



上海大学

SHANGHAI UNIVERSITY

操作系统（二）实验八报告

组	号	第 8 组
学	号	17122490
姓	名	秦敏浩

一、 代码截屏图片

shell.h: shell 类

```
5
6  DWORD  USERNAME_MAXSIZE=256;
7  DWORD  COMPUTERNAME_MAXSIZE=256;
8
9  class Shell{
10 protected:
11     HANDLE hout;
12     Command command;
13     char path[PATH_MAXSIZE];
14     char username[256];
15     char computename[256];
16
17 public:
18
19     Shell(){
20         hout=GetStdHandle(STD_OUTPUT_HANDLE);
21         getcwd(path,PATH_MAXSIZE);
22         GetUserName(username,&USERNAME_MAXSIZE);
23         GetComputerName(computename,&COMPUTERNAME_MAXSIZE);
24     }
25
26     ~Shell(){}
27
28     void run(){
29         while (true){
30             SetConsoleTextAttribute(hout,FOREGROUND_INTENSITY|FOREGROUND_GREEN);
31             printf("%s@%s ",username,computename);
32             SetConsoleTextAttribute(hout,FOREGROUND_INTENSITY|FOREGROUND_RED|FOREGROUND_BLUE);
33             printf("%s\n",path);
34             SetConsoleTextAttribute(hout,FOREGROUND_INTENSITY|FOREGROUND_RED|FOREGROUND_GREEN);
35             printf("$ ");
36
37             try{
38                 command.read();
39                 SetConsoleTextAttribute(hout,FOREGROUND_INTENSITY|FOREGROUND_RED|FOREGROUND_GREEN|FOREGROUND_BLUE);
40                 command.run(path);
41             }
42             catch(int err){
43                 SetConsoleTextAttribute(hout,FOREGROUND_INTENSITY|FOREGROUND_RED);
44                 switch(err){
45                     case -1:printf("Bash: Unknown Command \n");break;
46                     case -2:printf("Bash: Missing Parameters After \n");break;
47                 }
48                 putchar('\n');
49                 continue;
50             }
51             catch(const char* err){
52                 SetConsoleTextAttribute(hout,FOREGROUND_INTENSITY|FOREGROUND_RED);
53                 printf("Bash: %s\n", err);
54                 putchar('\n');
55                 continue;
56             }
57
58             command.output();
59             putchar('\n');
60         }
61     }
62 };
63
```

command.h 长指令类:

```
1 #ifndef _Lemon_Command
2 #define _Lemon_Command
3
4 #include "cmds.h"
5
6 static const unsigned int COMMAND_MAXSIZE=1000;
7
8 class Command{
9 public:
10     char command[COMMAND_MAXSIZE];
11     int length;
12     string result;
13     vector<string> cmds;
14
15     Command();
16     ~Command();
17
18     void read(){
19         fgets(command,COMMAND_MAXSIZE,stdin);
20         length=strlen(command);
21         command[length-1]='\0';
22
23         string cur;
24         bool stat=false;
25         cmds.clear();
26         for (int i=0;i<length;i++){
27             if (command[i]!=' '){
28                 if (stat){
29                     cur.push_back(' ');
30                 }
31                 else{
32                     if (cur.empty()){
33                         continue;
34                     }
35                     cmds.push_back(cur);
36                     cur.clear();
37                 }
38             }
39             else if (command[i]=='\n'){
40                 if (stat){
41                     stat=false;
42                     cmds.push_back(cur);
43                     cur.clear();
44                 }
45                 else{
46                     if (!cur.empty()){
47                         cmds.push_back(cur);
48                         cur.clear();
49                     }
50                     stat=true;
51                 }
52             }
53             else{
54                 cur.push_back(command[i]);
55             }
56         }
57         if (stat){
58             throw("unexpected EOF while looking for matching '\\'\n");
59         }
60     }
61
62     void run(char path[]){
63         int p=0;
64         result.clear();
65         while (p!=(int)cmds.size()){
66             try{
67                 if (cmds[p]=="exit") exit(0);
68                 else if (cmds[p]=="echo") Echo(p,cmds,result);
69                 else if (cmds[p]=="clear") Clear(p,cmds,result);
70                 else if (cmds[p]=="ls") Ls(p,cmds,result);
71                 else if (cmds[p]=="cd") Cd(p,cmds,result,path);
72                 else if (cmds[p]=="cat") Cat(p,cmds,result);
73                 else if (cmds[p]=="mkdir") Mkdir(p,cmds,result);
74                 else if (cmds[p]=="rmdir") Rmdir(p,cmds,result);
75                 else if (cmds[p]=="touch") Touch(p,cmds,result);
76                 else if (cmds[p]=="rm") Rm(p,cmds,result);
77                 else if (cmds[p]==">") Redirect(p,cmds,result);
78                 else if (cmds[p]==">>") RedirectAdd(p,cmds,result);
79                 else GetContent(p,cmds,result);
80             }
81             catch (int err){
82                 string cur="";
83                 switch(err){
84                     case -1: cur+="Command not found: '"+cmds[p]+'";break;
85                     case -2: cur+="Missing Parameters After '"+cmds[p]+'";break;
86                 }
87                 throw(cur.c_str());
88             }
89         }
90     }
91
92     void output(){
93         printf("%s\n",result.c_str());
94     }
95 };
96
97 #endif // _Lemon_Command
```

cmd.h 命令类:

```
11 static const unsigned int PATH_MAXSIZE=1000;
12
13 void Echo(int& p,vector<string>& cmds,string& ret){
14     if (p+1==(int)cmds.size()) throw -2;
15     ret=cmds[p+1];
16     p+=2;
17 }
18
19 void Clear(int &p,vector<string>& cmds,string& ret){
20     system("CLS");
21     ret.clear();
22     ++p;
23 }
24
25 void Ls(int &p,vector<string>& cmds,string& ret){
26     DIR* dir=opendir(".");
27     struct dirent* cur;
28     ret.clear();
29     while (cur=readdir(dir)){
30         ret+=cur->d_name;
31         ret+="\n";
32     }
33     ++p;
34 }
```

```

36 void Cd(int &p,vector<string>& cmds,string& ret,char path[]){
37     if (p+1==(int)cmds.size()) throw -2;
38     ret.clear();
39     if (cmds[p+1][1]==':'){
40         SetCurrentDirectoryA(cmds[p+1].c_str());
41         getcwd(path,PATH_MAXSIZE);
42         p+=2;
43         return;
44     }
45     if (cmds[p+1].back()!='\\'){
46         cmds[p+1]+='\\';
47     }
48     if (cmds[p+1][0]!='.'){
49         cmds[p+1].insert(0,"\\");
50     }
51     int cur=0,length=strlen(path);
52     if (cmds[p+1][0]=='.'&&cmds[p+1][1]=='\\'){
53         cur=1;
54     }
55     while (cur!=(int)cmds[p+1].length()){
56         char c=cmds[p+1][cur];
57         if (c=='.'){
58             if (cur+1==(int)cmds[p+1].length()){
59                 throw("Unexpected \".\" in address");
60             }
61             if (cmds[p+1][cur+1]=='\\'){
62                 cur+=2;
63                 continue;
64             }
65             if (cmds[p+1][cur+1]!='.'){
66                 throw("Unexpected \".\" in address");
67             }
68             if (path[length-1]=='\\'){
69                 path[--length]='\0';
70             }
71             while (path[length-1]!='\\'){
72                 path[--length]='\0';
73             }
74             path[--length]='\0';
75             cur+=2;
76         }
77         else{
78             path[length++]=c;
79             path[length]='\0';
80             ++cur;
81         }
82     }
83     if (length!=3){
84         path[--length]='\0';
85     }
86     SetCurrentDirectoryA(path);
87     getcwd(path,PATH_MAXSIZE);
88     p+=2;
89 }

```

```

91 void Cat(int &p,vector<string>& cmds,string& ret){
92     if (p+1==(int)cmds.size()) throw -2;
93     ret.clear();
94     char buffer[1024];
95     FILE* fp=fopen(cmds[p+1].c_str(),"r");
96     while (fgets(buffer,1024,fp)){
97         ret+=buffer;
98     }
99     fclose(fp);
100     p+=2;
101 }
102
103 void Mkdir(int &p,vector<string>& cmds,string& ret){
104     if (p+1==(int)cmds.size()) throw -2;
105     ret.clear();
106     _mkdir(cmds[p+1].c_str());
107     p+=2;
108 }
109
110 void Rmdir(int &p,vector<string>& cmds,string& ret){
111     if (p+1==(int)cmds.size()) throw -2;
112     ret.clear();
113     _rmdir(cmds[p+1].c_str());
114     p+=2;
115 }
116
117 void Touch(int &p,vector<string>& cmds,string& ret){
118     if (p+1==(int)cmds.size()) throw -2;
119     ret.clear();
120     FILE* fp=fopen(cmds[p+1].c_str(),"w");
121     fclose(fp);
122     p+=2;
123 }
124
125 void Rm(int &p,vector<string>& cmds,string& ret){
126     if (p+1==(int)cmds.size()) throw -2;
127     ret.clear();
128     DeleteFile(cmds[p+1].c_str());
129     p+=2;
130 }
131
132 void Redirect(int& p,vector<string>& cmds,string& ret){
133     if (p+1==(int)cmds.size()) throw -2;
134     FILE* fp=fopen(cmds[p+1].c_str(),"w");
135     fputs(ret.c_str(),fp);
136     fclose(fp);
137     ret.clear();
138     p+=2;
139 }
140
141 void RedirectAdd(int& p,vector<string>& cmds,string& ret){
142     if (p+1==(int)cmds.size()) throw -2;
143     FILE* fp=fopen(cmds[p+1].c_str(),"a");
144     fputs(ret.c_str(),fp);
145     fclose(fp);
146     ret.clear();
147     p+=2;
148 }
149

```

```

150 void GetContent(int &p,vector<string>& cmds,string& ret){
151     if (cmds[p].length()>=4&&cmds[p][cmds[p].length()-1]=='e'&&cmds[p][cmds[p].length()-2]=='x'&&cmds[p][cmds[p].length()-3]=='e'&&cmds[p][cmds[p].length()-4]=='.'){
152         system("CLS");
153         system(cmds[p].c_str());
154         system("CLS");
155         ++p;
156         return;
157     }
158     FILE* fp=fopen(cmds[p].c_str(),"r");
159     if (!fp) throw(-1);
160     ret.clear();
161     char buffer[1024];
162     while (fgets(buffer,1024,fp)){
163         ret+=buffer;
164     }
165     fclose(fp);
166     ++p;
167 }
168

```

环境：Windows GNU G++11 及以上

代码文件见附件或访问项目 github 仓库：

<https://github.com/Lemon-412/OS-project>

二、 代码实现的简要说明

目前实现的功能

- 简单的语法分析
 - 自左向右按空格分割
 - 但对双引号进行匹配（如 `echo "hello world"` 不会分割成三部分）
 - 捕获非法指令
 - 捕获缺少参数的指令
 - 捕获不合法的路径名
 - 捕获双引号不对齐的指令
 - 不支持括号语法
- 简单的界面
 - 展示当前用户和计算机名
 - 简单的语法高亮
 - 用户和计算机名高亮
 - 当前路径高亮
 - 指令高亮
 - 输出结果高亮
 - 错误高亮
- 实现 shell 一部分经典指令（但都不允许额外参数"-"）
 - `echo`: 回声
 - `clear`: 清屏
 - `ls`: 目录下文件
 - `cd`: 修改目录
 - 不区分大小写，并支持相对路径和绝对路径
 - 支持 `cd ../../..` 或 `cd` 等形式，虽然后者在语法多义性的考量上应该被禁止
 - `cat`: 文件内容展示
 - `mkdir`: 创建新文件夹
 - `rmdir`: 删除空文件夹
 - `touch`: 创建新文件
 - `rm`: 删除文件
 - `>`: 输出重定向
 - `>>`: 输出重定向，追加模式
 - `PATH\NAME.exe`: 执行 `PATH` 路径下的可执行程序 `NAME.exe`
 - `exit`: 退出

代码实现的简要说明

代码都是并行的，每个模块都非常短，逻辑也相对简单。没什么需要特别说明的。

- 设计 `shell.h` 类，实现了主体循环，异常捕获
- 设计 `command.h` 类，进行语法分析，异常捕获和抛出
- 设计 `cmds.h` 类，具体实现每一个指令，异常抛出

使用 `windows.h` 函数库，以获取用户名、计算机名

使用 `dirent.h` 实现目录操作和文件操作

使用 `windows command` 实现清屏，执行可执行程序等

三、 代码运行结果及结果说明

简单语法纠错:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe
Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ hello
Bash: Command not found: "hello"

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ cat
Bash: Missing Parameters After cat

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ echo "hello world
Bash: unexpected EOF while looking for matching ' "'

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ cd a.b
Bash: Unexpected "." in address
```

echo:

```
选择C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe
Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ echo hello world
Bash: Command not found: "world"

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ echo "hello world"
hello world

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ echo echo
echo

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ echo "echo echo"
echo echo
```


ls 与 cd:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe
Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ ls
.
..
bin
cmds.h
command.h
debug
Main.cbp
Main.depend
Main.layout
obj
shell.h
test.cpp

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main
$ cd .. \.. \exp1-2

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp1-2
$ ls
.
..
Main
第8组-17122490-秦敏浩-实验1-2.rar
运行结果.gif
验收报告.docx
验收报告.pdf
```

cd 复杂相对路径与绝对路径:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe
Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp1-2
$ cd c:\users\lemon\desktop

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop
$ cd OS2\..\OS2\Experiments\.\. \exp1-2\..\. \exp8\Main

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main
$ cd .\

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main
$ cd . \. \.

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main
$
```

mkdir 创建文件夹:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ ls
.
..
Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ mkdir "new folder"

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ ls
.
..
Main.exe
new folder

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ cd "new folder"

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ █
```

touch 创建文件:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ ls
.
..

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ touch 1.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ touch 2.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ ls
.
..
1.txt
2.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ █
```

cat 和>输出重定向:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 1.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ echo "hello world!" > 1.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 1.txt
hello world!

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 1.txt > 2.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 2.txt
hello world!

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ █
```

>>追加:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 1.txt
hello world!

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 2.txt
hello world!

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ echo " 17122490" >> 2.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 2.txt >> 1.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 2.txt
hello world! 17122490

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cat 1.txt
hello world!hello world! 17122490

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ █
```

rm 删除文件:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ ls
.
..
1.txt
2.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ rm 2.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ ls
.
..
1.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ rm 1.txt

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ ls
.
..
```

rmdir 删除空文件夹:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ ls
.
..

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug\new folder
$ cd ..

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ ls
.
..
Main.exe
new folder

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ rmdir "new folder"

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ ls
.
..
Main.exe
```

执行可执行程序:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\exp8\Main\bin\Debug
$ cd ..\..\..\..\expl-2\Main\bin\Debug

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\expl-2\Main\bin\Debug
$ ls
.
..
Main.exe

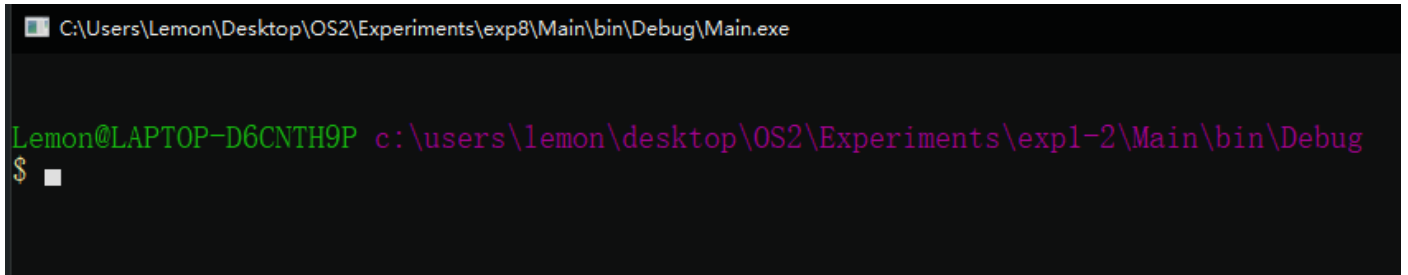
Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\expl-2\Main\bin\Debug
$ Main.exe
```

执行此 Main.exe 后立即发生跳转，回到了实验 1-2 中的可执行程序部分:

```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Inner Time:0
===== Process Ready =====
NAME
CPUTIME
ALLTIME
PRIORITY
STATE
===== Process Stuck =====
NAME
CPUTIME
ALLTIME
PRIORITY
STATE
REMAIN
===== Process Done =====
NAME
CPUTIME
ALLTIME
PRIORITY
STATE
INTIME
OUTTIME
===== System Resources =====
SYSTEM      2    2    5
请按任意键继续. . .
```

程序运行完成后返回：

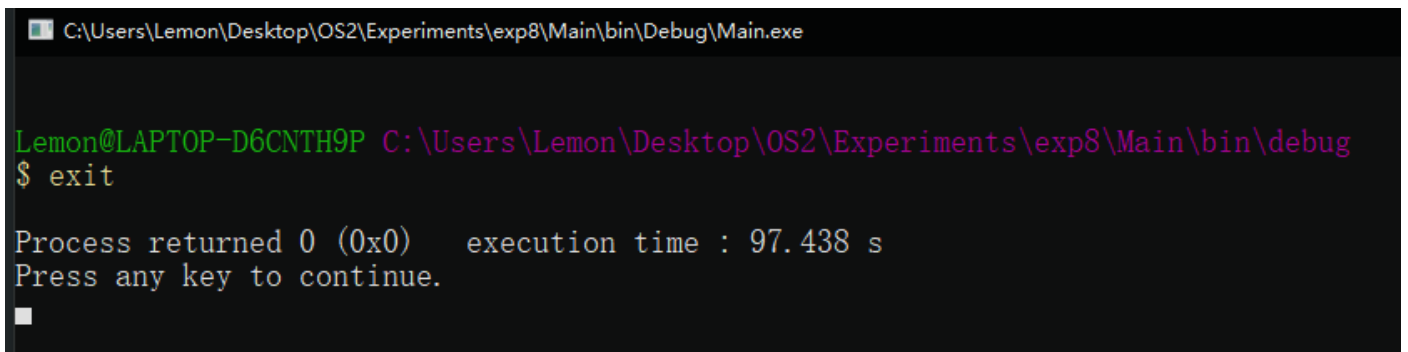


```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P c:\users\lemon\desktop\OS2\Experiments\expl-2\Main\bin\Debug
$ █
```

clear（清空屏幕）不宜使用截图展示。

exit 退出程序：



```
C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\Debug\Main.exe

Lemon@LAPTOP-D6CNTH9P C:\Users\Lemon\Desktop\OS2\Experiments\exp8\Main\bin\debug
$ exit

Process returned 0 (0x0)   execution time : 97.438 s
Press any key to continue.
█
```