重启进程

else if (strcmp(com, "start") == 0)

{

clear();

start\_a\_Process(number);

continue;

}

//弹出提示

else if (strcmp(readbuffer, "help") == 0)

{

clear();

Process\_Welcome();

}

//打印全部进程

else if (strcmp(readbuffer, "ps") == 0)

{

clear();

show\_all\_Process();

}

//退出进程管理

else if (strcmp(readbuffer, "quit") == 0)

{

clear();

break;

}

else if (!strcmp(readbuffer, "clear")) {

clear();

}

//错误命令提示

else

{

clear();

printf("\nSorry, there is no such command in this Process Management System.\n\n");

}

}

}

//打印欢迎界面

void Process\_Welcome()

{

printf(" ====================================================================\n");

printf(" # Welcome to \*\*\*\*\*\*\* \*\*\*\*\*\*\* #\n");

printf(" # SummerOS ~ Process Manager \* \* \* #\n");

printf(" # \* \* \*\*\*\*\*\*\* #\n");

printf(" # \* \* \* #\n");

printf(" # \*\*\*\*\*\*\* \*\*\*\*\*\*\* #\n");

printf(" # #\n");

printf(" # [COMMAND] [FUNCTION] #\n");

printf(" # #\n");

printf(" # $ clear | clear the cmd #\n");

printf(" # $ ps | show all process #\n");

printf(" # $ block id | block a process #\n");

printf(" # $ start id | start a process #\n");

printf(" # $ quit | quit process management system #\n");

printf(" # $ help | show command list of this system #\n");

printf(" # #\n");

printf(" # #\n");

printf(" # #\n");

printf(" # #\n");

printf(" ====================================================================\n");

printf("\n\n");

}

//打印所有进程

void show\_all\_Process()

{

int i;

printf("===============================================================================\n");

printf(" ProcessID \* ProcessName \* ProcessPriority \* Running? \n");

//进程号，进程名，优先级，是否在运行

printf("-------------------------------------------------------------------------------\n");

for (i = 0; i < 8; i++)//逐个遍历

{

printf(" %d", proc\_table[i].pid);

printf(" %5s", proc\_table[i].name);

printf(" %2d", proc\_table[i].priority);

if (proc\_table[i].priority == 0)

printf(" no\n");

else

printf(" yes\n");

}

printf("===============================================================================\n\n");

}

int get\_Mag(int n)

{

int mag = 1;

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

{

mag = mag \* 10;

}

return mag;

}

//计算进程pid

int get\_Pid(char str[])

{

int length = 0;

for (; length < MAX\_FILENAME\_LEN; length++)

{

if (str[length] == '\0')

{

break;

}

}

int pid = 0;

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

{

if (str[i] - '0' > -1 && str[i] - '9' < 1)

{

pid = pid + (str[i] + 1 - '1') \* get\_Mag(length - 1 - i);

}

else

{

pid = -1;

break;

}

}

return pid;

}

//结束进程

void block\_a\_Process(char str[])

{

int pid = get\_Pid(str);

//健壮性处理以及结束进程

if (pid >= 8 || pid < 0)

{

printf("The pid out of the range\n");

}

else if (pid < NR\_TASKS)

{

printf("System tasks cannot be blocked.\n");

}

else if (proc\_table[pid].priority == 0 || proc\_table[pid].p\_flags == -1)

{

printf("Process not found.\n");

}

else if (pid == 4 || pid == 6)

{

printf("This process cannot be blocked.\n");

}

else

{

proc\_table[pid].priority = 0;

proc\_table[pid].p\_flags = -1;

printf("Target process has been blocked.\n");

}

show\_all\_Process();

}

//启动进程

void start\_a\_Process(char str[])

{

int pid = get\_Pid(str);

if (pid >= 8 || pid < 0)

{

printf("The pid out of the range\n");

}

else if (proc\_table[pid].p\_flags == 1)

{

printf("This process is already running.\n");

}

else

{

proc\_table[pid].priority = 1;

proc\_table[pid].p\_flags = 1;

printf("Target process is running.\n");

}

show\_all\_Process();

}

## 3.2 文件管理

### 3.2.1 文件系统系统功能

#### 3.2.1.1 多级文件目录的实现

\*\*strip\_path()\*\*

```c++

PUBLIC int strip\_path(char \* filename, const char \* pathname,

              struct inode\*\* ppinode)

{

    const char \* s = pathname;

    char \* t = filename;

    struct inode \* dir\_inode = get\_inode(ROOT\_DEV, ROOT\_INODE);

    struct inode \* next\_inode;

    if (s == 0)

        return -1;

    if (\*s == '/' || \*s == '$' || \*s == '#')

        s++;

    if (\*s == '/')

        s++;

    while (\*s) /\* check each character \*/

    {

        if (\*s == '/')

        {

            \*t = 0;

            int dev = dir\_inode->i\_dev;

            int dir\_inode\_nr = get\_file(filename, dir\_inode, I\_DIRECTORY);

            next\_inode = get\_inode(dev, dir\_inode\_nr);

            put\_inode(dir\_inode);

            dir\_inode = next\_inode;

            t = filename;

            s++;

        }

        \*t++ = \*s++;

        /\* if filename is too long, just truncate it \*/

        if (t - filename >= MAX\_FILENAME\_LEN)

            break;

    }

    \*t = 0;

    \*ppinode = dir\_inode;

    return 0;

}

```

\*\*get\_file()\*\*

```c++

PUBLIC int get\_file(char\* filename, struct inode\* dir\_inode,

u32 mode)

{

    if (filename[0] == 0)   /\* path: "/" \*/

        return dir\_inode->i\_num;

     \* Search the dir for the file.

     \*/

    int dir\_blk0\_nr = dir\_inode->i\_start\_sect;

    int nr\_dir\_blks = (dir\_inode->i\_size + SECTOR\_SIZE - 1) / SECTOR\_SIZE;

    int nr\_dir\_entries =

      dir\_inode->i\_size / DIR\_ENTRY\_SIZE; /\*\*

                           \* including unused slots

                           \* (the file has been deleted

                           \* but the slot is still there)

                           \*/

    int i, j, m = 0;

    struct dir\_entry \* pde;

    int dev = dir\_inode->i\_dev;

    for (i = 0; i < nr\_dir\_blks; i++)

    {

        RD\_SECT(dev, dir\_blk0\_nr + i);

        pde = (struct dir\_entry \*)fsbuf;

        for (j = 0; j < SECTOR\_SIZE / DIR\_ENTRY\_SIZE; j++, pde++)

        {

            if (strcmp(filename, pde->name) == 0)

            {

                int result = pde->inode\_nr;

                struct inode \* pin = get\_inode(dev, pde->inode\_nr);

                if (pin->i\_mode == mode || mode == 0)

                {

                    put\_inode(pin);

                    return result;

                }

                put\_inode(pin);

                RD\_SECT(dev, dir\_blk0\_nr + i);

            }

            if (++m > nr\_dir\_entries)

                break;

        }

        if (m > nr\_dir\_entries) /\* all entries have been iterated \*/

            break;

    }

    /\* file not found \*/

    return 0;

}

```

#### \*\*3.2.1.2 文件的查找与创建\*\*

\*\*search\_file()\*\*

```c++

PUBLIC int search\_file(char \* path)

{

    char \* \_path = path;

    if (path[0] == '$' || path[0] == '#') \_path++;

    char filename[MAX\_PATH];

    memset(filename, 0, MAX\_FILENAME\_LEN);

    struct inode \* dir\_inode;

    if (strip\_path(filename, \_path, &dir\_inode) != 0)

        return 0;

    if (path[0] == '$')

    {

        return get\_file(filename, dir\_inode, I\_DIRECTORY);

    }

    if (path[0] == '#')

    {

        return get\_file(filename, dir\_inode, I\_REGULAR);

    }

    return get\_file(filename, dir\_inode, 0);

}

```

\*\*create\_file()\*\*

```c++

PRIVATE struct inode \* create\_file(char \* path, int flags)

{

    u32 mode = I\_REGULAR;

    if (path[0] == '$') mode = I\_DIRECTORY;

    char filename[MAX\_PATH];

    struct inode \* dir\_inode;

    if (strip\_path(filename, path, &dir\_inode) != 0)

        return 0;

    int inode\_nr = alloc\_imap\_bit(dir\_inode->i\_dev);

    int free\_sect\_nr = alloc\_smap\_bit(dir\_inode->i\_dev,

                      NR\_DEFAULT\_FILE\_SECTS);

    struct inode \*newino = new\_inode(dir\_inode->i\_dev, inode\_nr,

                     free\_sect\_nr, mode);

    new\_dir\_entry(dir\_inode, newino->i\_num, filename);

    return newino;

}

```

### 3.2.2 文件系统用户功能

#### 3.2.2.1 写文件

```c++

void wt(char \* cur\_dir, char \* filename)

{

char path[32] = "#";

int i = 0;

for (; ; i++)

{

if (cur\_dir[i] == 0) break;

path[i + 1] = cur\_dir[i];

}

i++;

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

{

if (filename[j] == 0) break;

path[i++] = filename[j];

}

path[i] = 0;

int fd = open (path, O\_RDWR);

if (fd == -1)

{

printf("Failed to open %s!\n", filename);

return;

}

char tty\_name[] = "/dev\_tty0";

int fd\_stdin  = open(tty\_name, O\_RDWR);

if (fd\_stdin == -1)

{

printf("Failed to write file!\n");

return;

}

char writeBuf[4096];  // 写缓冲区

int final = read(fd\_stdin, writeBuf, 4096);

writeBuf[final] = 0;

write(fd, writeBuf, final + 1);

printf("Totally write %d bytes in the file : %s\n",final,writeBuf);

close(fd);

}

```

#### 3.2.2.2读文件

```c++

void rd(char\*cur\_dir, char \* filename){

char path[32] = "#";

int i = 0;

for (; ; i++)

{

if (cur\_dir[i] == 0) break;

path[i + 1] = cur\_dir[i];

}

i++;

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

{

if (filename[j] == 0) break;

path[i++] = filename[j];

}

path[i] = 0;

int fd = open (path, O\_RDWR);

if(fd==-1){

printf("Failed to open %s!\n", filename);

        return;

}

char buf[4096];

    int rdFlag = read(fd, buf, 4096);

    if (rdFlag == -1)  // 读取文件内容失败

    {

        printf("Failed to read file!\n");

        close(fd);

        return;

    }

    printf("Content of file %s :\n", filename);

    printf(" %s\n",buf);

    close(fd);

}

```

#### 3.2.2.3创建文件

```c++

void touch(char \* cur\_dir, char \* filename)

{

    if (filename[0] <= 0)

    {

        printf("touch:: error: command touch need one parameter.\n");

        return;

    }

    if (filename[0] == ' ')

    {

        printf("touch:: error: file name cannot start with space.\n");

        return;

    }

    char path[32] = "#";   //普通目录标识符

    int i = 0;

    for (; ; i++)

    {

        if (cur\_dir[i] == 0) break;

        path[i + 1] = cur\_dir[i];

    }

    i++;

    for (int j = 0; ; j++)

    {

        if (filename[j] == 0) break;

        path[i++] = filename[j];

    }

    path[i] = 0;

    int fd = open(path, O\_CREAT);

    if (fd != -1)

    {

        close(fd);

    }

    else

    {

        printf("touch error: file exists.\n");

    }

}

```

#### 3.2.2.4展示文件列表

```c++

PUBLIC int do\_ls()

{

    char pathname[MAX\_PATH];

    /\* get parameters from the message \*/

    int name\_len = fs\_msg.NAME\_LEN; /\* length of filename \*/

    int src = fs\_msg.source;    /\* caller proc nr. \*/

    assert(name\_len < MAX\_PATH);

    phys\_copy((void\*)va2la(TASK\_FS, pathname),

          (void\*)va2la(src, fs\_msg.PATHNAME),

          name\_len);

    pathname[name\_len] = 0;

    char \* filename;

    struct inode \* dir\_inode;

    strip\_path(filename, pathname, &dir\_inode);

    return get\_list(dir\_inode, I\_DIRECTORY) + get\_list(dir\_inode, I\_REGULAR);

}

```

#### 3.2.2.5切换文件目录

```c++

void cd(char \* cur\_dir, char \* dir\_name)

{

    //处理特殊目录

    if (dir\_name[0] <= 0)

    {

        printf("cd:: error: command cd need one parameter.\n");

        return;

    }

    if (dir\_name[0] == ' ')

    {

        printf("cd:: error: directory cannot start with space.\n");

        return;

    }

    if (dir\_name[0] == '.')

    {

        if (dir\_name[1] == '.')

        {

            int i = 0;

            for (; ; i++)

            {

                if (cur\_dir[i] == '/' && cur\_dir[i + 1] == 0)

                {

                    break;

                }

            }

            if (i > 0)

            {

                cur\_dir[i--] = 0;

                for (; ; i--)

                {

                    if (cur\_dir[i] == '/') break;

                    cur\_dir[i] = 0;

                }

            }

        }

        return;

    }

    char path[32] = "$";

    int i = 0;

    for (; ; i++)

    {

        if (cur\_dir[i] == 0) break;

        path[i + 1] = cur\_dir[i];

    }

    i++;

    for (int j = 0; ; j++)

    {

        if (dir\_name[j] == 0) break;

        path[i++] = dir\_name[j];

    }

    path[i] = 0;

    int fd = open(path, O\_RDWR);

    if (fd == -1)

    {

        printf("No such directory.\n");

        return;

    }

    close(fd);

    strcat(cur\_dir, dir\_name);

    strcat(cur\_dir, "/");

}

```

#### 3.2.2.6 删除文件

```c++

void rm(char \* cur\_dir, char \* filename, int flag)

{

    char path[32] = "#";

    if (flag == 1)

    {

        path[0] = '$';

    }

    int i = 0;

    for (; ; i++)

    {

        if (cur\_dir[i] == 0) break;

        path[i + 1] = cur\_dir[i];

    }

    i++;

    for (int j = 0; ; j++)

    {

        if (filename[j] == 0) break;

        path[i++] = filename[j];

    }

    path[i] = 0;

    int fd = unlink(path);

    if (fd == 0)

    {

        printf("%s deleted!\n", filename);

    }

    else

    {

        printf("Failed to delete %s!\n", filename);

    }

}

```

## 3.3 游戏及工具

### 3.3.1 游戏

#### 3.3.1.1 sudoku（数独）

\*\*get\_map()\*\*

```c++

/\* 搜索算法建立数独棋盘 \*/

PUBLIC void dfs(int i, int j, bool \*success)

{

    if (\*success)

    {

        return;

    }

    if (i == 9 && j == 0)

    {

        /\* 成功建立棋盘 \*/

        \*success = true;

        copy\_map();

        return;

    }

    int next\_i = j == 8 ? i + 1 : i;

    int next\_j = j == 8 ? 0 : j + 1;

    /\*\*

     \*  建立1-9的随机数组进行遍历

     \*

     \*  but bochs中导入time.h有点问题

     \*/

    for (int k = 9; k > 0; k--)

    {

        map\_copy[i][j] = k;

        if (is\_legal(i, j))

        {

            dfs(next\_i, next\_j, success);

        }

        map\_copy[i][j] = 0;

    }

}

/\* 获取数独棋盘 \*/

PUBLIC void get\_map()

{

    bool success = false;

    dfs(0, 0, &success);

}

```

\*\*sudoku\_main()\*\*

```c++

PUBLIC void sudoku\_main(int \*fd\_stdin)

{

    /\* init the game \*/

    get\_map();

    char order[2];         // 指令

    char r[2], c[2], v[2]; // row,column,value

    /\* press ENTER to start the game \*/

    sudoku\_list();

    int start = read(\*fd\_stdin, order, 512);

    print\_map();

    while (1)

    {

        printf("[E] to enter a number [D] to delete a number [Q] to quit the game [P] to print the map\n");

        printf("Please enter the order: ");

        /\* read \*/

        int p = read(\*fd\_stdin, order, 512);

        order[p] = 0;

        if (strcmp(order, "E") == 0)

        {

            /\* 输入数字 \*/

            int row, col, val;

            printf("Please enter the row of the grid you want to complete: ");

            p = read(\*fd\_stdin, r, 512);

            r[p] = 0;

            row = r[0] - '0';

            printf("Please enter the column of the grid you want to complete: ");

            p = read(\*fd\_stdin, c, 512);

            c[p] = 0;

            col = c[0] - '0';

            if (status[row - 1][col - 1] != EMPTY)

            {

                printf("Sorry, grid[%d, %d] has been filled in.\n", row, col);

                print\_map();

                continue;

            }

            printf("Please enter the number you want to fill in: ");

            p = read(\*fd\_stdin, v, 512);

            v[p] = 0;

            val = v[0] - '0';

            if (val == sudoku\_map[row - 1][col - 1])

            {

                /\* 输入正确 \*/

                status[row - 1][col - 1] = CORRECT;

            }

            else

            {

                /\* 输入错误 \*/

                status[row - 1][col - 1] = WRONG;

            }

            player[row - 1][col - 1] = val;

        }

        else if (strcmp(order, "D") == 0)

        {

            /\* 删除数字 \*/

            int row, col;

            printf("Please enter the row of the grid you want to delete: ");

            p = read(\*fd\_stdin, r, 512);

            r[p] = 0;

            row = r[0] - '0';

            printf("Please enter the column of the grid you want to delete: ");

            p = read(\*fd\_stdin, c, 512);

            c[p] = 0;

            col = c[0] - '0';

            if (status[row - 1][col - 1] == GIVEN)

            {

                /\* 初始给定数字不能删除 \*/

                printf("Sorry, Grid[%d, %d] cannot be deleted.\n\n", row, col);

                continue;

            }

            else

            {

                status[row - 1][col - 1] = EMPTY;

                player[row - 1][col - 1] = 0;

            }

        }

        else if (strcmp(order, "Q") == 0)

        {

            printf("\n\n\n");

            break;

        }

        else if (strcmp(order, "P") != 0)

        {

            printf("False Command!\n\n");

            continue;

        }

        printf("\n\n\n\n\n\n\n\n\n");

        print\_map();

        /\* 判定是否获胜 \*/

        if (check\_win())

        {

            printf("            Congratulation!! You successfully complete the sudoku!!\n");

            break;

        }

    }

}

```

#### 3.3.1.2 bwchess（黑白棋）

\*\*Hint()\*\*

```c++

int Hint(char board[][MAXSIZE], int arrput[][MAXSIZE], char level) //最佳走法

{

int row, col, i, j;

char board1[MAXSIZE][MAXSIZE] = { 0 };

int maxscore = 0;

int score = 0;

char foe;

if (level == 1)

foe = -1;

else

foe = 1;

for (row = 0; row < MAXSIZE; row++)

for (col = 0; col < MAXSIZE; col++)

{

if (!arrput[row][col])

continue;

for (i = 0; i < MAXSIZE; i++)

for (j = 0; j < MAXSIZE; j++)

{

board1[i][j] = board[i][j];

}

Print(board1, row, col, level);

score = CalSore(board1, level);

if (maxscore < score)

maxscore = score;

}

return maxscore;

}

```

CalSore()

```c++

int CalSore(char board[][MAXSIZE], char level) //计算成绩

{

int score = 0;

int row, col;

char foe;

if (level == 1)

foe = -1;

else

foe = 1;

char player = -1 \* foe;

for (row = 0; row < MAXSIZE; row++)

for (col = 0; col < MAXSIZE; col++)

{

score = score - (board[row][col] == foe);

score = score + (board[row][col] == player);

}

return score;

}

```

\*\*bwchess\_main()\*\*

```c++

 for (i = count; count < (MAXSIZE \* MAXSIZE) && cross < 2;)

        {

            if (level == 1)

            {

                level = 0;

                if (Exa(board, arrput, 2))

                {

                    while (1)

                    {

                        printf("\nenter your action(r):");

                        p = read(\*fd\_stdin, r, 512);

                        r[p] = 0;

                        if (!strcmp(r, "Q"))

                        {

                            return;

                        }

                        x = r[0] - '0';

                        printf("\nenter your action(c):");

                        p = read(\*fd\_stdin, c, 512);

                        c[p] = 0;

                        y = c[0];

                        if (!strcmp(c, "Q"))

                        {

                            return;

                        }

                        //y = input[1];

                        x--;

                        if (y >= 'a')

                        {

                            y = y - 'a' + 1;

                        }

                        else

                        {

                            y = y - 'A' + 1;

                        }

                        y--;

                        if (x >= 0 && y >= 0 && x < MAXSIZE && y < MAXSIZE && arrput[x][y])

                        {

                            Print(board, x, y, 2);

                            count++;

                            break;

                        }

                        else

                        {

                            printf("\nwrong order try again .\n");

                        }

                    }

                    Show(board); //更新棋盘

                }

                else if (++cross < 2)

                {

                    printf("\n is my turn.\n");

                }

                else

                {

                    printf("\ngame over.\n");

                }

            }

            else

            {

                level = 1;

                if (Exa(board, arrput, 1))

                {

                    cross = 0;

                    Foeplay(board, arrput, 1);

                    count++;

                    Show(board);

                }

                else

                {

                    if (++cross < 2)

                    {

                        printf("\ni cannot move,you turn\n");

                    }

                    else

                    {

                        printf("\ngame over.");

                    }

                }

            }

        }

        Show(board);

        score[0] = score[1] = 0;

        for (row = 0; row < MAXSIZE; row++) //计算分数

        {

            for (col = 0; col < MAXSIZE; col++)

            {

                score[0] = score[0] + (board[row][col] == -1);

                score[1] = score[1] + (board[row][col] == 1);

            }

        }

        printf("final score:\n");

        printf("white:%d\nblack:%d\n", score[1], score[0]);

        if (score[1] > score[0])

        {

            printf("\nha ha ha ,you lose \n\n");

        }

        else

        {

            printf("\noh,i will be back!\n\n");

        }

    }

    //scanf\_s("%c", &ok);

}

```

#### 3.3.1.3 tic-tac-toe（井字棋）

\*\*ticTacToe\_main()\*\*

```c++

PUBLIC int ticTacToe\_main(int \*fd\_stdin)

{

    int player = 0;

    int winner = 0;

    int number = 0;

    int row = 0;

    int column = 0;

    char tic\_tac\_toe[3][3] = {

        {'1', '2', '3'},

        {'4', '5', '6'},

        {'7', '8', '9'}};

    //让双方玩家轮流输入自己想要标志的位置

    int i;

    char o[2];

    ticTacToe\_list();

    read(\*fd\_stdin, o, 512);

    clear();

    for (i = 0; i < 9 && winner == 0; i++)

    {

        printf("\n");

        printf("                       ");

        printf(" %c | %c | %c \n", tic\_tac\_toe[0][0], tic\_tac\_toe[0][1], tic\_tac\_toe[0][2]);

        printf("                       ");

        printf("---+---+---\n");

        printf("                       ");

        printf(" %c | %c | %c \n", tic\_tac\_toe[1][0], tic\_tac\_toe[1][1], tic\_tac\_toe[1][2]);

        printf("                       ");

        printf("---+---+---\n");

        printf("                       ");

        printf(" %c | %c | %c \n", tic\_tac\_toe[2][0], tic\_tac\_toe[2][1], tic\_tac\_toe[2][2]);

        player = i % 2 + 1;

        do

        {

            printf("\nif you want to exit, please input 0.\n");

            printf("\nplayer %c", (player == 1) ? 'A' : 'B');

            printf(" input a number of the grid to put your %c piece:", (player == 1) ? 'X' : 'O');

            //scanf\_s("%d", &number);

            int p = read(\*fd\_stdin, o, 512);

            o[p] = 0;

            number = o[0] - '0';

            if(number==0) return 0;

            row = (number - 1) / 3;    //行的索引码

            column = (number - 1) % 3; //列的索引码

        } while (number < 0 || number > 9 || tic\_tac\_toe[row][column] > '9');

        tic\_tac\_toe[row][column] = (player == 1) ? 'X' : 'O';

        //检查此玩家是否获胜

        //检查对角线上该玩家是否获胜

        if ((tic\_tac\_toe[0][0] == tic\_tac\_toe[1][1] && tic\_tac\_toe[0][0] == tic\_tac\_toe[2][2]) ||

            (tic\_tac\_toe[0][2] == tic\_tac\_toe[1][1] && tic\_tac\_toe[0][2] == tic\_tac\_toe[2][0]))

        {

            winner = player;

        }

        //检查横或竖上该玩家是否获胜

        else

        {

            for (int line = 0; line < 3; line++)

            {

                if ((tic\_tac\_toe[line][0] == tic\_tac\_toe[line][1] && tic\_tac\_toe[line][0] == tic\_tac\_toe[line][2]) ||

                    (tic\_tac\_toe[0][line] == tic\_tac\_toe[1][line] && tic\_tac\_toe[0][line] == tic\_tac\_toe[2][line]))

                {

                    winner = player;

                }

            }

        }

        clear();

    }

    //公布最后得分面板

    printf("\n");

    printf("                       ");

    printf(" %c | %c | %c \n", tic\_tac\_toe[0][0], tic\_tac\_toe[0][1], tic\_tac\_toe[0][2]);

    printf("                       ");

    printf("---+---+---\n");

    printf("                       ");

    printf(" %c | %c | %c \n", tic\_tac\_toe[1][0], tic\_tac\_toe[1][1], tic\_tac\_toe[1][2]);

    printf("                       ");

    printf("---+---+---\n");

    printf("                       ");

    printf(" %c | %c | %c \n", tic\_tac\_toe[2][0], tic\_tac\_toe[2][1], tic\_tac\_toe[2][2]);

    //打印最后胜利者结果

    if (winner == 0)

    {

        printf("\nWhat a pity! Nobody win the game!\n");

    }

    else

    {

        printf("\nCongratulation! The winner is %c!\n", (winner == 1) ? 'A' : 'B');

    }

    printf("\n\nPress ENTER to continue...\n");

    read(\*fd\_stdin, o, 512);

    return 0;

}

```

#### 3.3.1.4mine（扫雷）

void InitBoard()

```c++

void InitBoard(char board[ROWS][COLS], int rows, int cols, char set)//为数组初始化，设计展示的界面

{

int i = 0;

int j = 0；

for (i = 0; i < rows; i++)

{

for (j = 0; j < cols; j++)

{

board[i][j] = set;//set可以更改

}

}

}

```

void FindMine()

```c++

void FindMine(char mine[ROWS][COLS], char show[ROWS][COLS], int row, int col, int fd\_stdin)

{

int x = 0;

int y = 0;

int win = 0;

int p;

char xn[2];

char yn[2];

    printf("\nCurrent score is %d.\n", win);

    printf("You'll win when score reaches 71!\n\n");

while (win < (row \* col - EASYCOUNT))

{

printf("please input x:>");

p = read(fd\_stdin, xn, 512);

printf("please input y:>");

x=xn[0]-'0';

p = read(fd\_stdin, yn, 512);

y=yn[0]-'0';

if (x >= 1 && x <= row && y >= 1 && y <= col)

{

if (mine[x][y] == '1')

{

clear();

printf("Boom!!!  Game over!\n");

DispalyBoard(mine, row, col);

break;

}

else

{

                ShowMineCount(mine, show, row, col, x, y, &win);

clear();

DispalyBoard(show, row, col);

                printf("\nCurrent score is %d.\n", win);

                printf("You'll win when score reaches 71!\n\n");

}

if (win == row \* col - EASYCOUNT)

{

                clear();

                DispalyBoard(mine, row, col);

printf("\nYou win! Congratulation!!\n\n");

}

}

else

{

printf("wrong index:>\n");

}

}

}

```

#### 3.3.1.5 carrycraft（推箱子）

\*\*drawmain()\*\*

```c++

int drawmain()

{

clear();

int i, j;

win(); //判断输赢

for (i = 0; i < 9; i++)

{

printf("                   ");

for (j = 0; j < 11; j++)

{

switch (map[i][j])

{

case 0:

printf("  "); //空白的位置

break;

case 1:

printf("W "); //墙

break;

case 2:

printf("P "); //代表人

break;

case 3:

printf("C "); //代表箱子

break;

case 4:

printf("T "); //代表目标地址

break;

case 6:

printf("P "); //代表人和目标地址重合

break;

case 7:

printf("G "); //代表箱子和目标地址重合

break;

}

}

printf("\n");

}

return 0;

}

```

\*\*carry\_main()\*\*

```c++

PUBLIC int carrycraft\_main(int\* fd\_stdin)

{

    char o[2];

    craft\_list();

    read(\*fd\_stdin, o, 512);

    while (1)

    {

        drawmain();

        int p = read(\*fd\_stdin, o, 512);

        o[p] = 0;

        char push = o[0];

        int count, caw;

        for (int i = 0; i < 9; i++)

        {

            for (int j = 0; j < 11; j++)

            {

                if (map[i][j] == 2 || map[i][j] == 6)

                {

                    count = i;

                    caw = j;

                }

            }

        }

        //printf("%c,tui");

        if (push == 'w')

        {

            if (map[count - 1][caw] == 0 || map[count - 1][caw] == 4)

            {

                map[count][caw] -= 2;

                map[count - 1][caw] += 2;

            }

            else if (map[count - 1][caw] == 3 || map[count - 1][caw] == 7)

            {

                if (map[count - 2][caw] == 0 || map[count - 2][caw] == 4)

                {

                    map[count][caw] -= 2;

                    map[count - 1][caw] -= 1;

                    map[count - 2][caw] += 3;

                }

            }

            //break;

        }

        if (push == 's')

        {

            if (map[count + 1][caw] == 0 || map[count + 1][caw] == 4)

            {

                map[count][caw] -= 2;

                map[count + 1][caw] += 2;

            }

            else if (map[count + 2][caw] == 0 || map[count + 2][caw] == 4)

            {

                if (map[count + 1][caw] == 3 || map[count + 1][caw] == 7)

                {

                    map[count][caw] -= 2;

                    map[count + 1][caw] -= 1;

                    map[count + 2][caw] += 3;

                }

            }

            //break;

        }

        if (push == 'a')

        {

            if (map[count][caw - 1] == 0 || map[count][caw - 1] == 4)

            {

                map[count][caw] -= 2;

                map[count][caw - 1] += 2;

            }

            else if (map[count][caw - 2] == 0 || map[count][caw - 2] == 4)

            {

                if (map[count][caw - 1] == 3 || map[count][caw - 1] == 7)

                {

                    map[count][caw] -= 2;

                    map[count][caw - 1] -= 1;

                    map[count][caw - 2] += 3;

                }

            }

            //break;

        }

        if (push == 'd')

        {

            if (map[count][caw + 1] == 0 || map[count][caw + 1] == 4)

            {

                map[count][caw] -= 2;

                map[count][caw + 1] += 2;

            }

            else if (map[count][caw + 2] == 0 || map[count][caw + 2] == 4)

            {

                if (map[count][caw + 1] == 3 || map[count][caw + 1] == 7)

                {

                    map[count][caw] -= 2;

                    map[count][caw + 1] -= 1;

                    map[count][caw + 2] += 3;

                }

            }

            //break;

        }

    }

    return 0;

}

```

#### 3.3.1.6 snake（贪吃蛇）

#define ROW 10

#define COL 24

#define Up 0

#define Down 1

#define Left 2

#define Right 3

#define LENGTH 2

int score = 0;

bool eated = false;

int M[ROW][COL];

int direction=0;

int length=LENGTH;//4是因为move函数消去尾巴。这样蛇的长度打印出来为3

int Coordinate[ROW \* COL][2]={0,0};

void welcome() {

printf("\n\n\n\n\n\n");

printf(" \* \* \* \* \* \* \* \* \* \* \* \* \* \*\n");

printf(" \n");

printf(" \* WELCOME! \*\n");

printf(" \n");

printf(" \* \* \* \* \* \* \* \* \* \* \* \* \* \*\n");

}

void rule() {

printf("Rule:\n");

printf(" Control the direction of the snake, look for something to eat,\n and be careful not to hit the wall and bite yourself!\n");

printf("\n\n\nEating food will get corresponding points.\n");

printf(" Your only goal is to become a long snake!\n");

}

void Initmap() {

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

{

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

{

if (i == 0 || i == ROW - 1 || j == 0 || j == COL - 1)

{

M[i][j] = 1;//1为墙

}

else M[i][j] = 0;//0不为0

}

}

}

void Drawmap() {

clear();

printf("\n\n\n");

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

{

printf(" ");

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

{

if (M[i][j] == 1)//wall

printf("\*");

else if(M[i][j] == -1)//food

printf("&");

else if(M[i][j] == 2)//snake body

printf("#");

else if(M[i][j] == 3)//snake head

printf("@");

else printf(" ");

}

printf("\n");

}

printf(" score:%d\n",score);

}

void fuclength(){

length++;

}

void Initsnake() {

Coordinate[0][0] = ROW / 2;//0为横坐标

Coordinate[0][1] = COL / 2;//1为纵坐标

for (int i = 1; i < length; i++)

{

Coordinate[i][0] = Coordinate[0][0] + i;

Coordinate[i][1] = Coordinate[0][1];

}

}

void Drawsnake() {

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

{

if(i==0)

M[Coordinate[i][0]][Coordinate[i][1]]=3;

else

M[Coordinate[i][0]][Coordinate[i][1]]=2;

}

}

void Move() {//从尾部开始后面的等于前面的坐标

M[Coordinate[length-1][0]][Coordinate[length-1][1]]=0;

for (int i = length - 1; i > 0; i--)

{

Coordinate[i][0] = Coordinate[i - 1][0];//行坐标

Coordinate[i][1] = Coordinate[i - 1][1];//列坐标

}

switch (direction)

{

case Up:

Coordinate[0][0]--;

break;

case Down:

Coordinate[0][0]++;

break;

case Left:

Coordinate[0][1]--;

break;

case Right:

Coordinate[0][1]++;

break;

}

}

void directioncontrol(int fd\_stdin,char choice) {

switch (choice)

{

case 'W':

case 'w':

if (direction != Down)

direction = Up;

break;

case 'S':

case 's':

if (direction != Up)

direction = Down;

break;

case 'A':

case 'a':

if (direction != Right)

direction = Left;

break;

case 'D':

case 'd':

if (direction != Left)

direction = Right;

break;

}

}

bool death(){

bool isGameOver = false;

int sX = Coordinate[0][0], sY = Coordinate[0][1]; //蛇头的x坐标和y坐标

if (sX == 0 || sX == ROW - 1 || sY == 0 || sY == COL - 1)

isGameOver = true;

for (int i = 1; i < length; i++) //判断有没有吃到自己

{

if (Coordinate[i][0] == sX && Coordinate[i][1] == sY)

isGameOver = true;

}

return isGameOver;

}

void Drawfood(){

int error=0;

while(!error)

{

int i = rand() % (ROW-2)+1; //生成0~ROW-1之间的一个数

int j = rand() % (COL-2)+1;

error=1;

for (int k = 0; k < length; k++)

{

if (i == Coordinate[k][0] && j == Coordinate[k][1])

{

error=0;

break;

}

}

M[i][j]=-1;

}

}

void snake\_game(int fd\_stdin){

clear();

welcome();

rule();

printf("\n push any key and continue\n");

char in[1];

read(fd\_stdin, in, 1);

clear();

//init

Initmap();

Initsnake();

Drawsnake();

Drawfood();

Drawmap();

int lived=1;

while(lived)

{

char in[32];

printf(" W: Up\n S: Down\n A: Left\n D: Rigth \nWrite the direction what you want to walk: \n");

printf("command example: w 3(you can up three times(please be less than 10))\nif you want to follow last,please enter\n");

printf("command:");

int p=read(fd\_stdin, in, 32);

in[p]=0;

char comm;

comm=in[0];

int i=1;

char times='1';

while((in[i]<'0'||in[i]>'9')&&in[i]!=0)

{

i++;

}

if(in[i]!=0)

times=in[i];

int time=0;

time=time+(times+1-'1');

int t=0;

while(t<time)

{

directioncontrol(fd\_stdin,comm);

Move();

if(death())

{

clear();

printf("\n Game Over \n push any key and continue\n");

char in[1];

read(fd\_stdin, in, 1);

lived=0;

break;

}

if (M[Coordinate[0][0]][Coordinate[0][1]] == -1)

{

eated = true;

if (eated == true) //如果吃到了食物

{

fuclength();

eated = false; //设置为false，不然无限变长

}

score += 10;

Drawfood();

// M[Coordinate[0][0]][Coordinate[0][1]] = 0;

Drawsnake();

}

t++;

Drawsnake();

Drawmap();

}

}

}

### 3.3.2 工具

#### 3.3.2.1calendar（日历）

\*\*weekay()\*\*

```c++

PUBLIC int weekday(int year, int month, int day)//是礼拜几

{

    int count;

    count = (year - 1) + (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400 + total\_day(year, month, day);

    count %= 7;

    return count;

}

```

\*\*display\_month()\*\*

```c++

PUBLIC void display\_month(int year, int month, int day)

{

//打印该月份日历

int i = 0, j = 1;

int week, max;

week = weekday(year, month, 1);

max = max\_day(year, month);

printf("\n                 %d - %s", year, month\_dis[month - 1]);

printf("\n");

printf("\n     S     M     T     W     T     F     S\n\n");

for (i = 0; i < week; i++)

printf("      ");

for (j = 1; j <= max; j++)

{

printf("%6d", j);

if (i % 7 == 6)

printf("\n\n");

i++;

}

printf("\n\n");

}

```

\*\*calendar\_main()\*\*

```c++

PUBLIC int calendar\_main(int \*fd\_stdin)

{

    int year, month, day;

    char y[5], m[3], d[3]; // year, month, day

    calendar\_intro();

    printf("Please enter year:");

    int p = read(\*fd\_stdin, y, 512);

    y[p] = 0;

    year = 0;

    int i = 0;

    while (i < p)

    {

        year \*= 10;

        year += (y[i++] - '0');

    }

    printf("Please enter month:");

    p = read(\*fd\_stdin, m, 512);

    m[p] = 0;

    month = 0;

    i = 0;

    while (i < p)

    {

        month \*= 10;

        month += (m[i++] - '0');

    }

    printf("Please enter day:");

    p = read(\*fd\_stdin, d, 512);

    d[p] = 0;

    day = 0;

    i = 0;

    while (i < p)

    {

        day \*= 10;

        day += (d[i++] - '0');

    }

    if (month < 1 || month > 12 || day < 1 || day > 31)

    {

        printf("error!");

        return -1;

    }

    display\_week(year, month, day);

    display\_month(year, month, day);

    printf("\nPress ENTER to continue...\n");

    read(\*fd\_stdin, d, 512);

    return 0;

}

```

#### 3.3.2.2calculator（计算器）

```c++

PUBLIC int calculator\_main(int\* fd\_stdin)

{

    int option, num1, num2, result;

    char a[20], b[20]; // num1,um2

    calculator\_list();

    read(\*fd\_stdin, a, 512);

    clear();

    do

    {

        option = get\_option(fd\_stdin);

        if (option == 0)

            break;

        do

        {

            printf("\nplease input a number:");

            int p = read(\*fd\_stdin, a, 512);

            a[p] = 0;

            num1 = 0;

            int i = 0;

            while (i < p)

            {

                num1 \*= 10;

                num1 += (a[i++] - '0');

            }

            printf("\nplease input another number:");

            p = read(\*fd\_stdin, b, 512);

            b[p] = 0;

            num2 = 0;

            i = 0;

            while (i < p)

            {

                num2 \*= 10;

                num2 += (b[i++] - '0');

            }

            if (option == 4 && num2 == 0)

            {

                printf("\nsorry!!divid can not be 0");

            }

            else

            {

                switch (option)

                {

                case 1:

                    result = num1 + num2;

                    break;

                case 2:

                    result = num1 - num2;

                    break;

                case 3:

                    result = num1 \* num2;

                    break;

                case 4:

                    result = num1 / num2;

                }

                print\_result(num1, num2, result, option);

            }

        } while (option == 4 && num2 == 0);

    } while (option != 0);

    return 0;

}

```

calculate()

```c++

int calculate(struct C rlt[], int size)

{

num\_clear();

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

if (rlt[i].tag == 1) {//如果是数字就压入栈中

num\_stack\_push(rlt[i].data);

}

else {// 如果是操作符，就从栈中弹出两个数字做运算

int right = num\_stack\_pop();

int left = num\_stack\_pop();

num\_stack\_push(operate(left, rlt[i].data, right));// 再将计算结果压入栈中

}

}

return num\_stack\_pop();

}

```

# 4 成员分工

|学号|姓名|分工|

|:----|:----|:----|

|2051488|韩可欣|

|2053380|彭浩|

|2053512|孙博闻|进程管理以及贪吃蛇|