

Phoenix

项目设计与功能说明文档

小组成员：1753402 谢康
1754188 谢尚汝
1754197 刘岚心
1751251 周贤

指导教师：王冬青

时间：2019-08-31

目录

1.项目简介	2
1.1 项目要求	2
1.2 项目目的	2
1.3 开发环境	2
1.4 参考资料	2
2.项目设计	2
2.1 总体设计	2
2.2 功能设计	2
2.2.1 开机动画	2
2.2.2 帮助页面	4
2.2.3 进程管理功能	5
2.2.4 清屏	6
2.2.5 文件管理	7
2.2.6 游戏功能	12
3.核心代码	14
3.1 用户及应用	14
3.1.1 扫雷	14
3.1.2 飞机大战	20
3.1.3 井字棋游戏	24
3.1.4 2048	26
3.2 系统级应用	28
3.2.1 进程管理	28
3.2.2 文件管理	29
3.3 其它功能	31

3.3.1 清屏.....	31
4.成员分工	31

1.项目简介

1.1 项目要求

在经过一学期的理论学习后结合实操完成《Orange 'S:一个操作系统的实现》项目要求，实现 B 级项目难度。

1.2 项目目的

将理论课的内容结合实践，从 0 真正实现一个操作系统，不仅将课堂内容融会贯通，在操做中了解汇编语言等新知识，培养成就感，提高专业素养。

1.3 开发环境

- a.基于 32 位 Ubuntu 开源 GNU/Linux 操作系统
- b.使用用 Bochs 开源模拟器器

1.4 参考资料

- a. 开发环境搭建
- b. 参考书籍《一个操作系统的实现》
- c. 参考代码《一个操作系统的实现》

2.项目设计

2.1 总体设计

本系统基于《Orange' s 一一个操作系统的实现》提供的代码，在此基础上对文件系统进行了修改，实现了对文件/目录的增删改写操作，同时实现了多级目录，完成在不同目录级之间的转换操作；添加了开机动画，增加了飞机大战，扫雷，2048 和九宫格等趣味游戏，极大地增强了系统的交互性。

2.2 功能设计

2.2.1 开机动画

- 描述

进入系统后，三只飞鸟飞进后又飞出，引出本操作系统的名称的“Phoenix”，之后展现小组成员信息

■ 实现

动画使用逐帧打印的方式实现，通过对延迟时间的不同安排来形成动画效果。





2.2.2 帮助页面

■ 描述

动画结束后，系统清屏并显示欢迎界面，并显示了当前的路径。此时输入指令“help”后即可浏览当前的指令列表，清晰明了，方便用户使用，增强交互性。

■ 实现

用户输入指令后，判断字符串调用对应 API，进入相应功能。

```

[root@localhost: ~/help
=====Phoenix help info=====

                        Command List
=====

!1.process           || Run the process manage           |
!2.filemng           || Run the file manager             |
!3.clear             || Clear the screen                 |
!4.help              || Show the all the cmd and detail meaning |
!5.ls                || List all files in current directory |
!6.mk                || Create a new file in current directory |
!7.rm                || Delete a file in current directory |
!8.pr                || Print the content of a file in current directory|
!9.vi                || Write new content at the end of the file |
!10.mkdir            || Create a new directory in current directory |
!11.cd               || Go to a directory in current directory |
!12.runtank          || Run a small game on this OS       |
!13.runchess         || Run a small game on this OS       |
!14.run2048          || Run 2048 game on this OS          |
!15.runMS            || Run a small game on this OS       |
=====

[root@localhost: ~/

```

2.2.3 进程管理功能

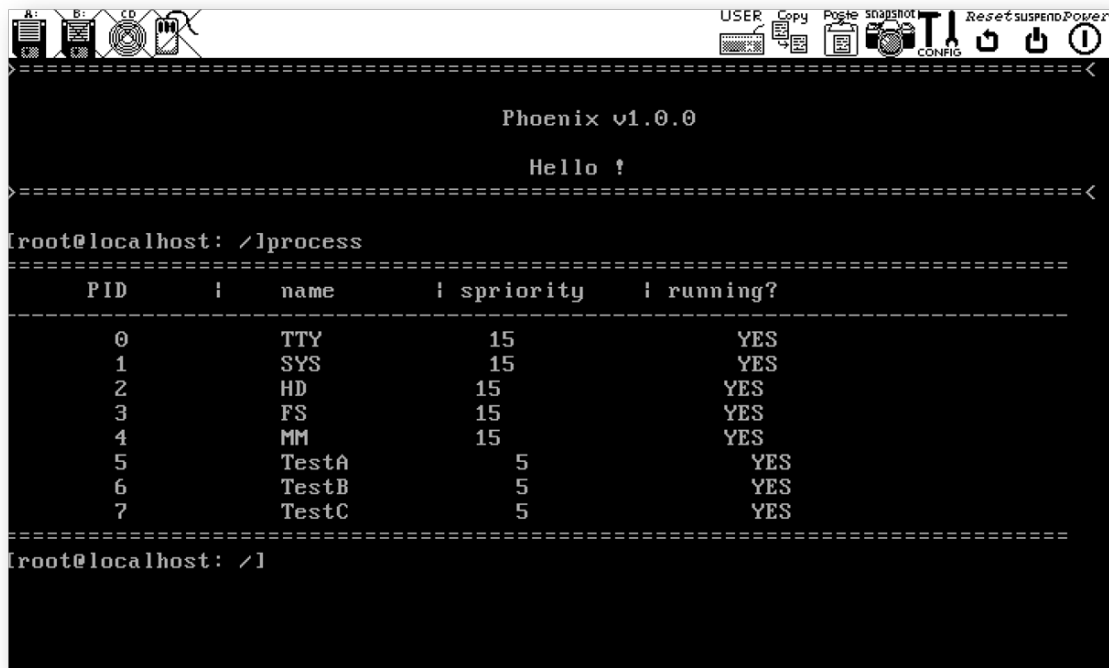
■ 描述

通过“process”指令进入进程管理，显示当前所有进程的 pid 名称和优先级和进程的运行状态，其中 5 表示用户级进程，15 表示系统级进程。

■ 实现

直接调用 ProcessManage()API 实现该功能，可以遍历并逐个打印进程列表。p_flags 的值

表示了进程的运行状态



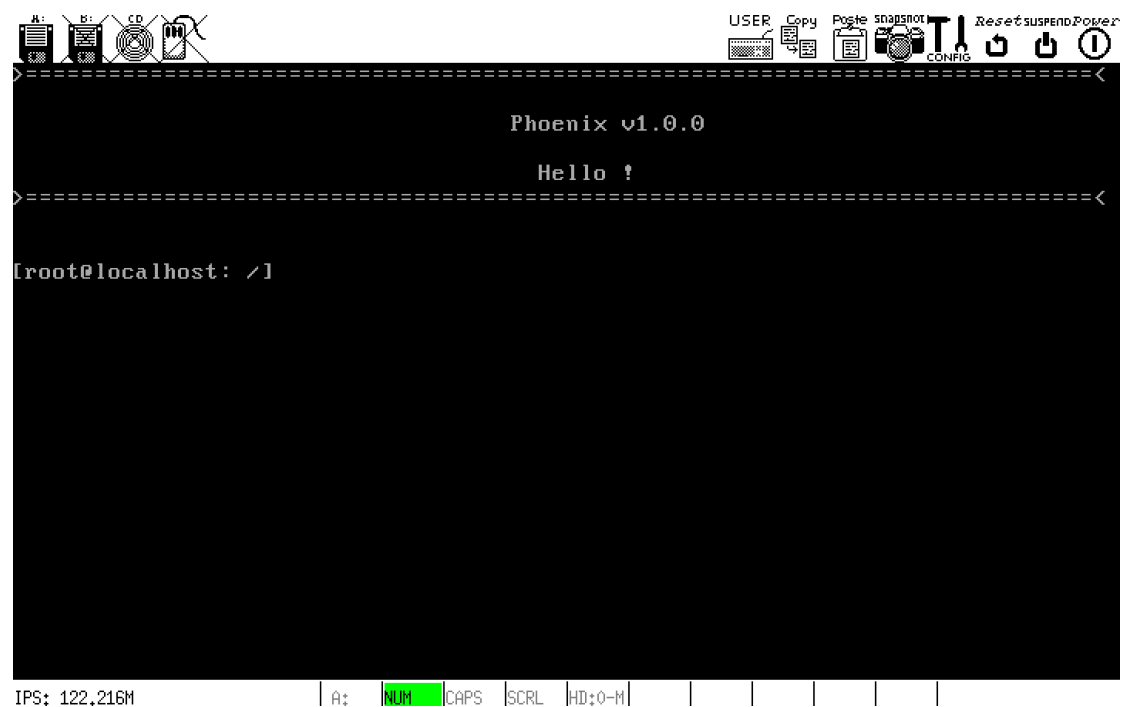
2.2.4 清屏

■ 描述

当用户输入“clear”指令时，系统清屏，显示初始的欢迎界面。

■ 实现

直接调用 clear () API 实现清屏功能，同时重新设置相应的处置



2.2.5 文件管理

2.2.5.1 文件管理

■ 描述

用户输入“filemng”指令，系统显示当前的控制台，文件管理系统的运行情况。

■ 实现

直接调用 filemng () API 实现文件管理功能。



2.2.5.2 显示文件列表

■ 描述

用户输入“ls”指令，系统打印出当前文件目录下的全部文件及目录

■ 实现

直接调用 ls(), 系统找到当前目录下的全部文件名


```
root@localhost: /lls

node    type    filename
=====
1       file    .
2       file    dev_tty0
3       file    dev_tty1
4       file    dev_tty2
5       [dir]   a
6       file    qqg
=====
```

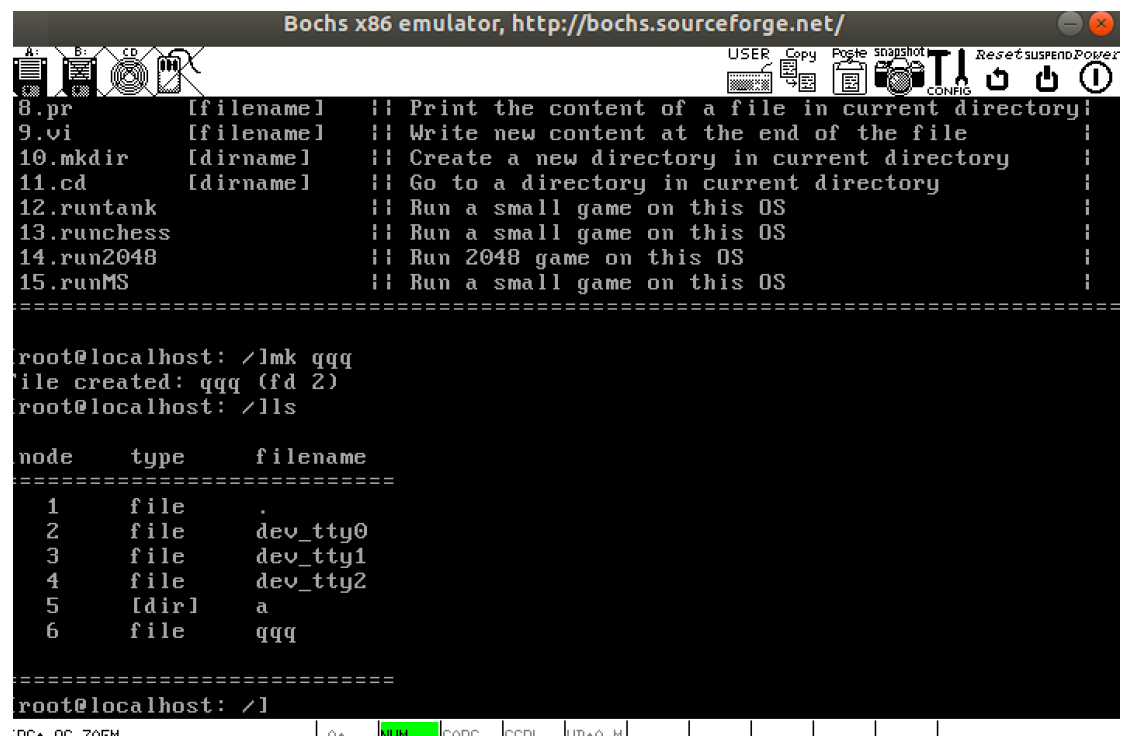
2.2.5.3 创建文件

■ 描述

用户输入“mk”指令并在之后加上所要创建文件的文件名，即在当前目录下创建一个新文件，系统给出创建成功或失败的提示。

■ 实现

调用 open()方法创建文件，根据返回值判断是否创建成功并发送反馈信息。



```
Bochs x86 emulator, http://bochs.sourceforge.net/

8.pr      [filename]    ;; Print the content of a file in current directory;
9.wi      [filename]    ;; Write new content at the end of the file
10.mkdir   [dirname]    ;; Create a new directory in current directory
11.cd      [dirname]    ;; Go to a directory in current directory
12.runtank                ;; Run a small game on this OS
13.runchess               ;; Run a small game on this OS
14.run2048                ;; Run 2048 game on this OS
15.runMS                 ;; Run a small game on this OS

=====

root@localhost: /lmk qqg
file created: qqg (fd 2)
root@localhost: /lls

node    type    filename
=====
1       file    .
2       file    dev_tty0
3       file    dev_tty1
4       file    dev_tty2
5       [dir]   a
6       file    qqg
=====

root@localhost: /l
```

2.2.5.4 删除文件

■ 描述

用户输入“rm”后再加上“filename”的指令，删除指定文件名的文件，系统给出删除成功或失败的提示。（本系统不允许相同目录下文件重名）

■ 实现

调用 `unlink()` 方法创建文件，判断是否创建成功并发送反馈信息

```
=====
root@localhost: /lrm qqg
qq deleted!
root@localhost: /l
```

IPS: 99.295M | A: NUM | CAPS | SCRL | HD: 0-M | | | | | |

2.2.5.5 打印文件内容

■ 描述

用户输入 “pr+文件名” 对文件的内容进行读取，系统显示文件内容。

■ 实现方法

先调用 `open()` 方法定位文件，再调用 `read()` 方法对文件进行读操作。

```
Bochs x86 emulator, http://bochs.sourceforge.net/
=====
5 [dir] a
6 file qqg

=====
root@localhost: /lrm qqg
qq deleted!
root@localhost: /lmk qqg
file created: qqg (fd 2)
root@localhost: /lls

node  type  filename
=====
1      file  .
2      file  dev_tty0
3      file  dev_tty1
4      file  dev_tty2
5      [dir] a
6      file  qqg

=====
root@localhost: /lvi qqg
23
root@localhost: /lpr qqg
23
root@localhost: /l
```

PS: 90.585M | A: NUM | CAPS | SCRL | HD: 0-M | | | | | |

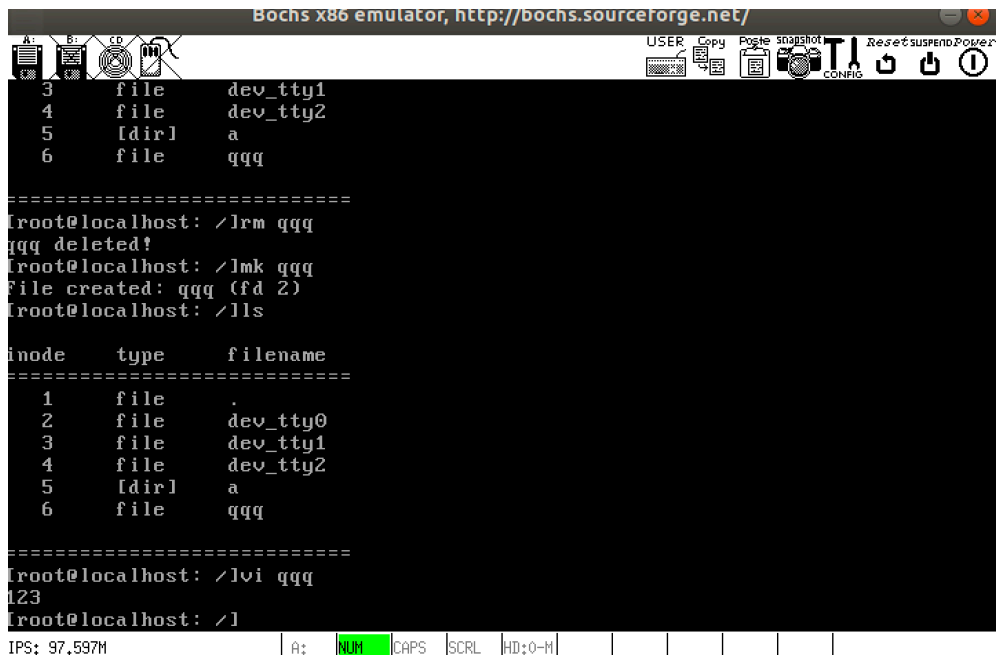
2.2.5.6 写文件

■ 描述

用户输入 “vi+文件名” 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车，完成写文件。

■ 实现方法

首先使用 open()方法定位文件，再调用 write()方法对文件进行写操作。



```
Bochs x86 emulator, http://bochs.sourceforge.net/
=====
3   file   dev_tty1
4   file   dev_tty2
5   [dir]   a
6   file   qqg

=====
[root@localhost: /]rm qqg
qqg deleted!
[root@localhost: /]mk qqg
File created: qqg (fd 2)
[root@localhost: /]ls

inode    type    filename
=====
1        file    .
2        file    dev_tty0
3        file    dev_tty1
4        file    dev_tty2
5        [dir]   a
6        file    qqg

=====
[root@localhost: /]vi qqg
123
[root@localhost: /]

IPS: 97,597M | A: NUM | CAPS | SCRL | HD:0-M |
```

2.2.5.7 创建目录

■ 描述

用户输入“mkdir+目录名”指令，在当前目录内创建一个目录，成功或失败都会有相应提示信息。

■ 实现方法

先调用 open()创建对应的文件路径，后调用 mkdir()创建目录。

```
Bochs x86 emulator, http://bochs.sourceforge.net/
USER Copy Paste Snapshot CONFIG Reset suspend Power
8.pr [filename] !! Print the content of a file in current directory!
9.vi [filename] !! Write new content at the end of the file
10.mkdir [dirname] !! Create a new directory in current directory
11.cd [dirname] !! Go to a directory in current directory
12.runtank !! Run a small game on this OS
13.runchess !! Run a small game on this OS
14.run2048 !! Run 2048 game on this OS
15.runMS !! Run a small game on this OS
=====
root@localhost: /lls
inode      type      filename
=====
1          file      .
2          file      dev_tty0
3          file      dev_tty1
4          file      dev_tty2
5          [dir]     a
6          file      qqg
=====
root@localhost: /mkdir bbb
creating directory /bbb succeeded!
root@localhost: /_
IPS: 97.260M  A: NUM CAPS SCRL HD:0-M
```

2.2.5.8 进入多级目录

■ 描述

用户输入“cd+目录名”指令，系统检索目录路径，检索成功后，指令输入区的路径由“/”更改为“/目录名/”，表示当前处于目标目录中。

■ 实现方法

调用 GoDir()方法，首先根据输入的目录名判断要进入下一级目录还是返回上一级目录，若返回上一级，则先找到目标位置，再将绝对路径修改为对应路径并更新；若进入下一级，则直接修改路径并更新。

```
Bochs x86 emulator, http://bochs.sourceforge.net/
USER Copy Paste Snapshot CONFIG Reset suspend Power
6          file      qqg
=====
root@localhost: /mkdir bbb
creating directory /bbb succeeded!
root@localhost: /lls
inode      type      filename
=====
1          file      .
2          file      dev_tty0
3          file      dev_tty1
4          file      dev_tty2
5          [dir]     a
6          file      qqg
7          [dir]     bbb
=====
root@localhost: /lcd bbb
root@localhost: /bbb/lls
inode      type      filename
=====
root@localhost: /bbb/_
IPS: 97.317M  A: NUM CAPS SCRL HD:0-M
```

2.2.6 游戏功能

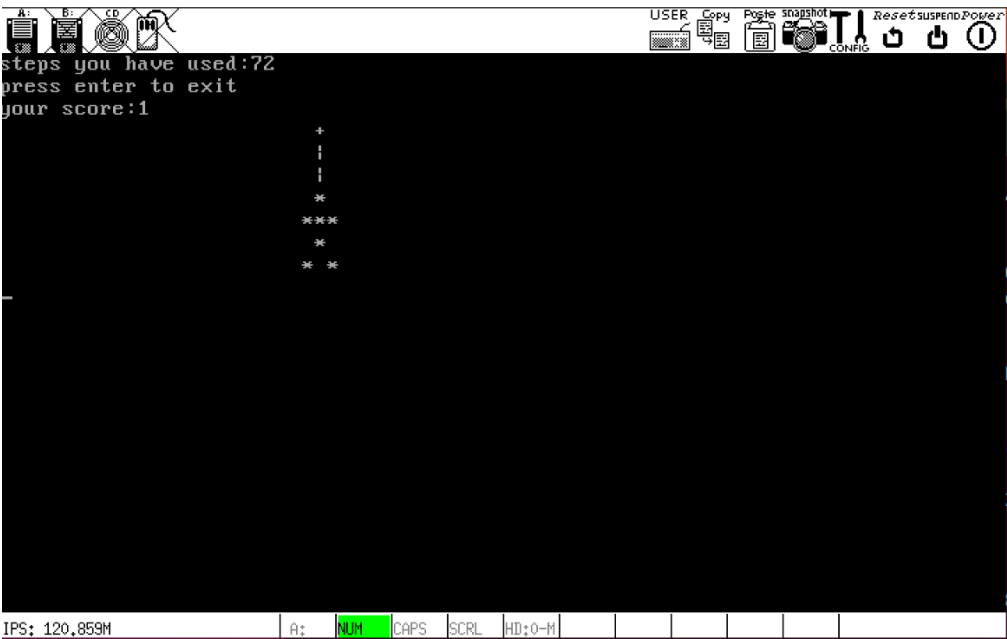
飞机大战

■ 描述

用户输入“runtank”指令，进入飞机大战游戏界面；系统打印一个飞机与一个靶子，用户通过 w,a,s,d 来控制上下左右，space 键发射炮弹，每击中一个靶子积一分，按 enter 键退出游戏，返回起始界面。

■ 实现方法

调用 Tank()方法，实现飞机大战相关操作。



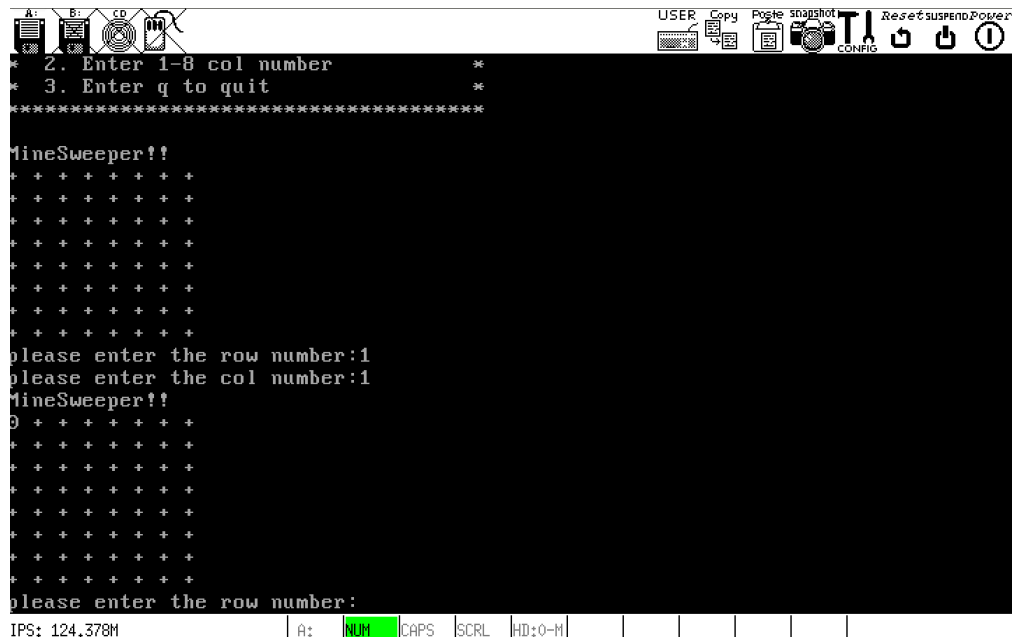
扫雷

■ 描述

用户输入“runMS”指令，进入扫雷游戏界面；系统打印游戏指示与扫雷地图，用户通过输入要查看的行列数进行操作，输入后地图显示该坐标周围的雷的个数，若踩到雷，则游戏自动结束；在游戏过程中，用户也可以随时按 q 键退出游戏。

■ 实现方法

调用 MS()方法，通过二维数组实现一个扫雷游戏。



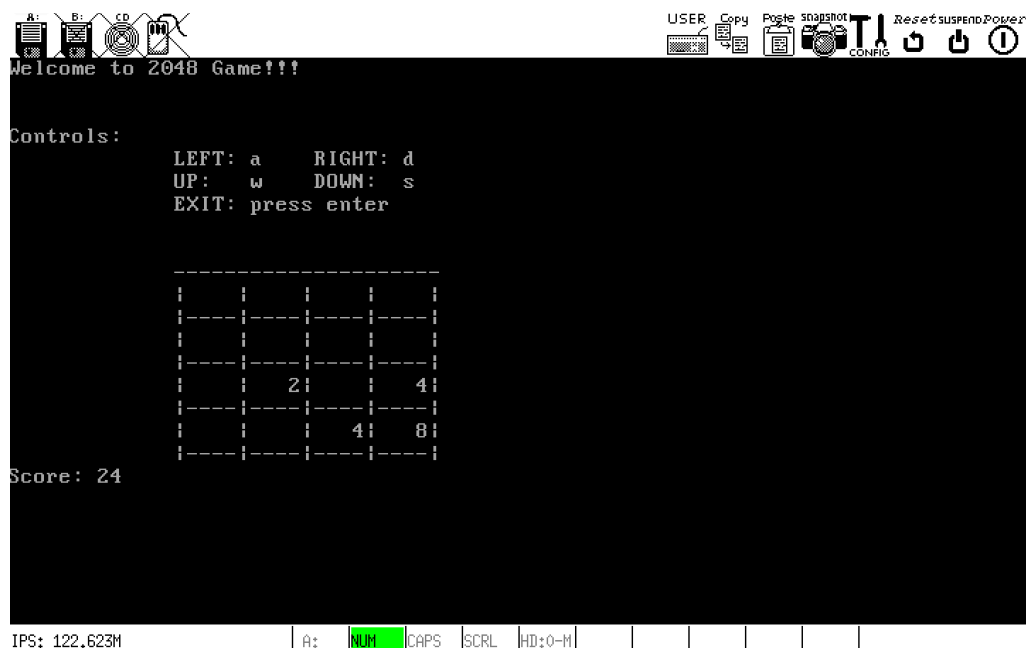
2048

■ 描述

用户输入“run2048”指令，进入2048游戏界面；系统打印游戏控制方法w,a,s,d控制上下左右，enter键退出游戏以及初始地图与初始分数，用户输入操作后系统自动刷新棋盘以及分数。

■ 实现方法

调用start2048Game()方法，其中调用多个2048相关函数，共同实现2048游戏的操作。



井字棋游戏

■ 描述

用户输入 runchess 进入游戏，系统打印初始棋盘并询问用户选择先手或后手，用户通过输入行列数进行游戏。

■ 实现方法

调用 TTT()方法实现游戏。

```
Please Input The Line Position where you put your Chessman (x): 2
Please Input The Column Position where you put your Chessman (y): 2
Input Error!Please Input The Line Position where you put your Chessman (x): 3
Please Input The Column Position where you put your Chessman (y): 3
The computer put a Chessman at: 3,2
The QP now is:
0 0 0
0 0 0
0 0 0
Please Input The Line Position where you put your Chessman (x): 1
Please Input The Column Position where you put your Chessman (y): 2
The computer put a Chessman at: 1,3
The QP now is:
0 0 0
0 0 0
0 0 0
Please Input The Line Position where you put your Chessman (x): 3
Please Input The Column Position where you put your Chessman (y): 1
The computer put a Chessman at: 2,3
The QP now is:
0 0 0
0 0 0
0 0 0
DRAW GAME!
Play Again?(y/n)
```

3.核心代码

3.1 用户及应用

3.1.1 扫雷

//使用二维数组实现 扫雷

```
void MS(int fd_stdin,int fd_stdout)
```

```
{
```

```
    clear();
```

```
    char ui[8][8]={
```

```
        '+','+', '+','+', '+','+', '+','+',
```

```
        '+','+', '+','+', '+','+', '+','+',
```

```
        '+','+', '+','+', '+','+', '+','+',
```

```
        '+','+', '+','+', '+','+', '+','+',
```

```

        '+' , '+' , '+' , '+' , '+' , '+' , '+' , '+' ,
        '+' , '+' , '+' , '+' , '+' , '+' , '+' , '+' ,
        '+' , '+' , '+' , '+' , '+' , '+' , '+' , '+' ,
        '+' , '+' , '+' , '+' , '+' , '+' , '+' , '+'

};

int map[8][8]={

    0,0,0,0,0,0,0,0,
    0,0,1,0,0,1,0,0,
    0,0,0,0,1,0,0,0,
    0,0,0,0,0,1,0,0,
    0,0,1,0,0,0,0,0,
    0,0,1,0,0,0,0,0,
    0,1,0,1,1,0,0,0,
    1,0,0,0,0,0,0,0

};

int p[8][2]={ {-1,-1} , {-1,0} , {-1,1} , {0,-1} , {0,1} , {1,-1} , {1,0} , {1,1} };

int i=0,j=0;

int h=0,l=0;

int h1=0,l1=0;

int n=0;//用来保存 雷的数量 计数

int win=0;

char x[2];

char y[2];


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

printf("*           Minesweeper           *\n");

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

```



```

printf("* 1. Enter 1-8 row number      *\n");
printf("* 2. Enter 1-8 col number      *\n");
printf("* 3. Enter q to quit            *\n");
printf("*****\n\n");
while(1)
{
    printf("MineSweeper!!\n");

    for(i=0;i<8;i++)
    {
        for(j=0;j<8;j++)
        {
            printf("%c ",ui[i][j]);
        }
        printf("\n");
    }

    printf("please enter the row number:");

    ClearArr(x, 2);

    int r = read(0, x, 2);

    h = x[0]-49+1;

    if(x[0] == 'q')
    {
        clear();

        break;
    }
}

```

```

while(!(h >=0 && h <= 7))
{
    printf("Wrong input!\nplease enter the row number:");

    r = read(0, x, 2);

    h = x[0]-49+1;
}

printf("please enter the col number:");

ClearArr(x, 2);

int r1 = read(0, y, 2);

l = y[0]-49+1;

if(y[0] == 'q')
{
    clear();

    break;
}

while(!(l >=0 && l <= 7))
{
    printf("Wrong input!\nplease enter the col number:");

    r1 = read(0, y, 2);

    l = y[0]-49+1;
}

if(map[h-1][l-1]==1)
{
    printf("GAME OVER!\n");

    break;
}

h=h-1;

```

```
l=l-1;
```

```
//没有踩到雷的情况 判断周围有几个雷 并把数字显示在 界面上
```

```
//-1 -1    -1, 0    -1,+1    0 ,-1    0 ,+1    +1 ,-1    +1,0    +1 ,+1
```

```
//n=map[h-1][l-1]+map[h-1][l]+map[h-1][l+1]+map[h][l-1]+map[h][l+1]+map[h+1][l-1]+map[h+1][l+1]+map[h+1][l];
```

```
i=0;
```

```
while(i<8)
```

```
{
```

```
    n=0;
```

```
    h1=h;
```

```
    l1=l;
```

```
    h1= h1+p[i][0];
```

```
    l1=l1+p[i][1];
```

```
    if(h1>=0&&h1<8&&l1>=0&&l1<8)
```

```
    {
```

```
        if(map[h1][l1]==1)
```

```
        {
```

```
            n++;
```

```
        }
```

```
    }
```

```
    i++;
```

```
}
```

```
//把得到的数字显示到 界面上 ui[h][l];
```

```
//把 int 数字转换成 字符
```

```
switch(n)
```

```
{
```

```
    case 0:
```

```
        ui[h][l]='0';
```

```
        break;
```

```
    case 1:
```

```
        ui[h][l]='1';
```

```
        break;
```

```
    case 2:
```

```
        ui[h][l]='2';
```

```
        break;
```

```
    case 3:
```

```
        ui[h][l]='3';
```

```
        break;
```

```
    case 4:
```

```
        ui[h][l]='4';
```

```
        break;
```

```
    case 5:
```

```
        ui[h][l]='5';
```

```
        break;
```

```
    case 6:
```

```
        ui[h][l]='6';
```

```
        break;
```

```
    case 7:
```

```

        ui[h][l]='7';

        break;

    case 8:

        ui[h][l]='8';

        break;

    }

    win++;

    if(win==54)

    {

        printf("You win!\n");

        break;

    }

}

return 0;

}

```

3.1.2 飞机大战

```

void Tank(int fd_stdin, int fd_stdout)

{

    int pox, poy;

    int x = 10;

    int y = 25;

    //可以定义一个 input 输入框，用于控制移动方向

    char input[1] = { 0 };

    // 是否发射子弹

    int isFired = 0;

```

```

////////////////////////////////////
//

//用加号代替靶子

int king_y = 5;//这是靶子的 y 坐标。靶子用“+”表示

//是否被击中

int isKilled = 0;

//计分器

int score = 0;

//计数器

long int count = 0;

//敌机移动 flag

int flag = 0;

//加入 while 循环

while (1)
{
    clear();

    printf("steps you have used:%d",count);

    printf("\n");

    printf("press enter to exit");

    printf("\n");

    printf("your score:%d",score);

    printf("\n");

    for (poy = 0; poy < king_y; poy++)
    {
        printf(" ");
    }

    printf("+\n");
}

```

```
//如果不开火，则不发射子弹
```

```
if (isFired == 0)
```

```
{
```

```
    for (pox = 0; pox < x; pox++)
```

```
    {
```

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

```
    }
```

```
}
```

```
else
```

```
{
```

```
    for (pox = 0; pox < x; pox++)
```

```
    {
```

```
        for (poy = 0; poy < y; poy++)
```

```
        {
```

```
            printf(" ");
```

```
        }
```

```
        printf(" | \n");
```

```
    }
```

```
if (king_y == (y + 2))
```

```
{
```

```
    score++;
```

```
    flag=0;
```

```
    srand(count);
```

```
    king_y = rand()%50+2;
```

```
}
```

```
isFired = 0; //按一次空格发射一次子弹，重置为 0;
```

```

}

if (flag == 2)
{
    flag=0;

    srand(count);

    if(king_y>2)

        king_y += rand()%3-1;
}

//雪花作为飞机

for (poy = 0; poy < y; poy++)

    printf(" ");

printf(" * \n");

for (poy = 0; poy < y; poy++)

    printf(" ");

printf(" *** \n");

for (poy = 0; poy < y; poy++)

    printf(" ");

printf(" * \n");

for (poy = 0; poy < y; poy++)

    printf(" ");

printf(" * * \n");


// x++; //此 x 的功能是让雪花点向下移动。

if(!read(fd_stdin, input, 1))

    break;

flag++;

//scanf 通过获取你输入的字符串进行移动雪花点。

```



```

//scanf("%c",&input); //输入字符需要按回车键才能生效、

if (input[0] == 'w')

    x--;//雪花点向上移动


if (input[0] == 's')

    x++;//雪花点向下移动


if (input[0] == 'a')

    y--;//向左


if (input[0] == 'd')

    y++;//向右


if (input[0] == ' ')//空格作为开火键

    isFired = 1;

    count++;

}

clear();

}

```

3.1.3 井字棋游戏

```

void TTT(int fd_stdin,int fd_stdout)

{

    char buf[80]={0};

    char IsFirst = 0;

    int IsFinish = FALSE;

    while(!IsFinish)

```

```

{

    Init();

    printf("The QiPan (QP) is: \n");

    PrintQP();

    printf("Do you want do first?(y/n):");
    read(fd_stdin,buf,2);
    IsFirst = buf[0];
    do{

        if(IsFirst=='y')
        {
            UserInput(fd_stdin, fd_stdout);
            IsFinish=AutoDone();
        }else{
            IsFinish=AutoDone();
            if(!IsFinish)UserInput(fd_stdin, fd_stdout);
        }

    }while(!IsFinish);
    if (IsFinish)
    {
        printf("Play Again?(y/n)");
        char cResult;
        read(fd_stdin,buf,2);
    }
}

```

```

        cResult = buf[0];
        printf("%c",cResult);

        if (cResult == 'y')
        {
            clear();

            IsFinish = FALSE;

        }

        else
        {
            clear();
        }

    }

}

}

```

3.1.4 2048

```

PUBLIC void start2048Game(int fd_stdin, int fd_stdout) {

    // Specify the rules of the game

    clear();

    printf("Welcome to 2048 Game!\n\n\n");
    printf("Control:\n");
    printf("          LEFT: a   RIGHT: d\n");
}

```

```

printf("          UP:  w   DOWN:  s\n");

printf("          EXIT: press enter  \n\n\n");

// Initialize the data

initData();


// Initalization

addrandom2048();

addrandom2048();

printNums2048();


// Turns in loops

// while (scanf(" %c", &option)) {
while (read(fd_stdin, option2048, 1)) {

    clear();

    printf("Welcome to 2048 Game!!!\n\n\n");

    printf("Controls:\n");

    printf("          LEFT: a   RIGHT: d\n");

    printf("          UP:  w   DOWN:  s\n");

    printf("          EXIT: press enter  \n\n\n");

    // Check if the player is dead

    if (!isAlive2048()) {

        printf("You lose!!!\a\n");

        break;

    }

    morge2048();

```

```

        if (validity2048) {
            addrandom2048();
        }

        validity2048 = 0;

        printNums2048();
    }

    clear();
}

```

3.2 系统级应用

3.2.1 进程管理

```

void ProcessManage()
{
    int i;

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

    printf("    PID    |   name    | spriority   | running?\n");
    //进程号, 进程名, 优先级, 是否是系统进程, 是否在运行

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

    for ( i = 0 ; i < NR_TASKS + NR_PROCS ; ++i )//逐个遍历
    {
        /*if ( proc_table[i].priority == 0) continue;//系统资源跳过*/

        if(proc_table[i].p_flags!=FREE_SLOT)

            printf("    %d    %s    %d    %s\n",
proc_table[i].pid, proc_table[i].name, proc_table[i].priority,
proc_table[i].p_flags==FREE_SLOT? "NO":"YES");

```

```

    }

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

}

```

3.2.2 文件管理

```

void CreateFile(char* path, char* file)
{
    char absoPath[512];
    convert_to_absolute(absoPath, path, file);

    int fd = open(absoPath, O_CREAT | O_RDWR);

    if (fd == -1)
    {
        printf("Failed to create a new file with name %s\\n", file);
        return;
    }

    char buf[1] = {0};
    write(fd, buf, 1);
    printf("File created: %s (fd %d)\\n", file, fd);
    close(fd);
}

void DeleteFile(char* path, char* file)
{
    char absoPath[512];
    convert_to_absolute(absoPath, path, file);
    int m=unlink(absoPath);
    if (m == 0)
        printf("%s deleted!\\n", file);
    else
        printf("Failed to delete %s!\\n", file);
}

void ReadFile(char* path, char* file)
{
    char absoPath[512];
    convert_to_absolute(absoPath, path, file);
    int fd = open(absoPath, O_RDWR);
    if (fd == -1)

```

```

{
    printf("Failed to open %s!\n", file);
    return;
}

char buf[4096];
int n = read(fd, buf, 4096);
if (n == -1) // 读取文件内容失败
{
    printf("An error has occurred in reading the file!\n");
    close(fd);
    return;
}

printf("%s\n", buf);
close(fd);
}

void WriteFile(char* path, char* file)
{
    char absoPath[512];
    convert_to_absolute(absoPath, path, file);
    int fd = open(absoPath, O_RDWR);
    if (fd == -1)
    {
        printf("Failed to open %s!\n", file);
        return;
    }

    char tty_name[] = "/dev_tty0";
    int fd_stdin = open(tty_name, O_RDWR);
    if (fd_stdin == -1)
    {
        printf("An error has occurred in writing the file!\n");
        return;
    }

    char writeBuf[4096]; // 写缓冲区
    int endPos = read(fd_stdin, writeBuf, 4096);
    writeBuf[endPos] = 0;
    write(fd, writeBuf, endPos + 1); // 结束符也应写入
    close(fd);
}

```

3.3 其它功能

3.3.1 清屏

```
void clear()
{
    clear_screen(0,console_table[current_console].cursor);

    console_table[current_console].crtc_start = 0;

    console_table[current_console].cursor = 0;

}
```

4.成员分工

成员名称	学号	占比
谢康（组长）	1753402	25%
谢尚汝	1754188	25%
刘岚心	1754197	25%
周贤	1751251	25%