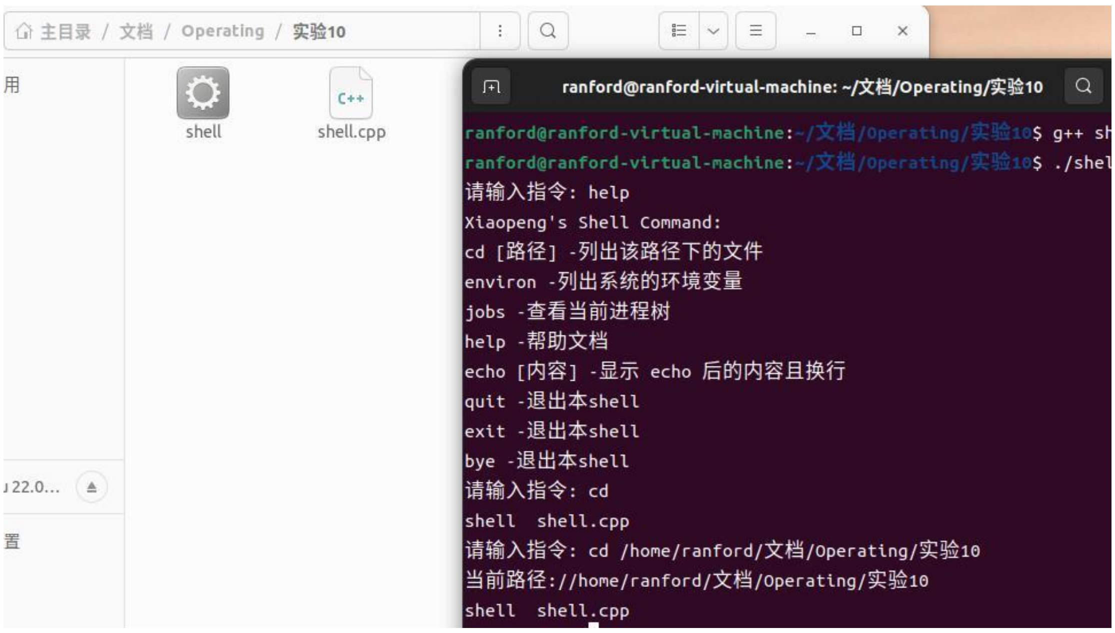
实验代码：

```
#include <sys/types.h>
#include <sys/wait.h>
#include <cstdio>
#include <unistd.h>
#include <iostream>
#include <string>

#define CMD_NUM 8
using namespace std;


const string sysCmds[CMD_NUM]={"cd","environ","jobs","help","echo","quit","exit","bye"};
//string.find("cd")!=string::npos

int main()
{
	string f_path;
	while(true)
	{
		string cmd;
		cout<<"请输入指令: ";
		getline(cin,cmd);

		//0 表示失败  1 表示内部命令  2 表示姓名
		int flag = 0;
		//内部命令的序号
		int index=-1;

		for(int i=0;i<CMD_NUM;i++)
		{
			if(cmd.find(sysCmds[i]) != string::npos)
			{
				flag = 1;
				index = i;
				if(index != 0)
				{
					f_path.clear();
				}
				else
				{
					if(cmd.size()>2)
					{
						string path(cmd,3,cmd.size()-3);
```

```
                f_path = f_path+"/"+path;
            }
        }

        break;
    }
}

if(flag==0 && cmd.find("xiaotang") != string::npos)
{
    flag = 2;
}


switch(flag)
{
case 0:
    printf("指令错误,请重新输入,帮助请输入  help\n");
    break;
case 1:
    if(index < 0)
    {
        printf("指令错误,请重新输入,帮助请输入  help\n");
    }
    else if(index < 3) //需要调用子进程
    {
        pid_t pid;
        /* fork a child process */
        pid = fork();
        if (pid < 0)
        {
            /* error occurred */
            fprintf(stderr, "Fork Failed");
            return 1;
        }
        else if (pid == 0)
        {
            /*  子进程  */
            if(index == 0) //cd
            {
                if(cmd.size()>2)
                {
                    cout<<"当前路径:"<<f_path<<endl;
                    execlp("/bin/ls","ls",f_path.c_str(),NULL);
                }
                else
                {
                        execlp("/bin/ls","ls",NULL);
                }

            }
            else if(index == 1) //environ
            {
                execlp("env","",NULL);
            }
            else if(index == 2) //jobs
            {
                execlp("pstree","-p",NULL);
            }

        }
        else    /* 父进程 */
        {
            /* 父进程将一直等待，直到子进程运行完毕*/
            waitpid(pid,NULL,0);
        }

    }
    else      //不需要子进程
    {
        if(index == 3) //help
        {
            cout<<"Xiaopeng's Shell Command:"<<endl;
            cout<<"cd [路径] -列出该路径下的文件"<<endl;
            cout<<"environ -列出系统的环境变量"<<endl;
            cout<<"jobs -查看当前进程树"<<endl;
            cout<<"help -帮助文档"<<endl;
            cout<<"echo [内容] -显示 echo 后的内容且换行"<<endl;
            cout<<"quit -退出本 shell"<<endl;
            cout<<"exit -退出本 shell"<<endl;
            cout<<"bye -退出本 shell"<<endl;
        }
        else if(index == 4) //echo
        {
            string text(cmd,5,cmd.size()-5);
            cout<<text<<endl;
        }
        else      //quit exit bye
```

```
                {
                    return 0;
                }
            }
            break;
        case 2:
            cout<<"小鹏，今天天气挺好的勒！"<<endl;
            break;
        }
    }
    return 0;

}
```

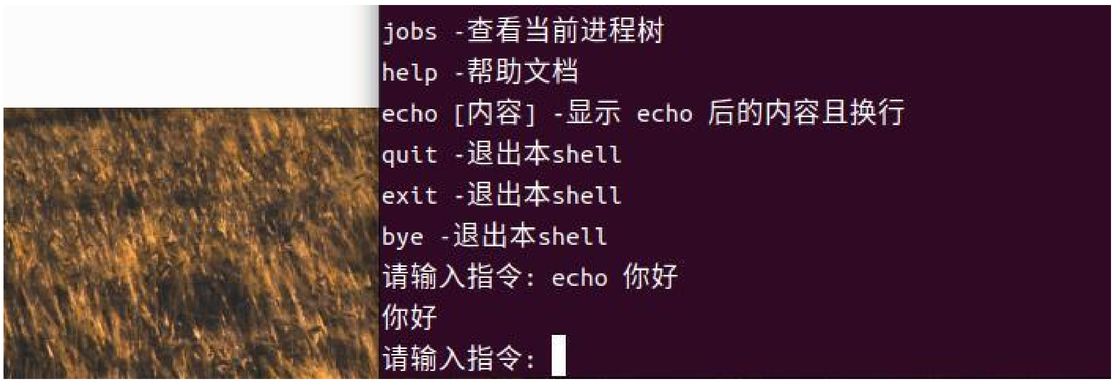
实验结果：
（1）cd



主目录 / 文档 / Operating / 实验10
用
shell
shell.cpp
22.0...
置
ranford@ranford-virtual-machine: ~/文档/Operating/实验10

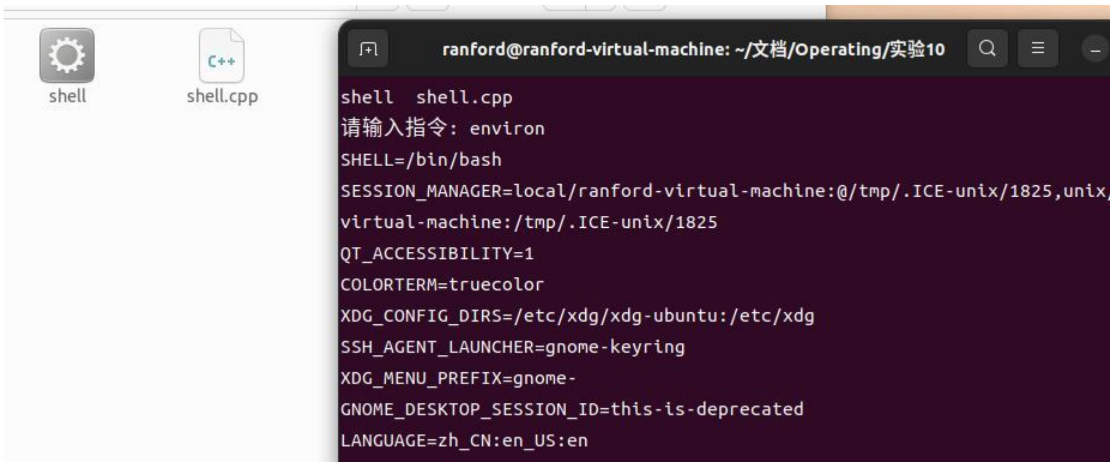
```
ranford@ranford-virtual-machine:~/文档/Operating/实验10$ g++ sh
ranford@ranford-virtual-machine:~/文档/Operating/实验10$ ./shel
请输入指令: help
Xiaopeng's Shell Command:
cd [路径] -列出该路径下的文件
environ -列出系统的环境变量
jobs -查看当前进程树
help -帮助文档
echo [内容] -显示 echo 后的内容且换行
quit -退出本shell
exit -退出本shell
bye -退出本shell
请输入指令: cd
shell  shell.cpp
请输入指令: cd /home/ranford/文档/Operating/实验10
当前路径://home/ranford/文档/Operating/实验10
shell  shell.cpp
```

（2）echo



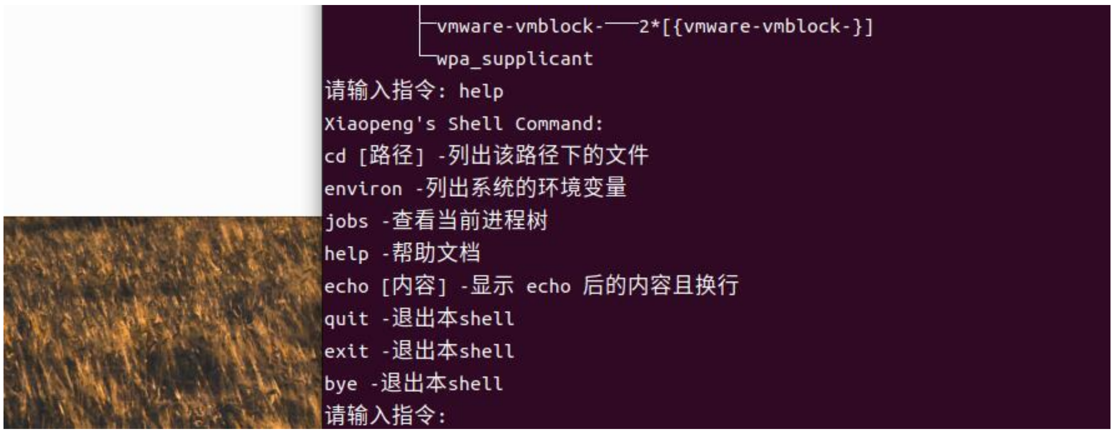
```
jobs -查看当前进程树
help -帮助文档
echo [内容] -显示 echo 后的内容且换行
quit -退出本shell
exit -退出本shell
bye -退出本shell
请输入指令: echo 你好
你好
请输入指令:
```

（3）environ



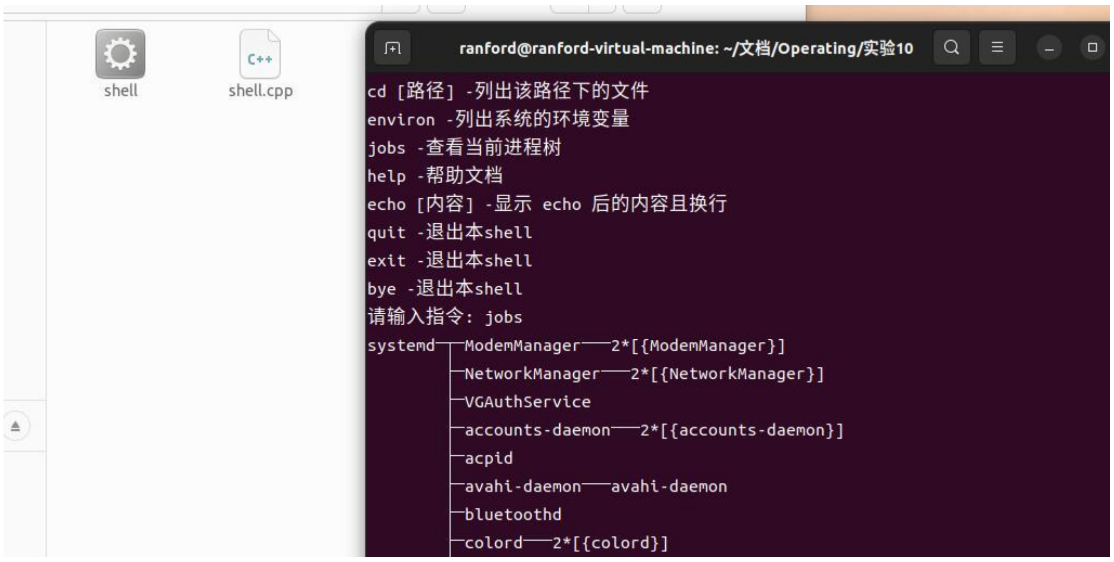
```
shell  shell.cpp
请输入指令: environ
SHELL=/bin/bash
SESSION_MANAGER=local/ranford-virtual-machine:@/tmp/.ICE-unix/1825,unix
virtual-machine:/tmp/.ICE-unix/1825
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
SSH_AGENT_LAUNCHER=gnome-keyring
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
LANGUAGE=zh_CN:en_US:en
```

（4）help



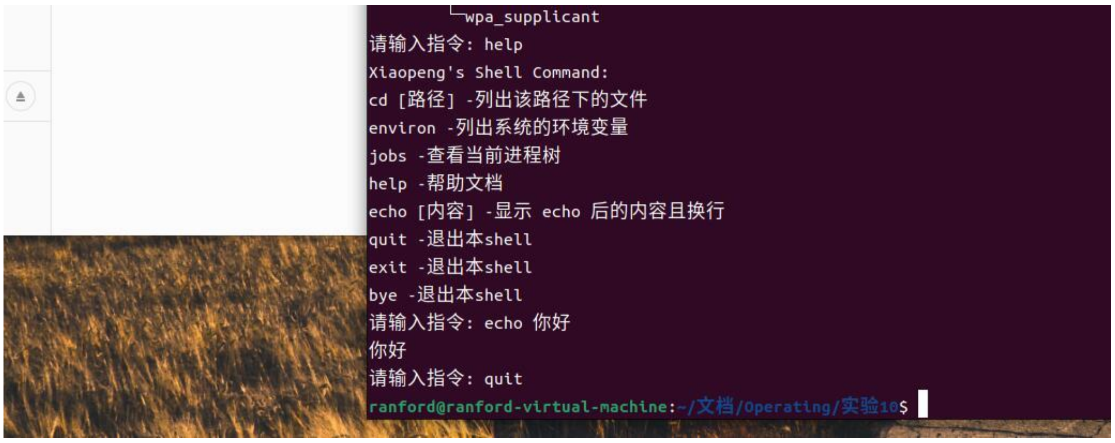
```
          ├vmware-vmblock-──2*[{vmware-vmblock-}]
          └wpa_supplicant
请输入指令: help
Xiaopeng's Shell Command:
cd [路径] -列出该路径下的文件
environ -列出系统的环境变量
jobs -查看当前进程树
help -帮助文档
echo [内容] -显示 echo 后的内容且换行
quit -退出本shell
exit -退出本shell
bye -退出本shell
请输入指令:
```

（5）jobs



```
cd [路径] -列出该路径下的文件
environ -列出系统的环境变量
jobs -查看当前进程树
help -帮助文档
echo [内容] -显示 echo 后的内容且换行
quit -退出本shell
exit -退出本shell
bye -退出本shell
请输入指令: jobs
systemd─┬ModemManager──2*[{ModemManager}]
        ├NetworkManager──2*[{NetworkManager}]
        ├VGAuthService
        ├accounts-daemon──2*[{accounts-daemon}]
        ├acpid
        ├avahi-daemon──avahi-daemon
        ├bluetoothd
        ├colord──2*[{colord}]
```

（6）quit



```
        └wpa_supplicant
请输入指令: help
Xiaopeng's Shell Command:
cd [路径] -列出该路径下的文件
environ -列出系统的环境变量
jobs -查看当前进程树
help -帮助文档
echo [内容] -显示 echo 后的内容且换行
quit -退出本shell
exit -退出本shell
bye -退出本shell
请输入指令: echo 你好
你好
请输入指令: quit
ranford@ranford-virtual-machine:~/文档/Operating/实验10$
```