

YeezyOS v1.0.0

操作系统课设项目

组号：13

组长：杨雨辰

组员：季潇熠、雷泓、张艺腾、赵希明

1 项目描述

- 1.1 项目目的
- 1.2 开发环境
- 1.3 项目选题及达成指标

2 设计方案

- 2.1 概述
- 2.2 系统功能及使用说明
 - 2.2.1 开机动画
 - 2.2.2 帮助界面/主界面
 - 2.2.3 清屏功能
 - 2.2.4 进程管理
 - 2.2.5 文件管理
 - 2.2.5.1 ls
 - 2.2.5.2 mkdir
 - 2.2.5.3 touch
 - 2.2.5.4 remove (rm)
 - 2.2.5.5 write (wt)
 - 2.2.5.6 read (rd)
 - 2.2.5.7 cd
 - 2.2.6 游戏及工具
 - 2.2.6.1 游戏界面
 - 2.2.6.2 sudoku (数独)
 - 2.2.6.3 bwchess (黑白棋)
 - 2.2.6.4 tic-tac-toe (井字棋)
 - 2.2.6.5 mine (扫雷)
 - 2.2.6.5 carrycraft (推箱子)
 - 2.2.6.6 calendar (日历)
 - 2.2.6.7 calculator (计算器)

3 项目核心代码

- 3.1 进程管理
- 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

3.2.1.2 文件的查找与创建

3.2.2 文件系统用户功能

3.2.2.1 写文件

3.2.2.2 读文件

3.2.2.3 创建文件

3.2.2.4 展示文件列表

3.2.2.5 切换文件目录

3.2.2.6 删除文件

3.3 游戏及工具

3.3.1 游戏

3.3.1.1 sudoku (数独)

3.3.1.2 bwchess (黑白棋)

3.3.1.3 tic-tac-toe (井字棋)

3.3.1.4 mine (扫雷)

3.3.1.5 carrycraft (推箱子)

3.3.2 工具

3.3.2.1 calendar (日历)

3.3.2.2 calculator (计算器)

4 成员分工

1 项目描述

1.1 项目目的

操作系统课程设计旨在让我们通过学习并实现一个简单而功能完善的操作系统，来加深对操作系统中进程管理、内存管理、文件管理等概念及原理的理解，并真正了解一个操作系统是如何从无到有、一步步实现的。

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

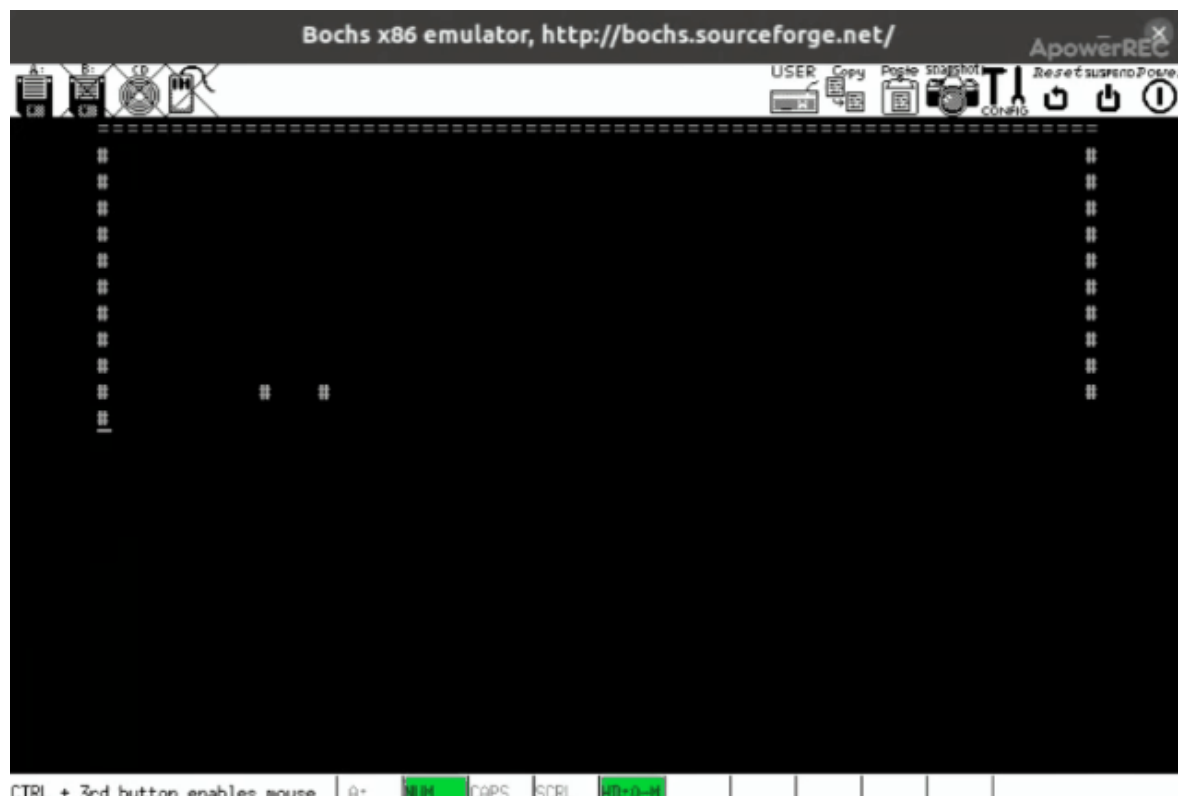
2.1 概述

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，并重新实现了其中文件系统的部分，包括多级目录，文件/目录的增加及删除，文件的读写等；同时通过调用部分系统API实现了数独、黑白棋、井字棋等小游戏，以及日历、计算器等工具，还为其添加了开机动画。

2.2 系统功能及使用说明

- 在 yeezyOS 打开终端
- 输入 `bochs -> 6 -> c`
- 进入 yeezyOS 操作系统

2.2.1 开机动画



2.2.2 帮助界面/主界面

菜单显示系统提供的所有功能编号及概要。输入各命令或其编号，进入相应功能界面。

若想要回到 帮助界面/主界面，在命令行中输入“help”按下回车即可。



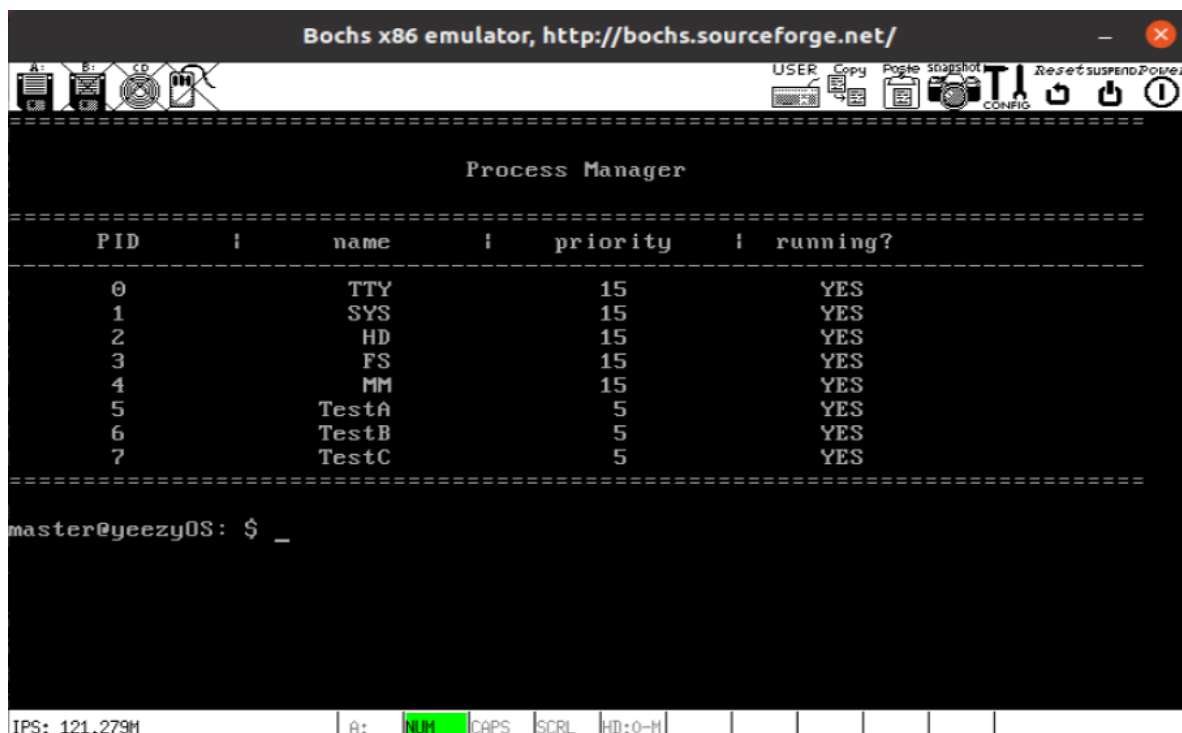
当需要清屏时，输入 `clear` 按下回车即可。



2.2.4 进程管理

描述

用户输入指令“**process**”，进入进程管理功能，系统将打印出当前所有进程的PID、进程名称、优先级（15表示系统级进程，5表示用户级进程）以及运行状态。



功能实现

- 调用 `process_manager()`，对进程列表 `proc_table[]` 进行遍历，逐个打印每个进程的信息，其中宏 `NR_TASKS + NR_PROCS` 表示所有进程的数量。

2.2.5 文件管理

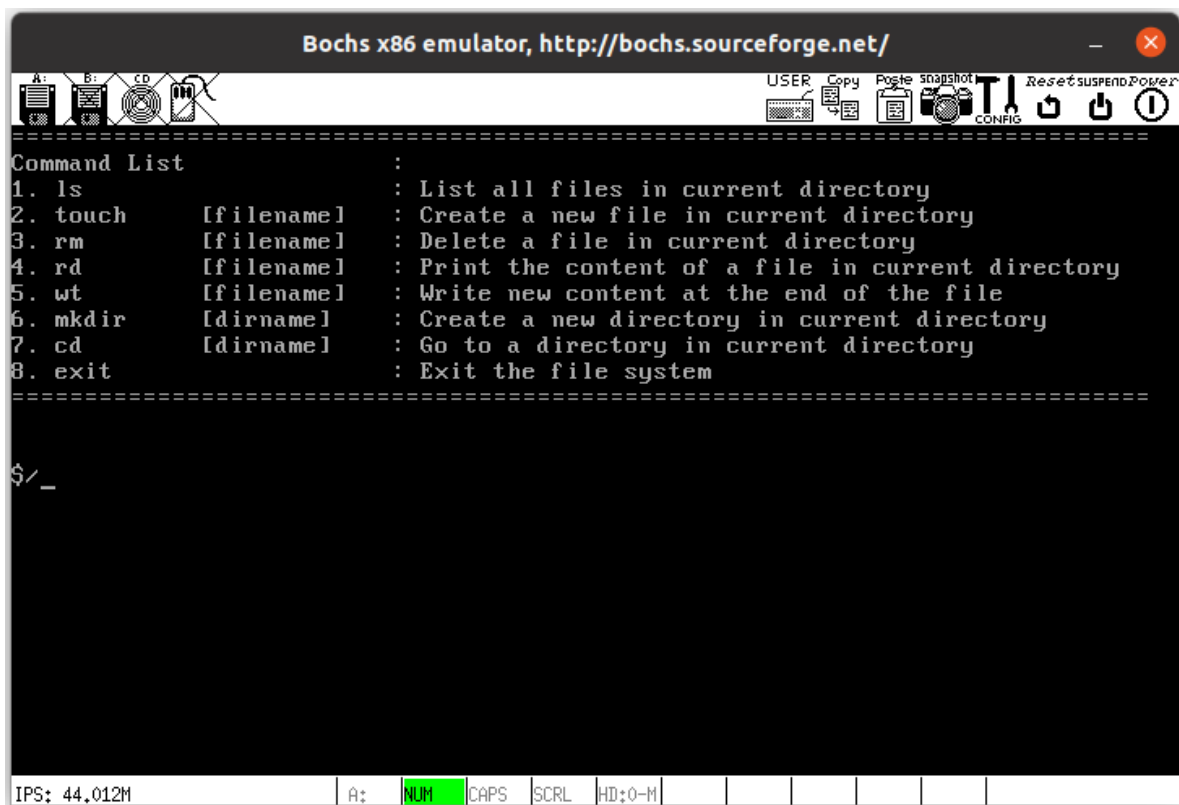
描述：

`runFileManager()`: 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：



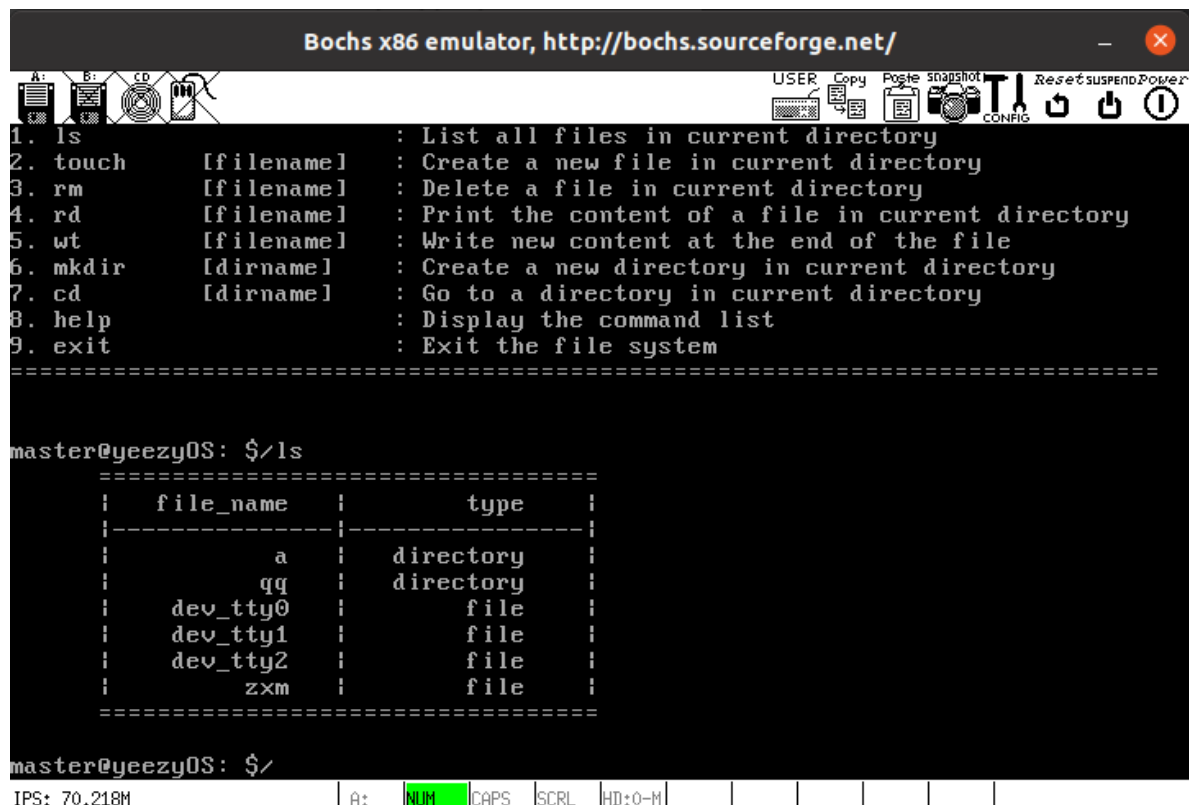
2.2.5.1 ls

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。



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。

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

=====
Command List
=====
1. ls                : List all files in current directory
2. touch [filename]  : Create a new file in current directory
3. rm [filename]     : Delete a file in current directory
4. rd [filename]     : Print the content of a file in current directory
5. wt [filename]     : Write new content at the end of the file
6. mkdir [dirname]   : Create a new directory in current directory
7. cd [dirname]      : Go to a directory in current directory
8. exit              : Exit the file system
=====

$ /ls
directories: . dir1
files: dev_tty0 dev_tty1 dev_tty2 file1 file2
$ /mkdir dir1
mkdir error: directory exists.
$ /mkdir dir2
$ /ls
directories: . dir1 dir2
files: dev_tty0 dev_tty1 dev_tty2 file1 file2
$ /_

IPS: 41.862M | A: NUM | CAPS | SCRL | HD:0-M | | | | | | | |
```

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。


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

=====
Command List
=====
1. ls          : List all files in current directory
2. touch      [filename] : Create a new file in current directory
3. rm         [filename] : Delete a file in current directory
4. rd         [filename] : Print the content of a file in current directory
5. wt         [filename] : Write new content at the end of the file
6. mkdir      [dirname]  : Create a new directory in current directory
7. cd         [dirname]  : Go to a directory in current directory
8. exit       : Exit the file system
=====

$/ls
directories: . dir1 dir2
files: dev_tty0 dev_tty1 dev_tty2 file1 file2
$/touch file1
touch error: file exists.
$/touch file3
$/ls
directories: . dir1 dir2
files: dev_tty0 dev_tty1 dev_tty2 file1 file2 file3
$/
$/>
IPS: 41,799M  A: NUM CAPS SCRL HD:0-M
```

2.2.5.4 remove (rm)

描述

用户进入文件系统某一目后，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。

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

=====
Command List
=====
1. ls          : List all files in current directory
2. touch      [filename] : Create a new file in current directory
3. rm         [filename] : Delete a file in current directory
4. rd         [filename] : Print the content of a file in current directory
5. wt         [filename] : Write new content at the end of the file
6. mkdir      [dirname]  : Create a new directory in current directory
7. cd         [dirname]  : Go to a directory in current directory
8. exit       : Exit the file system
=====

$/rm file2
dump_fd_graph: UNLINK just finished. (pid:5)
<!!_i0123456!>file2 deleted!
$/rm -r dir2
dump_fd_graph: UNLINK just finished. (pid:5)
<!!_i0123456!>dir2 deleted!
$/ls
directories: . dir1
files: dev_tty0 dev_tty1 dev_tty2 file1 file3
$/
$/>
IPS: 41,914M  A: NUM CAPS SCRL HD:0-M
```

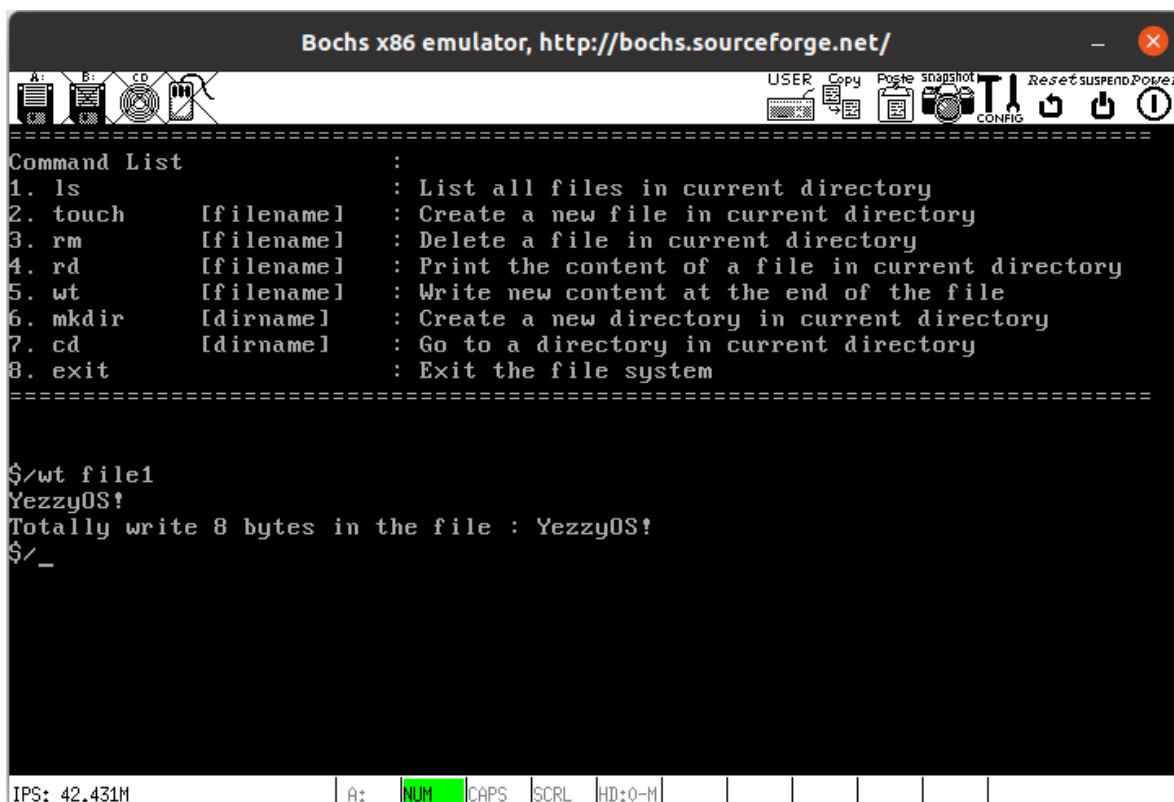
2.2.5.5 write (wt)

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车，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。



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

=====
Command List
1. ls                : List all files in current directory
2. touch [filename]  : Create a new file in current directory
3. rm [filename]     : Delete a file in current directory
4. rd [filename]     : Print the content of a file in current directory
5. wt [filename]     : Write new content at the end of the file
6. mkdir [dirname]   : Create a new directory in current directory
7. cd [dirname]      : Go to a directory in current directory
8. exit              : Exit the file system
=====

$/wt file1
YezzyOS!
Totally write 8 bytes in the file : YezzyOS!
$/_

IPS: 42,431M  A: NUM  CAPS  SCRL  HD: 0-M
```

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。

Bochs x86 emulator, <http://bochs.sourceforge.net/>

Command List :

1. ls : List all files in current directory
2. touch [filename] : Create a new file in current directory
3. rm [filename] : Delete a file in current directory
4. rd [filename] : Print the content of a file in current directory
5. wt [filename] : Write new content at the end of the file
6. mkdir [dirname] : Create a new directory in current directory
7. cd [dirname] : Go to a directory in current directory
8. exit : Exit the file system

```

$ /wt file1
YezzyOS!
Totally write 8 bytes in the file : YezzyOS!
$ /rd file1
Content of file file1 :
YezzyOS!
$ /rd file2
Content of file file2 :

$ /
  
```

IPS: 42.859M | A: NUM | CAPS | SCRL | HD: 0-M |

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。

Bochs x86 emulator, <http://bochs.sourceforge.net/>

```

master@yeezyOS: $ /ls
=====
| file_name | type |
|-----|-----|
| a         | directory |
| qq        | directory |
| dev_tty0  | file |
| dev_tty1  | file |
| dev_tty2  | file |
| zxm       | file |
=====

master@yeezyOS: $ /cd zxm
No such directory.
master@yeezyOS: $ /cd qq
master@yeezyOS: $ /qq/ls
=====
| file_name | type |
|-----|-----|
| al        | file |
=====

master@yeezyOS: $ /qq/
  
```

IPS: 108.073M | A: NUM | CAPS | SCRL | HD: 0-M |

2.2.6 游戏及工具

2.2.6.1 游戏界面

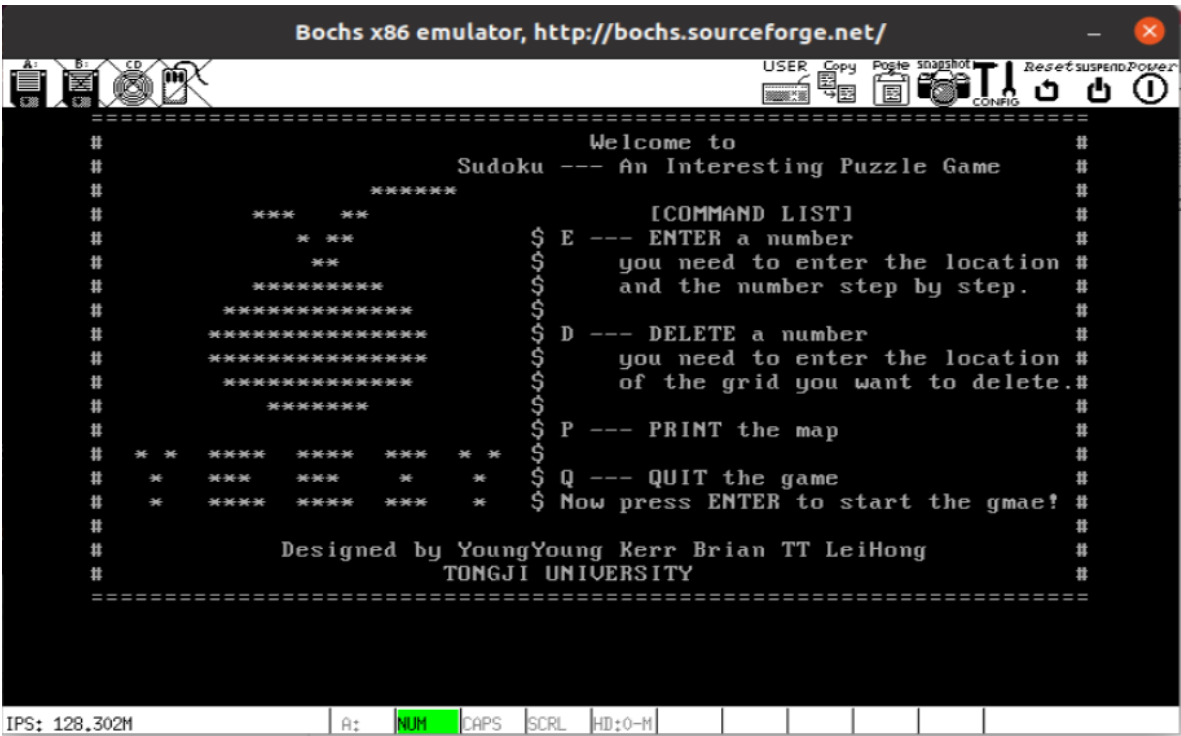
用户输入指令"game", 进入游戏界面, 控制台将显示所有游戏简介及相应指令。



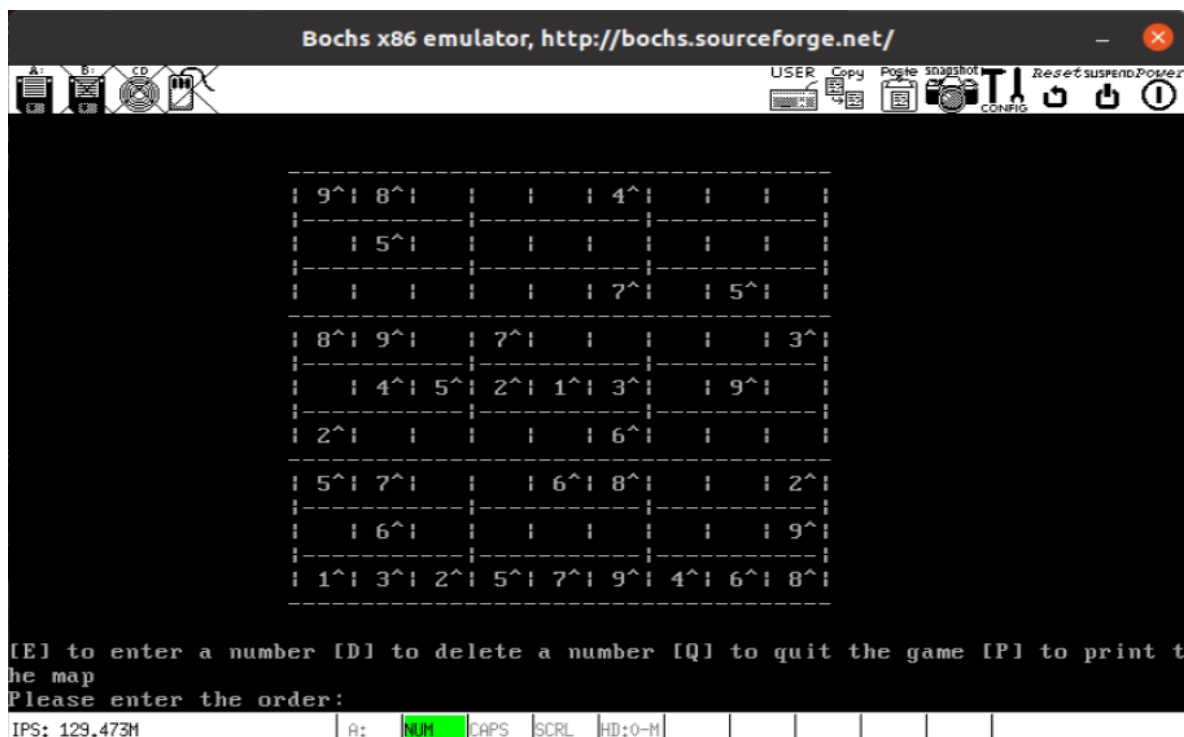
2.2.6.2 sudoku (数独)

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



- 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



- 用户成功完成数独时，将显示游戏获胜信息

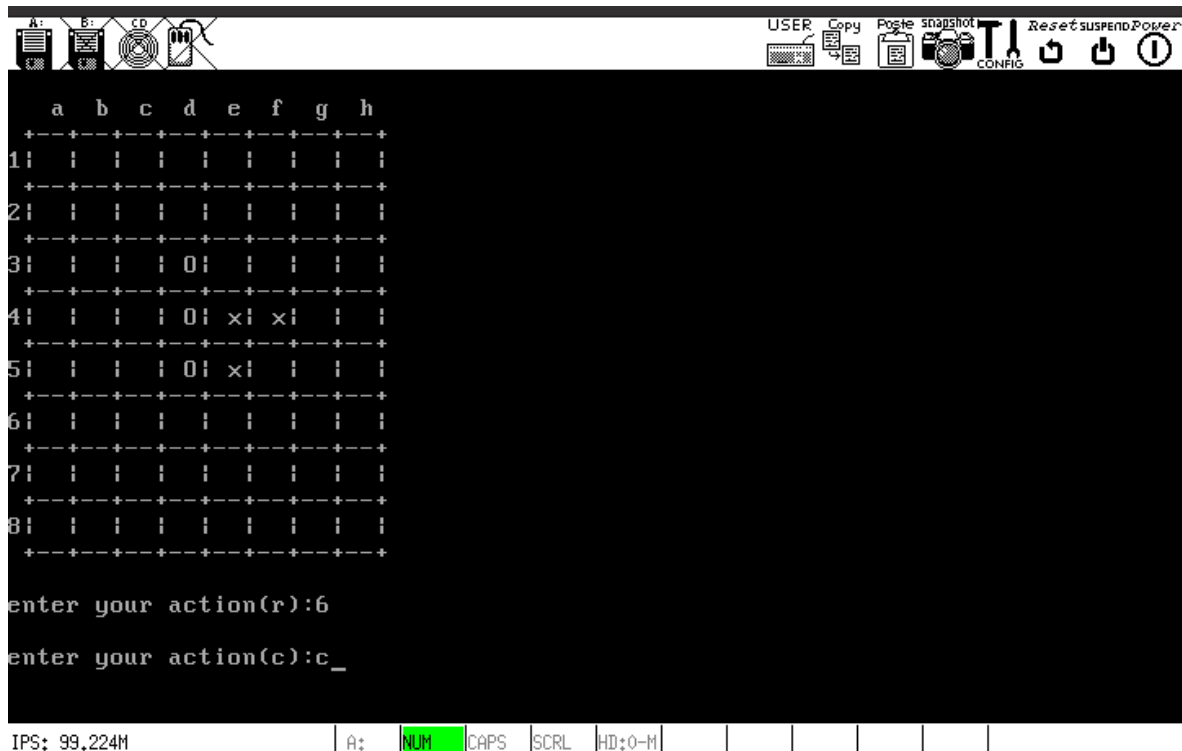
功能实现

- 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



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

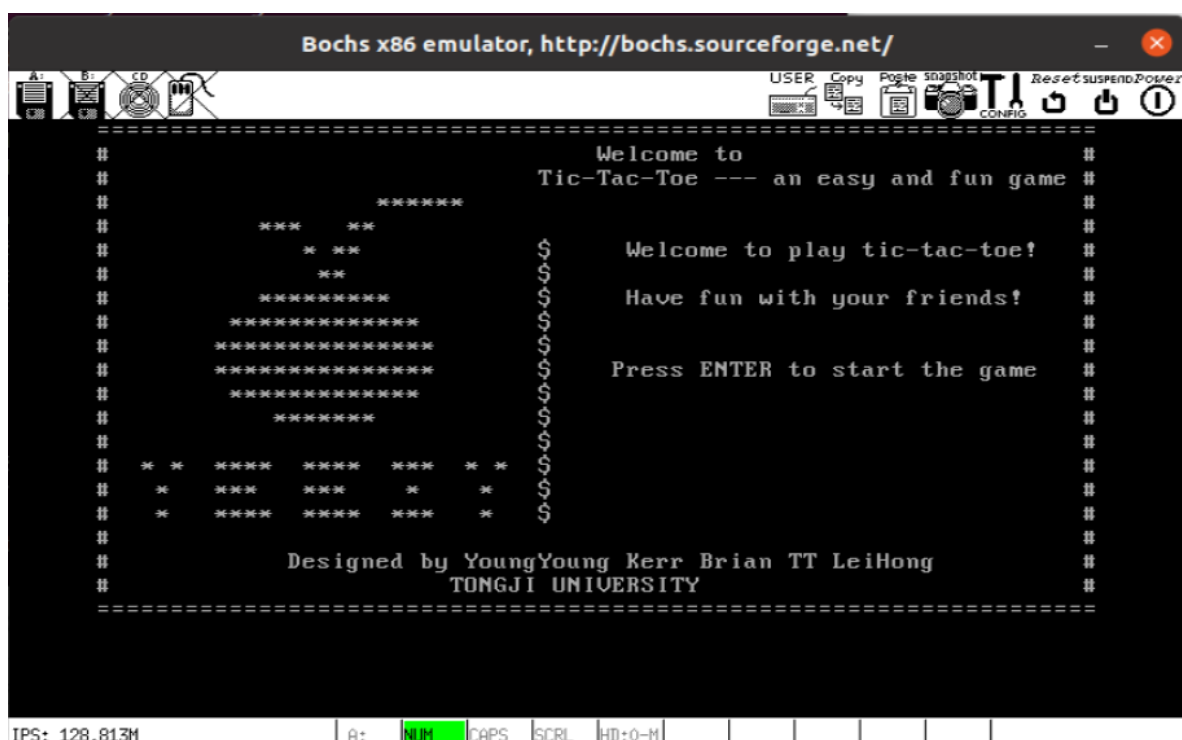
功能实现

- 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

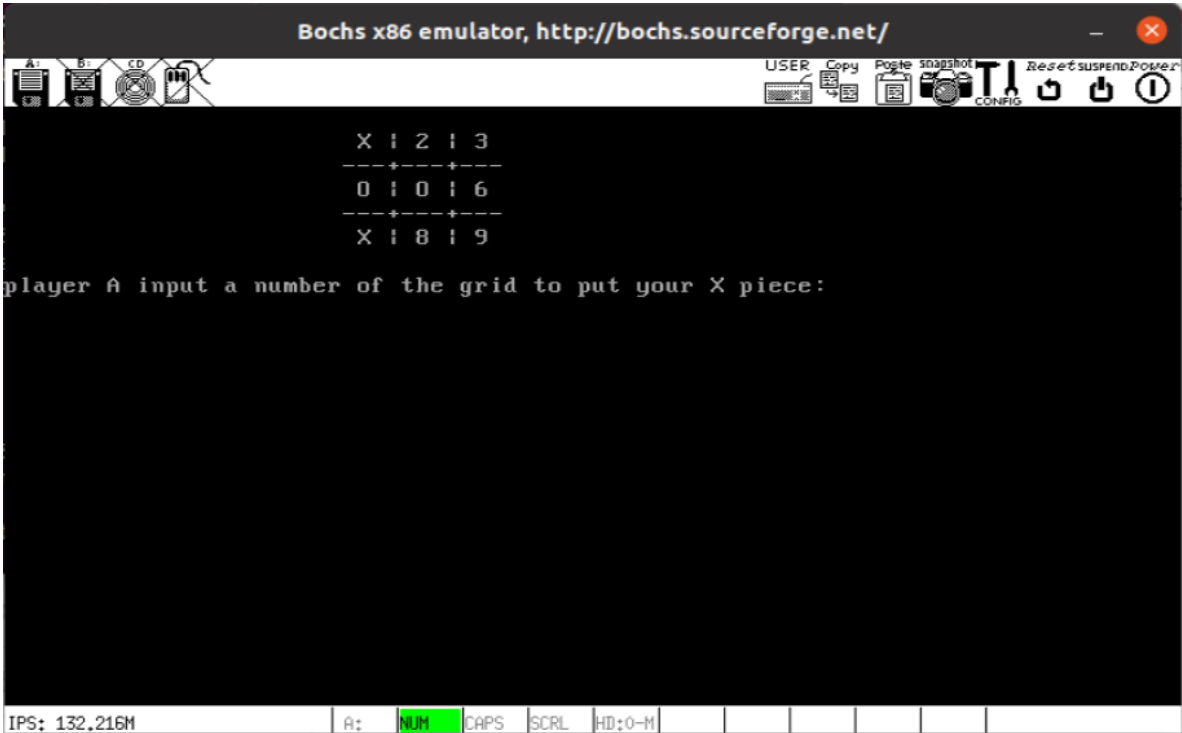
2.2.6.4 tic-tac-toe (井字棋)

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



- 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）



- 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



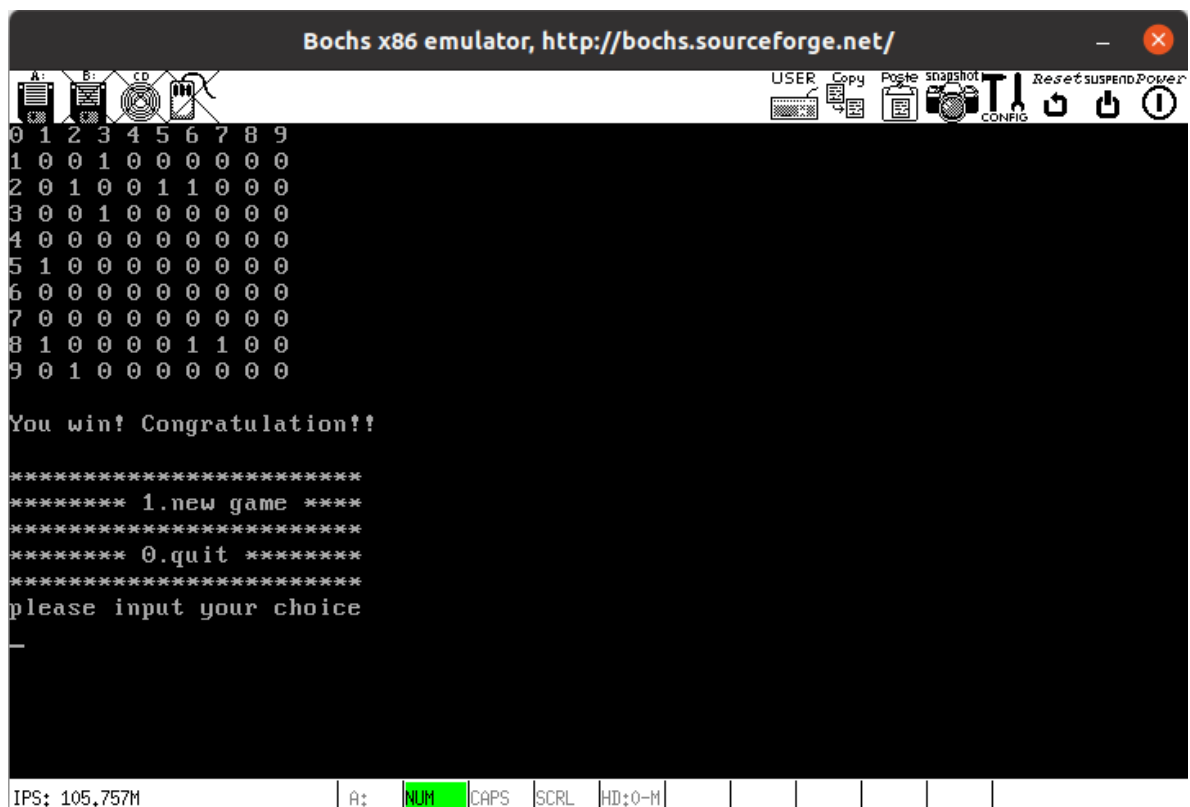
功能实现

- 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



- 用户“踩”到雷则游戏失败



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

```

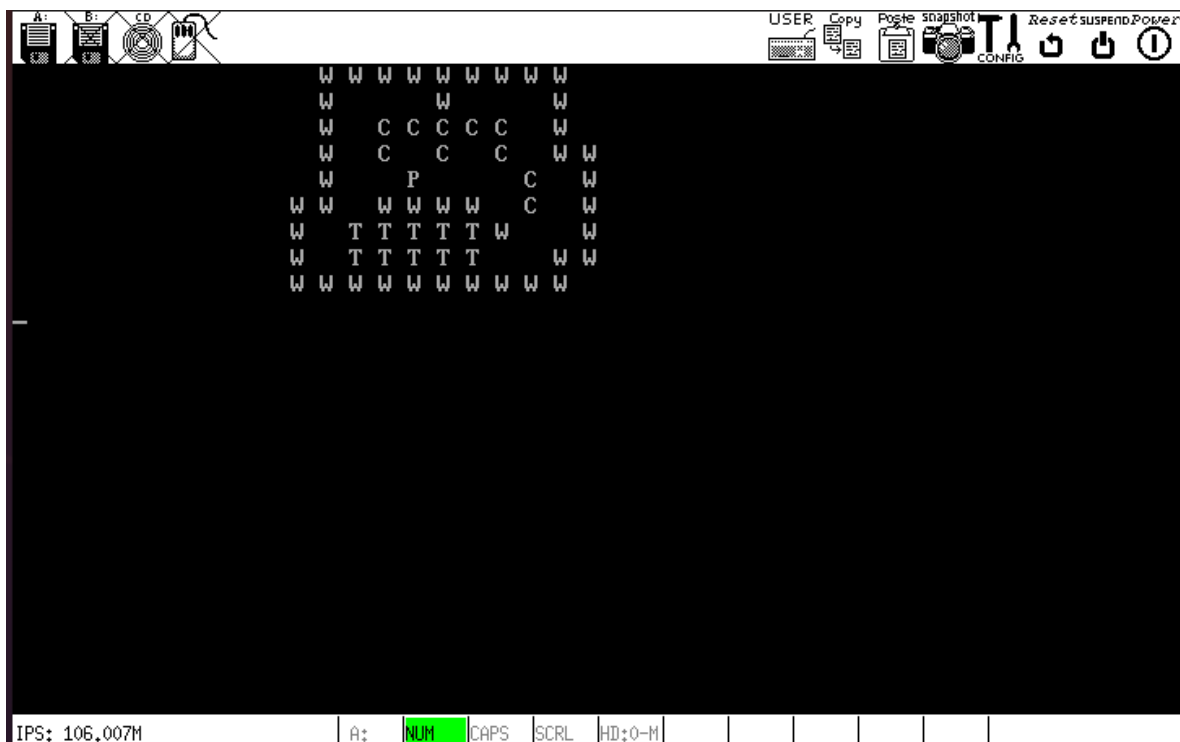
A: B: CD
master@yeezyOS: $ carry
=====
#                               Welcome to                               #
#                               Carrycraft --- Carry the craft to the target! #
#                               *****                                #
#                               [ COMMAND ]                             #
#                               $ In the map, 'W' means the wall        #
#                               $ 'P' means the player                 #
#                               $ 'C' means the craft                  #
#                               $ 'T' means the target                 #
#                               $ 'G' means the craft at target         #
#                               $ ENTER w to go up                     #
#                               $ ENTER s to go down                   #
#                               $ ENTER a to go left                   #
#                               $ ENTER d to go right                  #
#                               $ Now press ENTER to start the game!   #
#                               #                                       #
#                               Designed by YoungYoung Kerr Brian TT LeiHong #
#                               TONGJI UNIVERSITY                       #
#                               #                                       #
=====
IPS: 105.228M  A: NUM CAPS SCRL HD:0-M
```

- 玩家通过wasd输入来实现控制人物进行上下左右的行走

```

A: B: CD
W W W W W W W W
W           W
W   C C C C   W
W   C   C   C   W W
W           P   C   W
W W   W W W W   C   W
W   T T T T T W   W
W   T T T T T   W W
W W W W W W W W
a_
IPS: 105.297M  A: NUM CAPS SCRL HD:0-M
```

- 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



- 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

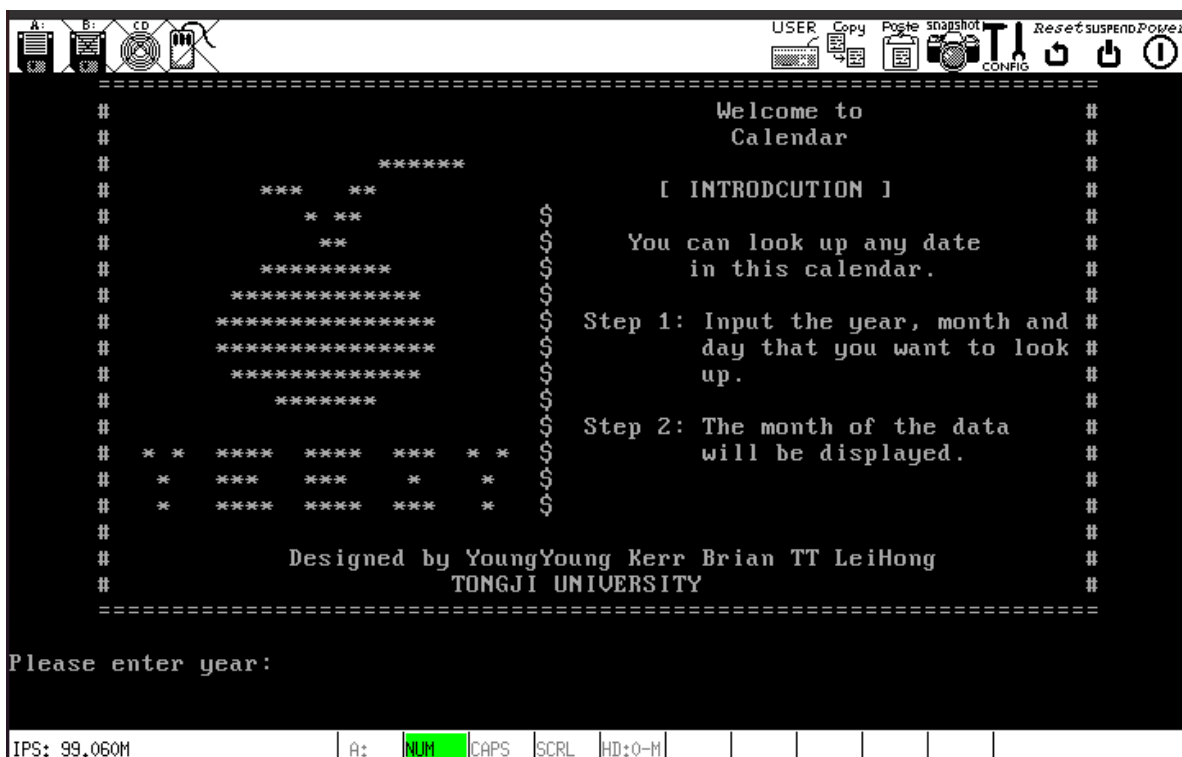
功能实现

- 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



- 用户输入要查询的年月日

```
=====
#                               Welcome to                               #
#                               Calendar                               #
#                               [ INTRODCUTION ]                       #
#                               You can look up any date              #
#                               in this calendar.                      #
#                               Step 1: Input the year, month and      #
#                               day that you want to look up.         #
#                               Step 2: The month of the data         #
#                               will be displayed.                   #
#                               Designed by YoungYoung Kerr Brian TT LeiHong
#                               TONGJI UNIVERSITY
#                               =====
Please enter year:2020
Please enter month:8
Please enter day:22
IPS: 102.549M | A: NUM | CAPS | SCRL | HD:0-M |
```

- 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

```
=====
#                               2020-8-22 is Saturday                #
#                               2020 - August                        #
#                               S    M    T    W    T    F    S      #
#                               1                                           #
#                               2    3    4    5    6    7    8      #
#                               9    10   11   12   13   14   15      #
#                               16   17   18   19   20   21   22      #
#                               23   24   25   26   27   28   29      #
#                               30   31                                     #
#                               Press ENTER to continue...             #
=====
IPS: 102.362M | A: NUM | CAPS | SCRL | HD:0-M |
```

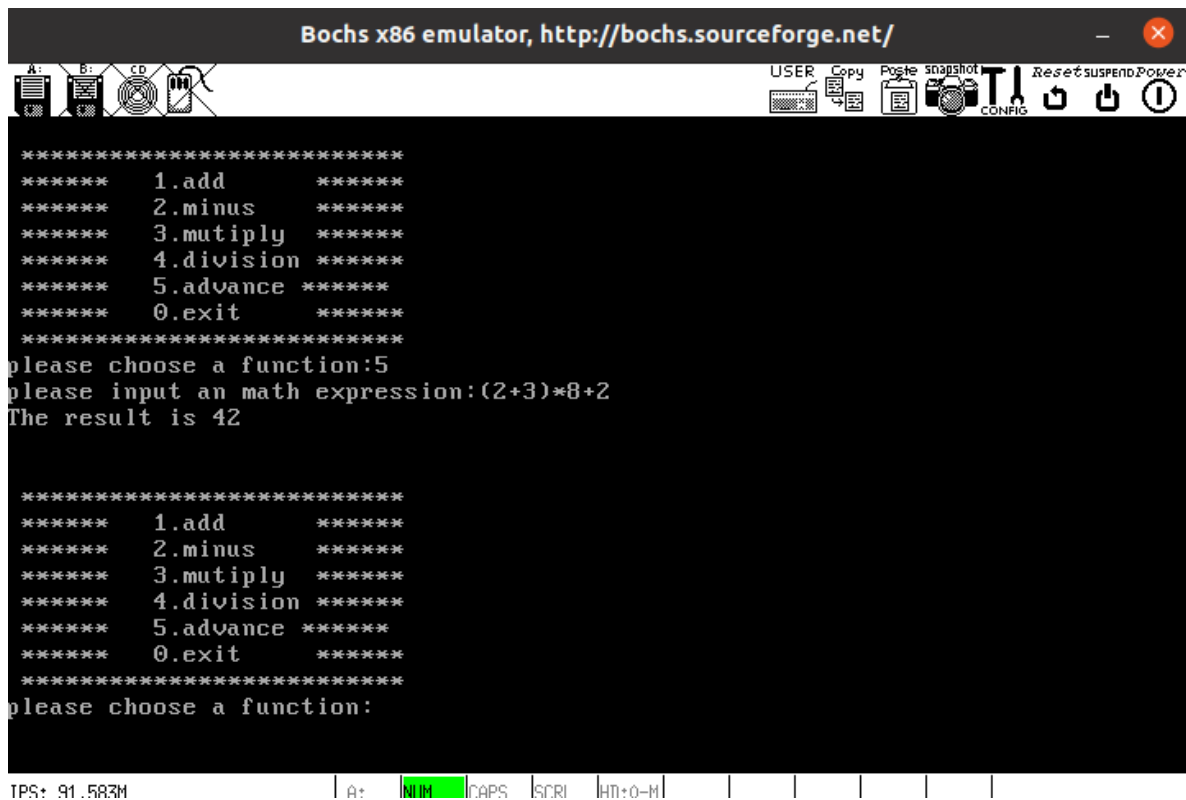
功能实现

- 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



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

3 项目核心代码

3.1 进程管理

processManage()

```

1  void processManage()
2  {
3      int i;
4      printf("=====
=====\\n\\n");
5      printf("                                Process Manager
                                \\n\\n");
6      printf("=====
=====\\n");
7      printf("%9s      |%10s      |%12s      |%10s  \\n", "PID", "name", "priori
ty", "running?");
8
9      printf("-----
-----\\n");
10     for (i = 0; i < NR_TASKS + NR_PROCS; ++i)
11     {
12         /* 遍历进程列表，输出每个进程的pid，名字，优先级和运行状态 */
13         if (proc_table[i].p_flags != FREE_SLOT)
14             printf("%8d      |%10s      |%10d      |%s\\n", proc_table
[i].pid, proc_table[i].name, proc_table[i].priority, proc_table[i].p_flags
== FREE_SLOT ? "NO" : "YES");

```

```

15     }
16     printf("=====
=====\\n\\n");
17 }

```

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()

```

1  PUBLIC int strip_path(char * filename, const char * pathname,
2                      struct inode** ppinode)
3  {
4      const char * s = pathname;
5      char * t = filename;
6      struct inode * dir_inode = get_inode(ROOT_DEV, ROOT_INODE);
7      struct inode * next_inode;
8
9      if (s == 0)
10         return -1;
11     if (*s == '/' || *s == '$' || *s == '#')
12         s++;
13     if (*s == '/')
14         s++;
15     while (*s) /* check each character */
16     {
17         if (*s == '/')
18         {
19             *t = 0;
20             int dev = dir_inode->i_dev;
21             int dir_inode_nr = get_file(filename, dir_inode, I_DIRECTORY);
22             next_inode = get_inode(dev, dir_inode_nr);
23             put_inode(dir_inode);
24             dir_inode = next_inode;
25             t = filename;
26             s++;
27         }
28         *t++ = *s++;
29         /* if filename is too long, just truncate it */
30         if (t - filename >= MAX_FILENAME_LEN)
31             break;
32     }
33     *t = 0;
34     *ppinode = dir_inode;
35     return 0;
36 }

```

get_file()

```

1  PUBLIC int get_file(char* filename, struct inode* dir_inode,
2                      u32 mode)
3  {

```



```

4   if (filename[0] == 0) /* path: "/" */
5       return dir_inode->i_num;
6   * Search the dir for the file.
7   */
8   int dir_blk0_nr = dir_inode->i_start_sect;
9   int nr_dir_blks = (dir_inode->i_size + SECTOR_SIZE - 1) / SECTOR_SIZE;
10  int nr_dir_entries =
11      dir_inode->i_size / DIR_ENTRY_SIZE; /**
12                                          * including unused slots
13                                          * (the file has been deleted
14                                          * but the slot is still there)
15                                          */
16  int i, j, m = 0;
17  struct dir_entry * pde;
18  int dev = dir_inode->i_dev;
19  for (i = 0; i < nr_dir_blks; i++)
20  {
21      RD_SECT(dev, dir_blk0_nr + i);
22      pde = (struct dir_entry *)fsbuf;
23      for (j = 0; j < SECTOR_SIZE / DIR_ENTRY_SIZE; j++, pde++)
24      {
25          if (strcmp(filename, pde->name) == 0)
26          {
27              int result = pde->inode_nr;
28              struct inode * pin = get_inode(dev, pde->inode_nr);
29              if (pin->i_mode == mode || mode == 0)
30              {
31                  put_inode(pin);
32                  return result;
33              }
34              put_inode(pin);
35              RD_SECT(dev, dir_blk0_nr + i);
36          }
37
38          if (++m > nr_dir_entries)
39              break;
40      }
41      if (m > nr_dir_entries) /* all entries have been iterated */
42          break;
43  }
44  /* file not found */
45  return 0;
46  }

```

3.2.1.2 文件的查找与创建

search_file()

```

1  PUBLIC int search_file(char * path)
2  {
3      char * _path = path;
4      if (path[0] == '$' || path[0] == '#') _path++;
5      char filename[MAX_PATH];
6      memset(filename, 0, MAX_FILENAME_LEN);
7      struct inode * dir_inode;
8      if (strip_path(filename, _path, &dir_inode) != 0)
9          return 0;

```

```

10     if (path[0] == '$')
11     {
12         return get_file(filename, dir_inode, I_DIRECTORY);
13     }
14     if (path[0] == '#')
15     {
16         return get_file(filename, dir_inode, I_REGULAR);
17     }
18     return get_file(filename, dir_inode, 0);
19 }

```

create_file()

```

1 PRIVATE struct inode * create_file(char * path, int flags)
2 {
3     u32 mode = I_REGULAR;
4     if (path[0] == '$') mode = I_DIRECTORY;
5     char filename[MAX_PATH];
6     struct inode * dir_inode;
7     if (strip_path(filename, path, &dir_inode) != 0)
8         return 0;
9     int inode_nr = alloc_imap_bit(dir_inode->i_dev);
10    int free_sect_nr = alloc_smap_bit(dir_inode->i_dev,
11                                     NR_DEFAULT_FILE_SECTS);
12    struct inode *newino = new_inode(dir_inode->i_dev, inode_nr,
13                                     free_sect_nr, mode);
14    new_dir_entry(dir_inode, newino->i_num, filename);
15    return newino;
16 }

```

3.2.2 文件系统用户功能

3.2.2.1 写文件

```

1 void wt(char * cur_dir, char * filename)
2 {
3     char path[32] = "#";
4     int i = 0;
5     for (; ; i++)
6     {
7         if (cur_dir[i] == 0) break;
8         path[i + 1] = cur_dir[i];
9     }
10    i++;
11    for (int j = 0; ; j++)
12    {
13        if (filename[j] == 0) break;
14        path[i++] = filename[j];
15    }
16    path[i] = 0;
17    int fd = open (path, O_RDWR);
18    if (fd == -1)
19    {
20        printf("Failed to open %s!\n", filename);
21        return;
22    }

```

```

23     char tty_name[] = "/dev_tty0";
24     int fd_stdin = open(tty_name, O_RDWR);
25     if (fd_stdin == -1)
26     {
27         printf("Failed to write file!\n");
28         return;
29     }
30     char writeBuf[4096]; // 写缓冲区
31     int final = read(fd_stdin, writeBuf, 4096);
32     writeBuf[final] = 0;
33     write(fd, writeBuf, final + 1);
34     printf("Totally write %d bytes in the file : %s\n", final, writeBuf);
35     close(fd);
36 }

```

3.2.2.2读文件

```

1 void rd(char*cur_dir, char * filename){
2     char path[32] = "#";
3     int i = 0;
4     for (; ; i++)
5     {
6         if (cur_dir[i] == 0) break;
7         path[i + 1] = cur_dir[i];
8     }
9     i++;
10    for (int j = 0; ; j++)
11    {
12        if (filename[j] == 0) break;
13        path[i++] = filename[j];
14    }
15    path[i] = 0;
16    int fd = open (path, O_RDWR);
17    if(fd==-1){
18        printf("Failed to open %s!\n", filename);
19        return;
20    }
21    char buf[4096];
22    int rdFlag = read(fd, buf, 4096);
23    if (rdFlag == -1) // 读取文件内容失败
24    {
25        printf("Failed to read file!\n");
26        close(fd);
27        return;
28    }
29
30    printf("Content of file %s :\n", filename);
31    printf(" %s\n", buf);
32    close(fd);
33 }

```

3.2.2.3创建文件

```

1 void touch(char * cur_dir, char * filename)
2 {
3     if (filename[0] <= 0)

```

```

4      {
5          printf("touch:: error: command touch need one parameter.\n");
6          return;
7      }
8      if (filename[0] == ' ')
9      {
10         printf("touch:: error: file name cannot start with space.\n");
11         return;
12     }
13     char path[32] = "#";    //普通目录标识符
14     int i = 0;
15     for (; ; i++)
16     {
17         if (cur_dir[i] == 0) break;
18         path[i + 1] = cur_dir[i];
19     }
20     i++;
21     for (int j = 0; ; j++)
22     {
23         if (filename[j] == 0) break;
24         path[i++] = filename[j];
25     }
26     path[i] = 0;
27     int fd = open(path, O_CREAT);
28     if (fd != -1)
29     {
30         close(fd);
31     }
32     else
33     {
34         printf("touch error: file exists.\n");
35     }
36 }

```

3.2.2.4展示文件列表

```

1  PUBLIC int do_ls()
2  {
3      char pathname[MAX_PATH];
4      /* get parameters from the message */
5      int name_len = fs_msg.NAME_LEN; /* length of filename */
6      int src = fs_msg.source; /* caller proc nr. */
7      assert(name_len < MAX_PATH);
8      phys_copy((void*)va2la(TASK_FS, pathname),
9               (void*)va2la(src, fs_msg.PATHNAME),
10               name_len);
11     pathname[name_len] = 0;
12     char * filename;
13     struct inode * dir_inode;
14     strip_path(filename, pathname, &dir_inode);
15     return get_list(dir_inode, I_DIRECTORY) + get_list(dir_inode, I_REGULAR
16 );
17 }

```

3.2.2.5切换文件目录

```

1 void cd(char * cur_dir, char * dir_name)
2 {
3     //处理特殊目录
4     if (dir_name[0] <= 0)
5     {
6         printf("cd:: error: command cd need one parameter.\n");
7         return;
8     }
9     if (dir_name[0] == ' ')
10    {
11        printf("cd:: error: directory cannot start with space.\n");
12        return;
13    }
14    if (dir_name[0] == '.')
15    {
16        if (dir_name[1] == '.')
17        {
18            int i = 0;
19            for (; ; i++)
20            {
21                if (cur_dir[i] == '/' && cur_dir[i + 1] == 0)
22                {
23                    break;
24                }
25            }
26            if (i > 0)
27            {
28                cur_dir[i--] = 0;
29                for (; ; i--)
30                {
31                    if (cur_dir[i] == '/') break;
32                    cur_dir[i] = 0;
33                }
34            }
35        }
36        return;
37    }
38    char path[32] = "$";
39    int i = 0;
40    for (; ; i++)
41    {
42        if (cur_dir[i] == 0) break;
43        path[i + 1] = cur_dir[i];
44    }
45    i++;
46    for (int j = 0; ; j++)
47    {
48        if (dir_name[j] == 0) break;
49        path[i++] = dir_name[j];
50    }
51    path[i] = 0;
52    int fd = open(path, O_RDWR);
53    if (fd == -1)
54    {
55        printf("No such directory.\n");
56        return;
57    }
58    close(fd);

```

```

59     strcat(cur_dir, dir_name);
60     strcat(cur_dir, "/");
61 }

```

3.2.2.6 删除文件

```

1  void rm(char * cur_dir, char * filename, int flag)
2  {
3      char path[32] = "#";
4
5      if (flag == 1)
6      {
7          path[0] = '$';
8      }
9      int i = 0;
10     for (; ; i++)
11     {
12         if (cur_dir[i] == 0) break;
13         path[i + 1] = cur_dir[i];
14     }
15     i++;
16     for (int j = 0; ; j++)
17     {
18         if (filename[j] == 0) break;
19         path[i++] = filename[j];
20     }
21     path[i] = 0;
22     int fd = unlink(path);
23     if (fd == 0)
24     {
25         printf("%s deleted!\n", filename);
26     }
27     else
28     {
29         printf("Failed to delete %s!\n", filename);
30     }
31 }

```

3.3 游戏及工具

3.3.1 游戏

3.3.1.1 sudoku (数独)

get_map()

```

1  /* 搜索算法建立数独棋盘 */
2  PUBLIC void dfs(int i, int j, bool *success)
3  {
4      if (*success)
5      {
6          return;
7      }
8      if (i == 9 && j == 0)
9      {

```

```

10     /* 成功建立棋盘 */
11     *success = true;
12     copy_map();
13     return;
14 }
15 int next_i = j == 8 ? i + 1 : i;
16 int next_j = j == 8 ? 0 : j + 1;
17 /**
18  * 建立1-9的随机数组进行遍历
19  *
20  * but bochs中导入time.h有点问题
21  */
22 for (int k = 9; k > 0; k--)
23 {
24     map_copy[i][j] = k;
25     if (is_legal(i, j))
26     {
27         dfs(next_i, next_j, success);
28     }
29     map_copy[i][j] = 0;
30 }
31
32 }
33 /* 获取数独棋盘 */
34 PUBLIC void get_map()
35 {
36     bool success = false;
37     dfs(0, 0, &success);
38 }

```

sudoku_main()

```

1 PUBLIC void sudoku_main(int *fd_stdin)
2 {
3     /* init the game */
4     get_map();
5     char order[2];          // 指令
6     char r[2], c[2], v[2]; // row,column,value
7     /* press ENTER to start the game */
8     sudoku_list();
9     int start = read(*fd_stdin, order, 512);
10    print_map();
11    while (1)
12    {
13        printf("
14        [E] to enter a number [D] to delete a number [Q] to quit the game [P] to pr
15        int the map\n");
16        printf("Please enter the order: ");
17        /* read */
18        int p = read(*fd_stdin, order, 512);
19        order[p] = 0;
20        if (strcmp(order, "E") == 0)
21        {
22            /* 输入数字 */
23            int row, col, val;
24            printf("Please enter the row of the grid you want to complete:
25            ");

```

```

23     p = read(*fd_stdin, r, 512);
24     r[p] = 0;
25     row = r[0] - '0';
26     printf("Please enter the column of the grid you want to complet
e: ");
27     p = read(*fd_stdin, c, 512);
28     c[p] = 0;
29     col = c[0] - '0';
30     if (status[row - 1][col - 1] != EMPTY)
31     {
32         printf("Sorry, grid[%d, %d] has been filled in.\n", row, co
l);
33         print_map();
34         continue;
35     }
36     printf("Please enter the number you want to fill in: ");
37     p = read(*fd_stdin, v, 512);
38     v[p] = 0;
39     val = v[0] - '0';
40     if (val == sudoku_map[row - 1][col - 1])
41     {
42         /* 输入正确 */
43         status[row - 1][col - 1] = CORRECT;
44     }
45     else
46     {
47         /* 输入错误 */
48         status[row - 1][col - 1] = WRONG;
49     }
50     player[row - 1][col - 1] = val;
51 }
52 else if (strcmp(order, "D") == 0)
53 {
54     /* 删除数字 */
55     int row, col;
56     printf("Please enter the row of the grid you want to delete: ")
;
57     p = read(*fd_stdin, r, 512);
58     r[p] = 0;
59     row = r[0] - '0';
60     printf("Please enter the column of the grid you want to delete:
");
61     p = read(*fd_stdin, c, 512);
62     c[p] = 0;
63     col = c[0] - '0';
64     if (status[row - 1][col - 1] == GIVEN)
65     {
66         /* 初始给定数字不能删除 */
67         printf("Sorry, Grid[%d, %d] cannot be deleted.\n\n", row, c
ol);
68         continue;
69     }
70     else
71     {
72         status[row - 1][col - 1] = EMPTY;
73         player[row - 1][col - 1] = 0;
74     }
75 }

```



```

76         else if (strcmp(order, "Q") == 0)
77         {
78             printf("\n\n\n");
79             break;
80         }
81         else if (strcmp(order, "P") != 0)
82         {
83             printf("False Command!\n\n");
84             continue;
85         }
86         printf("\n\n\n\n\n\n\n\n\n");
87         print_map();
88         /* 判定是否获胜 */
89         if (check_win())
90         {
91             printf("                Congratulation!! You successfully complete
the sudoku!!\n");
92             break;
93         }
94     }
95 }

```

3.3.1.2 bwchess (黑白棋)

Hint()

```

1  int Hint(char board[][MAXSIZE], int arrput[][MAXSIZE], char level) //最佳走法
2  {
3      int row, col, i, j;
4      char board1[MAXSIZE][MAXSIZE] = { 0 };
5      int maxscore = 0;
6      int score = 0;
7      char foe;
8      if (level == 1)
9          foe = -1;
10     else
11         foe = 1;
12     for (row = 0; row < MAXSIZE; row++)
13     for (col = 0; col < MAXSIZE; col++)
14     {
15         if (!arrput[row][col])
16             continue;
17         for (i = 0; i < MAXSIZE; i++)
18         for (j = 0; j < MAXSIZE; j++)
19         {
20             board1[i][j] = board[i][j];
21         }
22         Print(board1, row, col, level);
23         score = CalSore(board1, level);
24         if (maxscore < score)
25             maxscore = score;
26     }
27     return maxscore;
28 }

```

CalSore()

```

1  int calSore(char board[][MAXSIZE], char level) //计算成绩
2  {
3  int score = 0;
4  int row, col;
5  char foe;
6  if (level == 1)
7  foe = -1;
8  else
9  foe = 1;
10 char player = -1 * foe;
11 for (row = 0; row < MAXSIZE; row++)
12 for (col = 0; col < MAXSIZE; col++)
13 {
14 score = score - (board[row][col] == foe);
15 score = score + (board[row][col] == player);
16 }
17 return score;
18 }

```

bwchess_main()

```

1  for (i = count; count < (MAXSIZE * MAXSIZE) && cross < 2;)
2  {
3      if (level == 1)
4      {
5          level = 0;
6          if (Exa(board, arrput, 2))
7          {
8              while (1)
9              {
10                 printf("\nenter your action(r):");
11                 p = read(*fd_stdin, r, 512);
12                 r[p] = 0;
13                 if (!strcmp(r, "Q"))
14                 {
15                     return;
16                 }
17                 x = r[0] - '0';
18
19                 printf("\nenter your action(c):");
20                 p = read(*fd_stdin, c, 512);
21                 c[p] = 0;
22                 y = c[0];
23                 if (!strcmp(c, "Q"))
24                 {
25                     return;
26                 }
27                 //y = input[1];
28                 x--;
29                 if (y >= 'a')
30                 {
31                     y = y - 'a' + 1;
32                 }
33                 else
34                 {
35                     y = y - 'A' + 1;
36                 }

```

```

37         y--;
38         if (x >= 0 && y >= 0 && x < MAXSIZE && y < MAXSIZE
&& arrput[x][y])
39             {
40                 Print(board, x, y, 2);
41                 count++;
42                 break;
43             }
44             else
45             {
46                 printf("\nwrong order try again .\n");
47             }
48         }
49         Show(board); //更新棋盘
50     }
51     else if (++cross < 2)
52     {
53         printf("\n is my turn.\n");
54     }
55     else
56     {
57         printf("\ngame over.\n");
58     }
59 }
60 else
61 {
62     level = 1;
63     if (Exa(board, arrput, 1))
64     {
65         cross = 0;
66         Foeplay(board, arrput, 1);
67         count++;
68         Show(board);
69     }
70     else
71     {
72         if (++cross < 2)
73         {
74             printf("\ni cannot move,you turn\n");
75         }
76         else
77         {
78             printf("\ngame over.");
79         }
80     }
81 }
82 }
83 Show(board);
84 score[0] = score[1] = 0;
85 for (row = 0; row < MAXSIZE; row++) //计算分数
86 {
87     for (col = 0; col < MAXSIZE; col++)
88     {
89         score[0] = score[0] + (board[row][col] == -1);
90         score[1] = score[1] + (board[row][col] == 1);
91     }
92 }
93 printf("final score:\n");

```

```

94     printf("white:%d\nblack:%d\n", score[1], score[0]);
95     if (score[1] > score[0])
96     {
97         printf("\nha ha ha ,you lose \n\n");
98     }
99     else
100    {
101        printf("\noh,i will be back!\n\n");
102    }
103 }
104 //scanf_s("%c", &ok);
105 }

```

3.3.1.3 tic-tac-toe (井字棋)

ticTacToe_main()

```

1  PUBLIC int ticTacToe_main(int *fd_stdin)
2  {
3      int player = 0;
4      int winner = 0;
5      int number = 0;
6      int row = 0;
7      int column = 0;
8      char tic_tac_toe[3][3] = {
9          {'1', '2', '3'},
10         {'4', '5', '6'},
11         {'7', '8', '9'}};
12
13     //让双方玩家轮流输入自己想要标志的位置
14     int i;
15     char o[2];
16     ticTacToe_list();
17     read(*fd_stdin, o, 512);
18     clear();
19     for (i = 0; i < 9 && winner == 0; i++)
20     {
21         printf("\n");
22         printf("                ");
23         printf(" %c | %c | %c \n", tic_tac_toe[0][0], tic_tac_toe[0]
[1], tic_tac_toe[0][2]);
24         printf("                ");
25         printf("----+----+----\n");
26         printf("                ");
27         printf(" %c | %c | %c \n", tic_tac_toe[1][0], tic_tac_toe[1]
[1], tic_tac_toe[1][2]);
28         printf("                ");
29         printf("----+----+----\n");
30         printf("                ");
31         printf(" %c | %c | %c \n", tic_tac_toe[2][0], tic_tac_toe[2]
[1], tic_tac_toe[2][2]);
32         player = i % 2 + 1;
33         do
34         {
35             printf("\nif you want to exit, please input 0.\n");
36             printf("\nplayer %c", (player == 1) ? 'A' : 'B');

```

```

37         printf(" input a number of the grid to put your %c piece:", (player == 1) ? 'x' : 'o');
38         //scanf_s("%d", &number);
39         int p = read(*fd_stdin, o, 512);
40         o[p] = 0;
41         number = o[0] - '0';
42         if(number==0) return 0;
43         row = (number - 1) / 3;    //行的索引码
44         column = (number - 1) % 3; //列的索引码
45     } while (number < 0 || number > 9 || tic_tac_toe[row]
[0] == tic_tac_toe[1][0] || tic_tac_toe[1][0] == tic_tac_toe[2][0] ||
46         tic_tac_toe[row][column] = (player == 1) ? 'x' : 'o';
47         //检查此玩家是否获胜
48         //检查对角线上该玩家是否获胜
49         if ((tic_tac_toe[0][0] == tic_tac_toe[1][1] && tic_tac_toe[0]
[0] == tic_tac_toe[2][2]) ||
50             (tic_tac_toe[0][2] == tic_tac_toe[1][1] && tic_tac_toe[0]
[2] == tic_tac_toe[2][0]))
51         {
52             winner = player;
53         }
54         //检查横或竖上该玩家是否获胜
55         else
56         {
57             for (int line = 0; line < 3; line++)
58             {
59                 if ((tic_tac_toe[line][0] == tic_tac_toe[line]
[1] && tic_tac_toe[line][0] == tic_tac_toe[line][2]) ||
60                     (tic_tac_toe[0][line] == tic_tac_toe[1]
[line] && tic_tac_toe[0][line] == tic_tac_toe[2][line]))
61                 {
62                     winner = player;
63                 }
64             }
65         }
66         clear();
67     }
68     //公布最后得分面板
69     printf("\n");
70     printf("                ");
71     printf(" %c | %c | %c \n", tic_tac_toe[0][0], tic_tac_toe[0]
[1], tic_tac_toe[0][2]);
72     printf("                ");
73     printf("----+----+---\n");
74     printf("                ");
75     printf(" %c | %c | %c \n", tic_tac_toe[1][0], tic_tac_toe[1]
[1], tic_tac_toe[1][2]);
76     printf("                ");
77     printf("----+----+---\n");
78     printf("                ");
79     printf(" %c | %c | %c \n", tic_tac_toe[2][0], tic_tac_toe[2]
[1], tic_tac_toe[2][2]);
80     //打印最后胜利者结果
81     if (winner == 0)
82     {
83         printf("\nwhat a pity! Nobody win the game!\n");
84     }
85     else

```

```

86     {
87         printf("\nCongratulation! The winner is %c!\n", (winner == 1) ? 'A'
: 'B');
88     }
89     printf("\n\nPress ENTER to continue...\n");
90     read(*fd_stdin, o, 512);
91     return 0;
92 }

```

3.3.1.4mine (扫雷)

void InitBoard()

```

1  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
2  {
3      int i = 0;
4      int j = 0;
5      for (i = 0; i < rows; i++)
6      {
7          for (j = 0; j < cols; j++)
8          {
9              board[i][j] = set;//set可以更改
10         }
11     }
12 }

```

void FindMine()

```

1  void FindMine(char mine[ROWS][COLS], char show[ROWS][COLS], int row, int
   col, int fd_stdin)
2  {
3      int x = 0;
4      int y = 0;
5      int win = 0;
6      int p;
7      char xn[2];
8      char yn[2];
9      printf("\nCurrent score is %d.\n", win);
10     printf("You'll win when score reaches 71!\n\n");
11     while (win < (row * col - EASYCOUNT))
12     {
13         printf("please input x:>");
14         p = read(fd_stdin, xn, 512);
15         printf("please input y:>");
16         x=xn[0]-'0';
17         p = read(fd_stdin, yn, 512);
18         y=yn[0]-'0';
19         if (x >= 1 && x <= row && y >= 1 && y <= col)
20         {
21             if (mine[x][y] == '1')
22             {
23                 clear();
24                 printf("Boom!!! Game over!\n");
25                 DispalyBoard(mine, row, col);
26                 break;

```

```

27         }
28         else
29         {
30             ShowMineCount(mine, show, row, col, x, y, &win);
31             clear();
32             DispalyBoard(show, row, col);
33             printf("\nCurrent score is %d.\n", win);
34             printf("You'll win when score reaches 71!\n\n");
35         }
36         if (win == row * col - EASYCOUNT)
37         {
38             clear();
39             DispalyBoard(mine, row, col);
40             printf("\nYou win! Congratulation!!\n\n");
41         }
42     }
43     else
44     {
45         printf("wrong index:>\n");
46     }
47 }
48 }

```

3.3.1.5 carrycraft (推箱子)

drawmain()

```

1  int drawmain()
2  {
3      clear();
4      int i, j;
5      win(); //判断输赢
6      for (i = 0; i < 9; i++)
7      {
8          printf("                ");
9          for (j = 0; j < 11; j++)
10         {
11             switch (map[i][j])
12             {
13                 case 0:
14                     printf(" "); //空白的位置
15                     break;
16                 case 1:
17                     printf("W "); //墙
18                     break;
19                 case 2:
20                     printf("P "); //代表人
21                     break;
22                 case 3:
23                     printf("C "); //代表箱子
24                     break;
25                 case 4:
26                     printf("T "); //代表目标地址
27                     break;
28                 case 6:
29                     printf("P "); //代表人和目标地址重合
30                     break;

```

```

31 case 7:
32 printf("G "); //代表箱子和目标地址重合
33 break;
34 }
35 }
36 printf("\n");
37 }
38 return 0;
39 }

```

carry_main()

```

1 PUBLIC int carrycraft_main(int* fd_stdin)
2 {
3     char o[2];
4     craft_list();
5     read(*fd_stdin, o, 512);
6     while (1)
7     {
8         drawmain();
9         int p = read(*fd_stdin, o, 512);
10        o[p] = 0;
11        char push = o[0];
12        int count, caw;
13        for (int i = 0; i < 9; i++)
14        {
15            for (int j = 0; j < 11; j++)
16            {
17                if (map[i][j] == 2 || map[i][j] == 6)
18                {
19                    count = i;
20                    caw = j;
21                }
22            }
23        }
24        //printf("%c,tui");
25        if (push == 'w')
26        {
27            if (map[count - 1][caw] == 0 || map[count - 1][caw] == 4)
28            {
29                map[count][caw] -= 2;
30                map[count - 1][caw] += 2;
31            }
32            else if (map[count - 1][caw] == 3 || map[count - 1][caw] == 7)
33            {
34                if (map[count - 2][caw] == 0 || map[count - 2][caw] == 4)
35                {
36                    map[count][caw] -= 2;
37                    map[count - 1][caw] -= 1;
38                    map[count - 2][caw] += 3;
39                }
40            }
41            //break;
42        }
43        if (push == 's')
44        {
45            if (map[count + 1][caw] == 0 || map[count + 1][caw] == 4)

```



```

46     {
47         map[count][caw] -= 2;
48         map[count + 1][caw] += 2;
49     }
50     else if (map[count + 2][caw] == 0 || map[count + 2][caw] == 4)
51     {
52         if (map[count + 1][caw] == 3 || map[count + 1][caw] == 7)
53         {
54             map[count][caw] -= 2;
55             map[count + 1][caw] -= 1;
56             map[count + 2][caw] += 3;
57         }
58     }
59     //break;
60
61 }
62 if (push == 'a')
63 {
64     if (map[count][caw - 1] == 0 || map[count][caw - 1] == 4)
65     {
66         map[count][caw] -= 2;
67         map[count][caw - 1] += 2;
68     }
69     else if (map[count][caw - 2] == 0 || map[count][caw - 2] == 4)
70     {
71         if (map[count][caw - 1] == 3 || map[count][caw - 1] == 7)
72         {
73             map[count][caw] -= 2;
74             map[count][caw - 1] -= 1;
75             map[count][caw - 2] += 3;
76         }
77     }
78     //break;
79 }
80 if (push == 'd')
81 {
82     if (map[count][caw + 1] == 0 || map[count][caw + 1] == 4)
83     {
84         map[count][caw] -= 2;
85         map[count][caw + 1] += 2;
86     }
87     else if (map[count][caw + 2] == 0 || map[count][caw + 2] == 4)
88     {
89         if (map[count][caw + 1] == 3 || map[count][caw + 1] == 7)
90         {
91             map[count][caw] -= 2;
92             map[count][caw + 1] -= 1;
93             map[count][caw + 2] += 3;
94         }
95     }
96     //break;
97 }
98 }
99 return 0;
100 }

```

3.3.2 工具

3.3.2.1calendar (日历)

weekay()

```
1  PUBLIC int weekday(int year, int month, int day)//是礼拜几
2  {
3      int count;
4      count = (year - 1) + (year - 1) / 4 - (year - 1) / 100 + (year - 1) /
400 + total_day(year, month, day);
5      count %= 7;
6      return count;
7  }
```

display_month()

```
1  PUBLIC void display_month(int year, int month, int day)
2  {
3      //打印该月份日历
4      int i = 0, j = 1;
5      int week, max;
6      week = weekday(year, month, 1);
7      max = max_day(year, month);
8      printf("\n          %d - %s", year, month_dis[month - 1]);
9      printf("\n");
10     printf("\n    S    M    T    W    T    F    S\n\n");
11     for (i = 0; i < week; i++)
12         printf("        ");
13     for (j = 1; j <= max; j++)
14     {
15         printf("%6d", j);
16         if (i % 7 == 6)
17             printf("\n\n");
18         i++;
19     }
20     printf("\n\n");
21 }
```

calendar_main()

```
1  PUBLIC int calendar_main(int *fd_stdin)
2  {
3      int year, month, day;
4      char y[5], m[3], d[3]; // year, month, day
5      calendar_intro();
6      printf("Please enter year:");
7      int p = read(*fd_stdin, y, 512);
8      y[p] = 0;
9      year = 0;
10     int i = 0;
11     while (i < p)
12     {
13         year *= 10;
14         year += (y[i++] - '0');
15     }
16     printf("Please enter month:");
17     p = read(*fd_stdin, m, 512);
```

```

18     m[p] = 0;
19     month = 0;
20     i = 0;
21     while (i < p)
22     {
23         month *= 10;
24         month += (m[i++] - '0');
25     }
26     printf("Please enter day:");
27     p = read(*fd_stdin, d, 512);
28     d[p] = 0;
29     day = 0;
30     i = 0;
31     while (i < p)
32     {
33         day *= 10;
34         day += (d[i++] - '0');
35     }
36     if (month < 1 || month > 12 || day < 1 || day > 31)
37     {
38         printf("error!");
39         return -1;
40     }
41     display_week(year, month, day);
42     display_month(year, month, day);
43     printf("\nPress ENTER to continue...\n");
44     read(*fd_stdin, d, 512);
45     return 0;
46 }

```

3.3.2.2calculator (计算器)

```

1  PUBLIC int calculator_main(int* fd_stdin)
2  {
3      int option, num1, num2, result;
4      char a[20], b[20]; // num1,um2
5      calculator_list();
6      read(*fd_stdin, a, 512);
7      clear();
8      do
9      {
10         option = get_option(fd_stdin);
11         if (option == 0)
12             break;
13         do
14         {
15             printf("\nplease input a number:");
16             int p = read(*fd_stdin, a, 512);
17             a[p] = 0;
18             num1 = 0;
19             int i = 0;
20             while (i < p)
21             {
22                 num1 *= 10;
23                 num1 += (a[i++] - '0');
24             }
25             printf("\nplease input another number:");

```

```

26     p = read(*fd_stdin, b, 512);
27     b[p] = 0;
28     num2 = 0;
29     i = 0;
30     while (i < p)
31     {
32         num2 *= 10;
33         num2 += (b[i++] - '0');
34     }
35     if (option == 4 && num2 == 0)
36     {
37         printf("\nsorry!!divid can not be 0");
38     }
39     else
40     {
41         switch (option)
42         {
43             case 1:
44                 result = num1 + num2;
45                 break;
46             case 2:
47                 result = num1 - num2;
48                 break;
49             case 3:
50                 result = num1 * num2;
51                 break;
52             case 4:
53                 result = num1 / num2;
54         }
55         print_result(num1, num2, result, option);
56     }
57     } while (option == 4 && num2 == 0);
58 } while (option != 0);
59 return 0;
60 }

```

calculate()

```

1  int calculate(struct C rlt[], int size)
2  {
3      num_clear();
4      for (int i = 0; i < size; ++i) {
5          if (rlt[i].tag == 1) { //如果是数字就压入栈中
6              num_stack_push(rlt[i].data);
7          }
8          else { // 如果是操作符，就从栈中弹出两个数字做运算
9              int right = num_stack_pop();
10             int left = num_stack_pop();
11             num_stack_push(operate(left, rlt[i].data, right)); // 再将计算结果
12             // 压入栈中
13         }
14     }
15     return num_stack_pop();
16 }

```

4 成员分工

学号	姓名	分工
1853829	杨雨辰	多级目录文件系统的实现，文件操作（文件及目录的创建，文件列表的展示、文件目录的切换）的设计与实现
1850250	赵湙明	进程管理、数独的设计与实现 系统各应用画面显示的设计与实现
1851343	季潇熠	日历、推箱子、黑白棋及AI算法设计与实现
1852137	张艺腾	控制台、开机动画、清屏功能、计算器、扫雷的设计与实现
1851202	雷泓	文件操作（读文件、写文件、删除文件）、井字棋的设计与实现