

实验报告

实验名称	实验一 Linux 常用命令（一）		
实验教室	丹青 922	实验日期	2023 年 3 月 10 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

- 1、掌握Linux下文件和目录操作命令：cd、ls、mkdir、rmdir、rm
- 2、掌握Linux下文件信息显示命令：cat、more、head、tail
- 3、掌握Linux下文件复制、删除及移动命令：cp、mv
- 4、掌握 Linux 的文件排序命令：sort

二、 实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 使用命令切换到/etc 目录，并显示当前工作目录路径

```
root@ghc-virtual-machine:~/Workspace# cd /etc
root@ghc-virtual-machine:/etc# pwd
/etc
root@ghc-virtual-machine:/etc#
```

2、使用命令显示/home/lyj 目录下所有文件目录的详细信息，包括隐藏文件。

```
root@ghc-virtual-machine:/home# cd /home/ghc
root@ghc-virtual-machine:/home/ghc# ls
app  examples.desktop  number  test.b  text  x123y.test  公共的  视频  文档  音乐
date  fruit  test.a  testsort.txt  text.c  x123y.txt  模板  图片  下载  桌面
root@ghc-virtual-machine:/home/ghc# ls -a
.  .bash_logout  date  .ICEauthority  .profile  testsort.txt  x123y.test  模板  下载
..  .bashrc  examples.desktop  .local  .sudo_as_admin_successful  text  x123y.txt  模板  音乐
app  .cache  fruit  .mozilla  test.a  text.c  x123y.txt  模板  音乐
.bash_history  .config  .gnupg  number  test.b  .thunderbird  公共的  文档  图片  桌面
```

3、使用命令创建目录/home/lyj/linux，然后删除该目录。

```
root@ghc-virtual-machine:/home/ghc# mkdir linux
root@ghc-virtual-machine:/home/ghc# ls
app  examples.desktop  linux  test.a  testsort.txt  text.c  x123y.test  公共的  视频  文档  音乐
date  fruit  number  test.b  text  x123y.txt  模板  图片  下载  桌面
root@ghc-virtual-machine:/home/ghc# rmdir linux
root@ghc-virtual-machine:/home/ghc# ls
app  examples.desktop  number  test.b  text  x123y.test  公共的  视频  文档  音乐
date  fruit  test.a  testsort.txt  text.c  x123y.txt  模板  图片  下载  桌面
root@ghc-virtual-machine:/home/ghc#
```

4、使用命令 cat 用输出重定向在/home/lyj 目录下创建文件 abc，文件内容为“Hello, Linux!”，并查看该文件的内容

```
root@ghc-virtual-machine:/home/ghc# cat > abc
Hello,Linux!
^C
root@ghc-virtual-machine:/home/ghc# ls
abc  date  fruit  test.a  testsort.txt  text.c  x123y.test  公共的  视频  文档  音乐
app  examples.desktop  number  test.b  text  x123y.txt  模板  图片  下载  桌面
root@ghc-virtual-machine:/home/ghc# cat abc
Hello,Linux!
```

5、使用命令创建目录/home/lyj/ak，然后将/home/lyj/abc文件复制到该目录下，最后将该目录及其目录下的文件一起删除。

```

root@ghc-virtual-machine:/home/ghc# mkdir ak
root@ghc-virtual-machine:/home/ghc# ls
abc  app  examples.desktop  number  test.b  text  x123y.test  公共的  视频  文档  音乐
ak   date  fruit             test.a  testsort.txt  text.c  x123y.txt  模板  图片  下载  桌面
root@ghc-virtual-machine:/home/ghc# cp abc ak
root@ghc-virtual-machine:/home/ghc# ls ak
abc
root@ghc-virtual-machine:/home/ghc# rm -rf ak
root@ghc-virtual-machine:/home/ghc# ls
abc  date  fruit  test.a  testsort.txt  text.c  x123y.txt  公共的  模板  图片  下载  桌面
app  examples.desktop  number  test.b  text  x123y.test

```

6、查看文件 `/etc/adduser.conf` 的前 3 行内容，查看文件 `/etc/adduser.conf` 的最后 5 行内容。

```

root@ghc-virtual-machine:/etc# head -n adduser.conf
head: 无效的号码%s: "adduser.conf"
root@ghc-virtual-machine:/etc# head -3 adduser.conf
# /etc/adduser.conf: 'adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.

root@ghc-virtual-machine:/etc# tail -5 adduser.conf
# check user and group names also against this regular expression.
#NAME_REGEX="^[a-z][-a-z0-9_]*\$"

# use extrausers by default
#USE_EXTRAUSERS=1

```

7、分屏查看文件 `/etc/adduser.conf` 的内容。

```

# /etc/adduser.conf: 'adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.

# The DSHELL variable specifies the default login shell on your
# system.
DSHELL=/bin/bash

# The DHOME variable specifies the directory containing users' home
# directories.
DHOME=/home

# If GROUPHOMES is "yes", then the home directories will be created as
# /home/groupname/user.
GROUPHOMES=no

# If LETTERHOMES is "yes", then the created home directories will have
# an extra directory - the first letter of the user name. For example:
# /home/u/user.
LETTERHOMES=no

# The SKEL variable specifies the directory containing "skeletal" user
# files; in other words, files such as a sample .profile that will be
# copied to the new user's home directory when it is created.
SKEL=/etc/skel

# FIRST_SYSTEM_[GU]ID to LAST_SYSTEM_[GU]ID inclusive is the range for UIDs
# for dynamically allocated administrative and system accounts/groups.
# Please note that system software, such as the users allocated by the base-passwd
# package, may assume that UIDs less than 100 are unallocated.
FIRST_SYSTEM_UID=100
LAST_SYSTEM_UID=999

FIRST_SYSTEM_GID=100
LAST_SYSTEM_GID=999

# FIRST_[GU]ID to LAST_[GU]ID inclusive is the range of UIDs of dynamically
# allocated user accounts/groups.
FIRST_UID=1000
adduser.conf

```

8、使用命令cat用输出重定向在/home/lyj目录下创建文件facebook.txt，文件内容为：

google 110 5000

baidu 100 5000

guge 50 3000

sohu 100 4500

```
root@ghc-virtual-machine:/etc# cat > facebook.txt
google 110 5000
baidu 100 5000
guge 50 3000
sohu 100 4500
^C
root@ghc-virtual-machine:/etc# cat facebook.txt
google 110 5000
baidu 100 5000
guge 50 3000
sohu 100 4500
```

9. 第一列为公司名称，第2列为公司人数，第3列为员工平均工资。

利用sort命令完成下列排序：

(1) 按公司字母顺序排序

```
root@ghc-virtual-machine:/etc# sort facebook.txt
baidu 100 5000
google 110 5000
guge 50 3000
sohu 100 4500
```

(2) 按公司人数排序

```
root@ghc-virtual-machine:/etc# sort -t ' ' -k2 -n facebook.txt
guge 50 3000
baidu 100 5000
sohu 100 4500
google 110 5000
```

(3) 按公司人数排序，人数相同的按照员工平均工资升序排序

```
root@ghc-virtual-machine:/etc# sort -t ' ' -k2 -k3 -n facebook.txt
guge 50 3000
sohu 100 4500
baidu 100 5000
google 110 5000
```

(4) 按员工工资降序排序，如工资相同，则按公司人数升序排序

```
root@ghc-virtual-machine:/etc# sort -t ' ' -k3r -k2 -n facebook.txt
baidu 100 5000
google 110 5000
sohu 100 4500
guge 50 3000
```

(5) 从公司英文名称的第2个字母开始进行排序。

```
root@ghc-virtual-machine:/etc# sort -t ' ' -k1.2 facebook.txt
baidu 100 5000
sohu 100 4500
google 110 5000
guge 50 3000
t9 100 4500
```

四、 实验过程分析与讨论

遇到的困难在最后那个实验，排序的部分，对于多重要求和非第一行的排序命令还是不太熟悉，在查询 CSDN 之后学会了相关命令。

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验二 Linux 常用命令（二）		
实验教室	丹青 922	实验日期	2023 年 3 月 25 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

- 1.掌握Linux下查找文件和统计文件行数、字数和字节数命令:find ,wc ;
2. 掌握Linux下文件打包命令:tar ;
3. 掌握 Linux 下符号链接命令和文件比较命令:ln ,comm ,diff;
4. 掌握 Linux 的文件权限管理命令:chmod 。

二、 实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2)计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1.

(1)

```
root@ghc-virtual-machine:/etc# cd /home/ghc
root@ghc-virtual-machine:/home/ghc# ls
abc date fruit test.a testsort.txt text.c x123y.txt 模板 图片 下载
app examples.desktop number test.b text x123y.test 公共的 视频 文档 音乐
root@ghc-virtual-machine:/home/ghc# mkdir baz
root@ghc-virtual-machine:/home/ghc# ls
abc baz examples.desktop number test.b text x123y.test 公共的 视频 文档 音乐
app date fruit test.a testsort.txt text.c x123y.test 模板 图片 下载 桌面
root@ghc-virtual-machine:/home/ghc# cd baz
root@ghc-virtual-machine:/home/ghc/baz# cat >qux
1
22
333
4444
55555
666666
7777777
88888888
999999999
^C
root@ghc-virtual-machine:/home/ghc/baz# cat qux
1
22
333
4444
55555
666666
7777777
88888888
999999999
root@ghc-virtual-machine:/home/ghc/baz#
```

(2)

```
root@ghc-virtual-machine:/home/ghc# find -name qux -exec ls -l {} \;  
-rw-r--r-- 1 root root 54 4月 14 22:19 ./baz/qux
```

(3)

```
root@ghc-virtual-machine:/home/ghc# find -name qux -exec wc {} \;  
9 9 54 ./baz/qux
```

(4)

```
root@ghc-virtual-machine:/home/ghc# find -name qux -exec rm {} \;  
root@ghc-virtual-machine:/home/ghc# ls baz/  
root@ghc-virtual-machine:/home/ghc# ls baz  
root@ghc-virtual-machine:/home/ghc# cd baz  
root@ghc-virtual-machine:/home/ghc/baz# ls  
root@ghc-virtual-machine:/home/ghc/baz#
```

2.

```
root@ghc-virtual-machine:/home/ghc# mkdir path1  
root@ghc-virtual-machine:/home/ghc# touch path1/file1 path1/file2  
root@ghc-virtual-machine:/home/ghc# ls path1  
file1 file2  
root@ghc-virtual-machine:/home/ghc# mkdir path2  
root@ghc-virtual-machine:/home/ghc# touch path2/file3  
root@ghc-virtual-machine:/home/ghc# ls path2  
file3  
root@ghc-virtual-machine:/home/ghc# touch file4  
root@ghc-virtual-machine:/home/ghc# ls  
abc date file4 number path1 path2 test.b text x123y.test 公共的 视频 文档 音乐  
app examples.desktop fruit path1 test.a testsort.txt text.c x123y.txt 模板 图片 下载 桌面  
root@ghc-virtual-machine:/home/ghc# tar -cvf package.tar path1 file4  
path1/  
path1/file2  
path1/file1  
file4  
root@ghc-virtual-machine:/home/ghc# ls  
abc date file4 number path1 path2 test.b text x123y.test 公共的 视频 文档 音乐  
app examples.desktop fruit package.tar path1 test.a testsort.txt text.c x123y.test 模板 图片 下载 桌面  
root@ghc-virtual-machine:/home/ghc# tar -tvf package.tar  
drwxr-xr-x root/root 0 2023-04-14 22:26 path1/  
-rw-r--r-- root/root 0 2023-04-14 22:26 path1/file2  
-rw-r--r-- root/root 0 2023-04-14 22:26 path1/file1  
-rw-r--r-- root/root 0 2023-04-14 22:27 file4  
root@ghc-virtual-machine:/home/ghc# tar -rvf package.tar path2  
path2/  
path2/file3  
root@ghc-virtual-machine:/home/ghc# tar -tvf package.tar  
drwxr-xr-x root/root 0 2023-04-14 22:26 path1/  
-rw-r--r-- root/root 0 2023-04-14 22:26 path1/file2  
-rw-r--r-- root/root 0 2023-04-14 22:26 path1/file1  
-rw-r--r-- root/root 0 2023-04-14 22:27 file4  
drwxr-xr-x root/root 0 2023-04-14 22:26 path2/  
-rw-r--r-- root/root 0 2023-04-14 22:26 path2/file3
```

```
root@ghc-virtual-machine:/home/ghc# mkdir path3  
root@ghc-virtual-machine:/home/ghc# cp package.tar path3  
root@ghc-virtual-machine:/home/ghc# cd path3  
root@ghc-virtual-machine:/home/ghc/path3# tar -xvf package.tar  
path1/  
path1/file2  
path1/file1  
file4  
path2/  
path2/file3  
root@ghc-virtual-machine:/home/ghc/path3# ls  
file4 package.tar path1 path2
```

3.

```

root@ghc-virtual-machine:/home/ghc# cat >foo.txt
123
^C
root@ghc-virtual-machine:/home/ghc# ls
abc date file4 fruit package.tar path2 test.a testsort.txt
app examples.desktop foo.txt number path1 path3 test.b text
root@ghc-virtual-machine:/home/ghc# ln bar.txt foo.txt
ln: failed to access 'bar.txt': 没有那个文件或目录
root@ghc-virtual-machine:/home/ghc# ln foo.txt bar.txt
root@ghc-virtual-machine:/home/ghc# ls
abc bar.txt examples.desktop foo.txt number path1 path3 test.b
app date file4 fruit package.tar path2 test.a testsort.txt
root@ghc-virtual-machine:/home/ghc# ls -l foo.txt bar.txt
-rw-r--r-- 2 root root 4 4月 14 22:36 bar.txt
-rw-r--r-- 2 root root 4 4月 14 22:36 foo.txt
root@ghc-virtual-machine:/home/ghc# ls -li foo.txt bar.txt
264557 bar.txt 264557 foo.txt
root@ghc-virtual-machine:/home/ghc# echo "abc" > bar.txt
root@ghc-virtual-machine:/home/ghc# ls -li foo.txt bar.txt
264557 bar.txt 264557 foo.txt
root@ghc-virtual-machine:/home/ghc# cat foo.txt
abc
root@ghc-virtual-machine:/home/ghc# cat bar.txt
abc
root@ghc-virtual-machine:/home/ghc# rm foo.txt
root@ghc-virtual-machine:/home/ghc# ls -li bar.txt
264557 bar.txt
root@ghc-virtual-machine:/home/ghc# cat bar.txt
abc
root@ghc-virtual-machine:/home/ghc# ln -s baz.txt
root@ghc-virtual-machine:/home/ghc# ls -li bar.txt baz.txt
ls: 无法访问'baz.txt': 符号连接的层数过多
264557 bar.txt
root@ghc-virtual-machine:/home/ghc# ls -l bar.txt baz.txt
-rw-r--r-- 1 root root 4 4月 14 22:38 bar.txt
lrwxrwxrwx 1 root root 7 4月 14 22:40 baz.txt -> baz.txt

```

```

root@ghc-virtual-machine:/home/ghc# ls -li bar.txt baz.txt
264557 -rw-r--r-- 1 root root 4 4月 14 22:38 bar.txt
264558 lrwxrwxrwx 1 root root 7 4月 14 22:40 baz.txt -> baz.txt
root@ghc-virtual-machine:/home/ghc# cat baz.txt bar.txt
cat: baz.txt: 符号连接的层数过多
abc
root@ghc-virtual-machine:/home/ghc# cat baz.txt
cat: baz.txt: 符号连接的层数过多
root@ghc-virtual-machine:/home/ghc# rm bar.txt
root@ghc-virtual-machine:/home/ghc# cat baz.txt
cat: baz.txt: 符号连接的层数过多
root@ghc-virtual-machine:/home/ghc# s

```

4.

```

root@ghc-virtual-machine:/home/ghc# touch qux.txt
root@ghc-virtual-machine:/home/ghc# ls qux.txt
qux.txt
root@ghc-virtual-machine:/home/ghc# ls -li qux.txt
-rw-r--r-- 1 root root 0 4月 14 22:44 qux.txt
root@ghc-virtual-machine:/home/ghc# chmod 777 qux.txt
root@ghc-virtual-machine:/home/ghc# ls -li qux.txt
-rwxrwxrwx 1 root root 0 4月 14 22:44 qux.txt
root@ghc-virtual-machine:/home/ghc#

```

四、 实验过程分析与讨论

遇到的困难在与 tar 和 find 指令的不熟练，需要不时翻看笔记
有一处软连接文件显示链接层数过多，不知道如何处理

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验三 vim 编辑器及 gcc 编译器的使用		
实验教室	丹青 922	实验日期	2023 年 4 月 6 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

掌握 vim 编辑器及 gcc 编译器的使用方法。

二、 实验环境

(1) 计算机的硬件配置 PC 系列微机。

(2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. vim 编辑器和 gcc 编译器的简单使用:

(1) 在用户目录下新建一个目录，命名为 workspace1 ；

```
ghc@ghc-virtual-machine:~$ mkdir workspace1
ghc@ghc-virtual-machine:~$ ls
abc      errorInfo  fruit      path2      test.b      workspace1  模板      下载
app      examples.desktop  number     path3      testsort.txt x123y.test  视频      音乐
baz.txt  exp1       package.tar qux.txt    text        x123y.txt  图片      桌面
date     file4      path1      test.a     text.c      公共的    文档
```

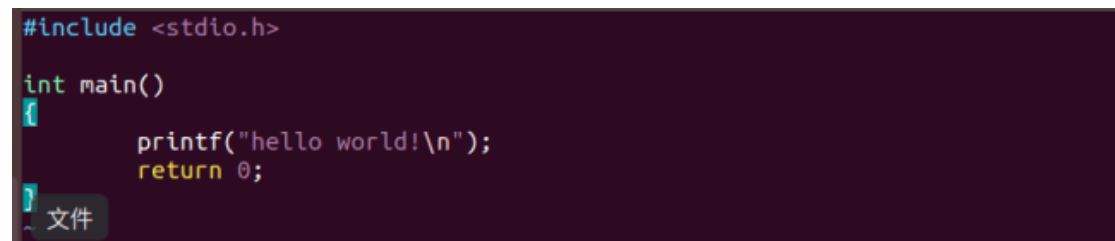
(2) 进入目录 workspace1 ；

```
ghc@ghc-virtual-machine:~$ cd workspace1/
ghc@ghc-virtual-machine:~/workspace1$ vim test.c
ghc@ghc-virtual-machine:~/workspace1$
```

(3) 在 workspace1 下用 vim 编辑器新建一个 c 语言程序文件，文件名为 test.c ， 内容为：

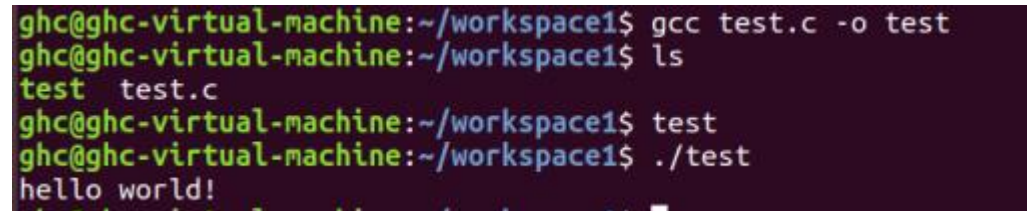
```
#include <stdio.h>
```

```
int main( )  
{  
    printf("hello world!\n");  
    return 0;  
}
```

A screenshot of a code editor with a dark background. It shows the same C program code as the previous block: `#include <stdio.h>`, `int main()`, `{`, `printf("hello world!\n");`, `return 0;`, and `}`. The cursor is at the end of the file, indicated by a small icon and the text "文件".

(4) 保存 test.c 的内容，并退出；

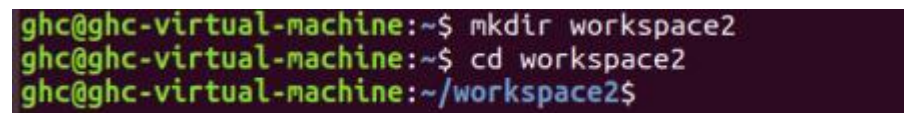
(5) 编译 test.c 文件，生成可执行文件 test，并执行，查看执行结果

A screenshot of a terminal window. It shows the following commands and output: `ghc@ghc-virtual-machine:~/workspace1$ gcc test.c -o test`, `ghc@ghc-virtual-machine:~/workspace1$ ls` (output: `test test.c`), `ghc@ghc-virtual-machine:~/workspace1$ test` (output: `hello world!`), and `ghc@ghc-virtual-machine:~/workspace1$./test` (output: `hello world!`).

2. vim 编辑器的详细使用：

(1) 在用户目录下创建一个名为 workspace2 的目录；

(2) 进入 workspace2 目录；

A screenshot of a terminal window showing the commands `ghc@ghc-virtual-machine:~$ mkdir workspace2`, `ghc@ghc-virtual-machine:~$ cd workspace2`, and `ghc@ghc-virtual-machine:~/workspace2$`.

(3) 使用以下命令：

将文件 /etc/gai.conf 的内容复制到当前目录下的新建文件 gai.conf 中；


```
ghc@ghc-virtual-machine:~/workspace2$ cat /etc/gai.conf > ./gai.conf
ghc@ghc-virtual-machine:~/workspace2$ ls
gai.conf
```

(4) 使用 vim 编辑当前目录下的 gai.conf ；

```
1 # Configuration for getaddrinfo(3).
2 #
3 # So far only configuration for the destination address sorting is needed.
4 # RFC 3484 governs the sorting. But the RFC also says that system
5 # administrators should be able to overwrite the defaults. This can be
6 # achieved here.
7 #
8 # All lines have an initial identifier specifying the option followed by
9 # up to two values. Information specified in this file replaces the
10 # default information. Complete absence of data of one kind causes the
11 # appropriate default information to be used. The supported commands include:
12 #
13 # reload <yes|no>
14 #   If set to yes, each getaddrinfo(3) call will check whether this file
15 #   changed and if necessary reload. This option should not really be
16 #   used. There are possible runtime problems. The default is no.
17 #
18 # label <mask> <value>
19 #   Add another rule to the RFC 3484 label table. See section 2.1 in
20 #   RFC 3484. The default is:
21 #
22 #label ::1/128      0
23 #label ::/0        1
24 #label 2002::/16    2
25 #label ::/96       3
26 #label ::ffff:0:0/96 4
27 #label fec0::/10    5
28 #label fc00::/7     6
29 #label 2001:0::/32  7
30 #
:set nu                                     1,1
```

(5) 将光标移到第 18 行；

```
1 # Configuration for getaddrinfo(3).
2 #
3 # So far only configuration for the destination address sorting is needed.
4 # RFC 3484 governs the sorting. But the RFC also says that system
5 # administrators should be able to overwrite the defaults. This can be
6 # achieved here.
7 #
8 # All lines have an initial identifier specifying the option followed by
9 # up to two values. Information specified in this file replaces the
10 # default information. Complete absence of data of one kind causes the
11 # appropriate default information to be used. The supported commands include:
12 #
13 # reload <yes|no>
14 #   If set to yes, each getaddrinfo(3) call will check whether this file
15 #   changed and if necessary reload. This option should not really be
16 #   used. There are possible runtime problems. The default is no.
17 #
18 # label <mask> <value>
19 #   Add another rule to the RFC 3484 label table. See section 2.1 in
20 #   RFC 3484. The default is:
21 #
22 #label ::1/128      0
23 #label ::/0        1
24 #label 2002::/16    2
25 #label ::/96       3
26 #label ::ffff:0:0/96 4
27 #label fec0::/10    5
28 #label fc00::/7     6
29 #label 2001:0::/32  7
30 #
```

(6) 复制该行内容;

(7) 将光标移到最后一行行首;

```
36 # site-local IPv4 and IPv6 addresses a lookup for a global address would
37 # see the IPv6 be preferred. The result is a long delay because the
38 # site-local IPv6 addresses cannot be used while the IPv4 address is
39 # (at least for the foreseeable future) NATed. We also treat Teredo
40 # tunnels special.
41 #
42 # precedence <mask> <value>
43 # Add another rule to the RFC 3484 precedence table. See section 2.1
44 # and 10.3 in RFC 3484. The default is:
45 #
46 #precedence ::1/128 50
47 #precedence ::/0 40
48 #precedence 2002::/16 30
49 #precedence ::/96 20
50 #precedence ::ffff:0:0/96 10
51 #
52 # For sites which prefer IPv4 connections change the last line to
53 #
54 #precedence ::ffff:0:0/96 100
55
56 #
57 # scopev4 <mask> <value>
58 # Add another rule to the RFC 6724 scope table for IPv4 addresses.
59 # By default the scope IDs described in section 3.2 in RFC 6724 are
60 # used. Changing these defaults should hardly ever be necessary.
61 # The defaults are equivalent to:
62 #
63 #scopev4 ::ffff:169.254.0.0/112 2
64 #scopev4 ::ffff:127.0.0.0/104 2
65 #scopev4 ::ffff:0.0.0.0/96 14
```

65,1

(8) 粘贴复制行的内容;

```
45 #
46 #precedence ::1/128 50
47 #precedence ::/0 40
48 #precedence 2002::/16 30
49 #precedence ::/96 20
50 #precedence ::ffff:0:0/96 10
51 #
52 # For sites which prefer IPv4 connections change the last line to
53 #
54 #precedence ::ffff:0:0/96 100
55
56 #
57 # scopev4 <mask> <value>
58 # Add another rule to the RFC 6724 scope table for IPv4 addresses.
59 # By default the scope IDs described in section 3.2 in RFC 6724 are
60 # used. Changing these defaults should hardly ever be necessary.
61 # The defaults are equivalent to:
62 #
63 #scopev4 ::ffff:169.254.0.0/112 2
64 #scopev4 ::ffff:127.0.0.0/104 2
65 #scopev4 ::ffff:0.0.0.0/96 14
66 # label <mask> <value>
```

(9) 撤销第 8 步的动作;

```

51 #
52 #   For sites which prefer IPv4 connections change the last line to
53 #
54 #precedence ::ffff:0:0/96  100
55
56 #
57 # scopev4 <mask> <value>
58 #   Add another rule to the RFC 6724 scope table for IPv4 addresses.
59 #   By default the scope IDs described in section 3.2 in RFC 6724 are
60 #   used. Changing these defaults should hardly ever be necessary.
61 #   The defaults are equivalent to:
62 #
63 #scopev4 ::ffff:169.254.0.0/112  2
64 #scopev4 ::ffff:127.0.0.0/104    2
65 #scopev4 ::ffff:0.0.0.0/96       14

```

~
1 行被去掉; before #2 32 秒之前

(10) 存盘但不退出;

```

59 #   By default the scope IDs described in section 3.2 in RFC 6724 are
60 #   used. Changing these defaults should hardly ever be necessary.
61 #   The defaults are equivalent to:
62 #
63 #scopev4 ::ffff:169.254.0.0/112  2
64 #scopev4 ::ffff:127.0.0.0/104    2
65 #scopev4 ::ffff:0.0.0.0/96       14

```

~
"gai.conf" 65L, 2584C 已写入

(11) 将光标移到首行;

```

1  Configuration for getaddrinfo(3).
2 #
3 # So far only configuration for the destination address sorting is needed.
4 # RFC 3484 governs the sorting. But the RFC also says that system
5 # administrators should be able to overwrite the defaults. This can be

```

(12) 插入模式下输入 "Hello, this is vim world!";

```

1 Hello,this is vim world!# Configuration for getaddrinfo(3).
2 #
3 # So far only configuration for the destination address sorting is needed.
4 # RFC 3484 governs the sorting. But the RFC also says that system

```

(13) 删除字符串 "this";

```

1 Hello, is vim world!# Configuration for getaddrinfo(3).
2 #
3 # So far only configuration for the destination address sorting is needed.
4 # RFC 3484 governs the sorting. But the RFC also says that system
5 # administrators should be able to overwrite the defaults. This can be

```

(14) 强制退出 vim , 不存盘

```

29 #label 2001:0:0:/32  7
30 #
:q!

```

四、 实验过程分析与讨论

对 vim 的使用仍然需要熟练度

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验四 用户和用户组管理		
实验教室	丹青 922	实验日期	2023 年 4 月 21 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

1. 掌握用户管理命令，包括命令 `useradd` 、 `usermod` 、 `userdel` 、 `newusers` ；
2. 掌握用户组管理命令，包括命令 `groupadd` 、 `groupdel` 、 `gpasswd` ；
3. 掌握用户和用户组维护命令，包括命令 `passwd` 、 `su` 、 `sudo`

二、 实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 创建一个名为 foo ，描述信息为 bar ，登录 shell 为 /bin/sh ，家目录为 /home/foo 的用户，并设置登陆口令为 123456 ；

```
root@ghc-virtual-machine:~# useradd foo -c bar -s /bin/bash -m
root@ghc-virtual-machine:~# passwd foo
输入新的 UNIX 密码:
重新输入新的 UNIX 密码:
passwd: 已成功更新密码
```

2. 使用命令从 root 用户切换到用户 foo ，修改 foo 的 UID 为 2000 ，其 shell 类型为 /bin/csh ；

```
root@ghc-virtual-machine:/home/ghc/workspace2# su foo
foo@ghc-virtual-machine:/home/ghc/workspace2$
```

```
root@ghc-virtual-machine:/home/ghc/workspace2# usermod -u 2000 -s /bin/csh foo
root@ghc-virtual-machine:/home/ghc/workspace2# finger foo
Login: foo                               Name: bar
Directory: /home/foo                     Shell: /bin/csh
Never logged in.
No mail.
No Plan.
```

3. 从用户 foo 切换到 root ；

```
foo@ghc-virtual-machine:/home/ghc/workspace2# su root
root@ghc-virtual-machine:/home/ghc/workspace2#
```

4. 删除 foo 用户，并在删除该用户的同时一并删除其家目录；

```
root@ghc-virtual-machine:/home/ghc/workspace2# userdel -r foo
```

5. 使用命令 newusers 批量创建用户，并使用命令 chpasswd 为这些批量创建的用户设置密码（密码也需要批量设置），查看 /etc/passwd 文件检查用户是否创建成功；

```
root@ghc-virtual-machine:/home/ghc/workspace2# vim userfile
root@ghc-virtual-machine:/home/ghc/workspace2# vim passwdfile
root@ghc-virtual-machine:/home/ghc/workspace2# newusers < userfile
root@ghc-virtual-machine:/home/ghc/workspace2# chpasswd < passwdfile
root@ghc-virtual-machine:/home/ghc/workspace2# less /etc/passwd
```



```
gdm:x:121:123:Gnome Display Manager:/var/lib/xdm/gdm3:/bin/sh
ghc:x:1000:1000:GHC,,,:/home/ghc:/bin/bash
A1:x:2021:2021:/:/home/A1:/bin/bash
A2:x:2022:2022:/:/home/A2:/bin/bash
A3:x:2023:2023:/:/home/A3:/bin/bash
A4:x:2024:2024:/:/home/A4:/bin/bash
A5:x:2025:2025:/:/home/A5:/bin/bash
(END)
```

6. 创建用户组 group1 ，并在创建时设置其 GID 为 3000 ；

```
root@ghc-virtual-machine:/home/ghc/workspace2# groupadd -g 3000 group1
```

7. 在用户组 group1 中添加两个之前批量创建的用户；

```
root@ghc-virtual-machine:/home/ghc/workspace2# gpasswd -a A1 group1
正在将用户“A1”加入到“group1”组中
root@ghc-virtual-machine:/home/ghc/workspace2# gpasswd -a A2 group1
正在将用户“A2”加入到“group1”组中
```

8. 切换到 group1 组中的任一用户，在该用户下使用 sudo 命令查看 /etc/shadow 文件，检查上述操作是否可以执行；若不能执行，修改 sudoers 文件使得该用户可以查看文件 /etc/shadow 的内容

```
root@ghc-virtual-machine:/home/ghc/workspace2# su A1
A1@ghc-virtual-machine:/home/ghc/workspace2$ sudo cat /etc/shadow
[sudo] A1 的密码:
A1 不在 sudoers 文件中。此事将被报告。
```

```
# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
A1      ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin   ALL=(ALL) ALL

# Allow members of group sudo to execute any command
```

```
A1@ghc-virtual-machine:/home/ghc/workspace2$ sudo cat /etc/shadow | tail -n 5
A1:$6$06qmmvGe$nS8p3GrZz523ee0Xp0TDP2dF3eWeRbc6rGv8a2dYCHCAoa6pdBuCxIsRhNHwcsPY0oSAppqzGf0JeCuf0zB1Sp0:19
:99999:7:::
A2:$6$2AUBgo3C$9zSDaLsmJYACcoPGQs8nLJMVHvHkjJMe7FVE60cFJQ/AnJwje.i30vZG0ye6BDKW5ac8/iYRYGi/YhwpZw6Xk1:19
:99999:7:::
A3:$6$zN6LZpXd$L4PPch0nLzN5jVbqtoQTVbj8Kt9BdZH.g/5BfuRn6uDkSCU66Ddz5GGIIDicQAsg.3a3VRaIN.OD.D0yokV0g.:19
:99999:7:::
A4:$6$/OhfHUw/$npIFKua/ODyQ/4Mevia1yzLns4zVqo517pL/7fiLITME8RXRVvt9X0AmOz4b90026HhEcWEKjwBwY1lk9kEof.:19
:99999:7:::
A5:$6$t1x.zkYs$ICImcr0f6TsTZpOPT0hC0iMbIMnQr0.yzqtLXMZaoDesnW8ERe0tju0ff5dVDw53CKSor3o0Pn2XsDjk5A4sy0:19
:99999:7:::
A1@ghc-virtual-machine:/home/ghc/workspace2$
```

四、 实验过程分析与讨论

在批量创建用户时遇到了部分困难，但在查询资料后成功解决了

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验五 Shell 程序的创建及条件判断语句		
实验教室	丹青 922	实验日期	2023 年 4 月 29 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

1. 掌握Shell程序的创建过程及Shell程序的执行方法；
2. 掌握Shell变量的定义方法，及用户定义变量、参数位置等；
3. 掌握变量表达式，包括字符串比较、数字比较、逻辑测试、文件测试；
4. 掌握条件判断语句，如 if 语句、 case 语句

二、 实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 定义变量 foo 的值为 200 ，并将其显示在屏幕上（终端上执行）；

```
root@ghc-virtual-machine:/home/ghc/workspace2# foo=200
root@ghc-virtual-machine:/home/ghc/workspace2# echo $foo
200
```

2. 定义变量 bar 的值为 100 ，并使用 test 命令比较其值是否大于 150 ，并显示 test 命令的退出码（终端上执行）；

```
root@ghc-virtual-machine:/home/ghc/workspace2# bar=100
root@ghc-virtual-machine:/home/ghc/workspace2# test $bar -gt 150
root@ghc-virtual-machine:/home/ghc/workspace2# echo $?
1
```

3. 创建一个Shell程序，其功能为显示计算机主机名（ hostname ）和系统时间（ date ）；

```
root@ghc-virtual-machine:/home/ghc/workspace2# vim pri.sh
root@ghc-virtual-machine:/home/ghc/workspace2# bash pri.sh
ghc-virtual-machine
2023年 05月 23日 星期二 13:46:58 CST
```

4. 创建一个Shell程序，要求可以处理一个输入参数，判断该输入参数是否为水仙花数；

所谓水仙花数是指一个 3 位数，该数字每位数字的 3 次幂之和等于其本身，例如：

$$153 == 1^3 + 3^3 + 5^3$$

根据上述定义 153 是水仙花数。编写程序时要求首先进行输入参数个数判断，判断是否有输入参数存

在：如果没有则给出提示信息；否则给出该数是否是水仙花数。

要求对 153 、 124 和 370 进行测试判

断。

```
#!/bin/bash
PATH=$PATH
export PATH

if [ "$#" -ge 1 ]; then
    num=$1
    a=$((num%10))
    #echo "$a"
    num=$((num/10))
    b=$((num%10))
    #echo "$b"
    num=$((num/10))
    c=$((num%10))
    #echo "$c"

    sum=$(( ${a}*$a*$a+${b}*$b*$b+${c}*$c*$c ))
    #echo "$sum"

    if [ "$sum" == "$1" ]; then
        echo "$1 is a flower number"
    else
        echo "$1 is not a flower number"
    fi
else
    echo "too few arguments"
fi
```

```
root@ghc-virtual-machine:/home/ghc/workspace2# vim flower.sh
root@ghc-virtual-machine:/home/ghc/workspace2# bash flower.sh 153
153 is a flower number
root@ghc-virtual-machine:/home/ghc/workspace2# bash flower.sh 124
124 is not a flower number
root@ghc-virtual-machine:/home/ghc/workspace2# bash flower.sh 370
370 is a flower number
root@ghc-virtual-machine:/home/ghc/workspace2# bash flower.sh
too few arguments
```

5. 创建一个Shell程序，输入 3 个参数，计算 3 个输入变量的和并输出；

```
#!/bin/bash
echo "$(($1+$2+$3))"
```

```
root@ghc-virtual-machine:/home/ghc/workspace2# vim plus.sh
root@ghc-virtual-machine:/home/ghc/workspace2# bash plus.sh 1 2 3
6
```

6. 创建一个Shell程序，输入学生成绩，给出该成绩对应的等级：90 分以上为 A ， 80-90 为 B ， 70-80为 C ， 60-70 为 D ， 小于 60 分为 E 。要求使用

if

elif

else

fi

实现

```
#!/bin/bash
PATH=$PATH
export PATH

if [ "$1" -ge 90 ];then
    echo "level is A"
elif [ "$1" -ge 80 ];then
    echo "level is B"
elif [ "$1" -ge 70 ];then
    echo "level is C"
elif [ "$1" -ge 60 ];then
    echo "level is D"
else
    echo "level is E"
fi
```

```
root@ghc-virtual-machine:/home/ghc/workspace2# vim checkgrade.sh
root@ghc-virtual-machine:/home/ghc/workspace2# bash checkgrade.sh 100
level is A
root@ghc-virtual-machine:/home/ghc/workspace2# bash checkgrade.sh 10
level is E
root@ghc-virtual-machine:/home/ghc/workspace2# bash checkgrade.sh 88
level is B
```

四、 实验过程分析与讨论

对于条件判断的使用仍然不熟练

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验六 Shell 循环控制语句		
实验教室	丹青 922	实验日期	2023 年 4 月 30 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

1. 熟练掌握 Shell 循环语句： for 、 while 、 until ；
2. 熟练掌握 Shell 循环控制语句： break 、 continue

二、 实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 编写一个 Shell 脚本，利用 for 循环把当前目录下的所有 *.c 文件复制到指定的目录中（如~/workspace ）；
可以事先在当前目录下建立若干 *.c 文件用于测试。

```
root@ghc-virtual-machine:~/workspace2# touch a.c b.c c.c d.c
root@ghc-virtual-machine:~/workspace2# ls
a.c b.c c.c d.c
```

```
root@ghc-virtual-machine:~/workspace2# vim cpc.sh
root@ghc-virtual-machine:~/workspace2# bash cpc.sh
which directory do you want to copy in:workspaceGHC
```

```
#!/bin/bash
read -p "which directory do you want to copy in:" dir
test -d $dir || mkdir $dir
tmp=$(ls *.c)
for i in $tmp
do
    cp $i $dir
done
```

```

root@ghc-virtual-machine:~/workspace2# ls
a.c b.c c.c cpc.sh d.c workspaceGHC
root@ghc-virtual-machine:~/workspace2# cd workspaceGHC/
root@ghc-virtual-machine:~/workspace2/workspaceGHC# ls
a.c b.c c.c d.c

```

2. 编写Shell脚本，利用 while 循环求前 10 个偶数之和，并输出结果；

```

root@ghc-virtual-machine:~/workspace2# vim sum.sh
root@ghc-virtual-machine:~/workspace2# bash sum.sh
110

```

```

#!/bin/bash

i=0
j=1
sum=0
while [ "$i" -lt 10 ]
do
    if [ $((j%2)) -eq 0 ]; then
        sum=$((sum+j))
        i=$((i+1))
        #echo "$sum $j $i"
    fi
    j=$((j+1))
done

echo "$sum"

```

3. 编写Shell脚本，利用 until 循环求 1 到 10 的平方和，并输出结果；

```

root@ghc-virtual-machine:~/workspace2# vim sum.sh
root@ghc-virtual-machine:~/workspace2# bash sum.sh
385

```

```

文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

#!/bin/bash

i=0
sum=0
until [ "$i" -eq 10 ]
do
    i=$((i+1))
    sum=$((sum+i*i))
done

echo "$sum"

```

4. 运行下列程序，并观察程序的运行结果。将程序中的 --- 分别替换为 break 、 break

2 、 continue 、 continue 2 ，并观察四种情况下的实验结果。

```
#!/bin/bash
```

```
for i in a b c d; do
```

```
    echo -n $i
```

```
    for j in 1 2 3 4 5 6 7 8 9 10; do
```

```
        if [[ $j -eq 5 ]]; then
```

```
            ---
```

```
        fi
```

```
    echo -n $j
```

```
done
```

```
echo ''
```

```
done
```

```
root@ghc-virtual-machine:~/workspace2# vim test.sh
root@ghc-virtual-machine:~/workspace2# vim test1.sh
root@ghc-virtual-machine:~/workspace2# vim test2.sh
root@ghc-virtual-machine:~/workspace2# vim test3.sh
root@ghc-virtual-machine:~/workspace2# vim test4.sh
```

```
#!/bin/bash

for i in a b c d; do
    echo -n $i
    for j in 1 2 3 4 5 6 7 8 9 10;do
        if [[ $j -eq 5 ]]; then
            break
        fi
        echo -n $j
    done
    echo ' '
done
```

```
a1234
b1234
c1234
d1234
```

```
#!/bin/bash

for i in a b c d; do
    echo -n $i
    for j in 1 2 3 4 5 6 7 8 9 10;do
        if [[ $j -eq 5 ]]; then
            break 2
        fi
        echo -n $j
    done
    echo ' '
done
```

```
root@ghc-virtual-machine:~/workspace2# bash test2.sh
a1234root@ghc-virtual-machine:~/workspace2# bash tes
```

```
#!/bin/bash

for i in a b c d; do
    echo -n $i
    for j in 1 2 3 4 5 6 7 8 9 10;do
        if [[ $j -eq 5 ]]; then
            continue
        fi
        echo -n $j
    done
    echo ' '
done
```

```
a1234root@ghc-virtual-machine:~/workspace2# bash test3.sh
a1234678910
b1234678910
c1234678910
d1234678910
```

```
#!/bin/bash

for i in a b c d; do
    echo -n $i
    for j in 1 2 3 4 5 6 7 8 9 10;do
        if [[ $j -eq 5 ]]; then
            continue 2
        fi
        echo -n $j
    done
    echo ' '
done

root@ghc-virtual-machine:~/workspace2# bash test4.sh
a1234b1234c1234d1234root@ghc-virtual-machine:~/workspace2#
```

四、 实验过程分析与讨论

对 break 和 continue 语句的含义需要熟记

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验七 Shell 函数		
实验教室	丹青 922	实验日期	2023 年 4 月 30 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

1. 掌握 Shell 函数的定义方法；
2. 掌握 Shell 函数的参数传递、调用和返回值；
3. 掌握 Shell 函数的递归调用方法；
4. 理解 Shell 函数的嵌套。

二、 实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 编写Shell脚本，实现一个函数，对两个数的和进行求解，并输出结果；

```
root@ghc-virtual-machine:~/workspace2# vim func.sh
root@ghc-virtual-machine:~/workspace2# bash func.sh
input two number:1 2
3
```

```
#!/bin/bash
read -p "input two number:" a b
function ff(){
    echo $((a+b))
}
ff a b
```

2. 编写Shell脚本，在脚本中定义一个递归函数，实现 n 的阶乘的求解；

```
#!/bin/bash

read -p "input the number:" a
function ff(){
    if [ $1 -eq 1 ]; then
        return 1
    else
        ff $(( $1 - 1 ))
        return $(( $1 * $? ))
    fi
}

ff $a
echo $?
```

```
root@ghc-virtual-machine:~/workspace2# vim func.sh
root@ghc-virtual-machine:~/workspace2# bash func.sh
input the number:5
120
```

3. 一个Shell脚本的内容如下所示:

```
#!/bin/bash

function first() {
    function second() {
        function third() {
            echo "-3- here is in the third func."
        }
        echo "-2- here is in the second func."
        third
    }
    echo "-1- here is in the first func."
    second
}

echo "starting..."
```

first

试运行该程序，并观察程序运行结果，理解函数嵌套的含义

```
#!/bin/bash
function first(){
    function second(){
        function third(){
            echo "-3- here is in the third func."
        }
        echo "-2- here is in the second func."
        third
    }
    echo "-1- here is in the first func."
    second
}
echo "starting..."
first
```

```
root@ghc-virtual-machine:~/workspace2# vim func.sh
root@ghc-virtual-machine:~/workspace2# bash func.sh
starting...
-1- here is in the first func.
-2- here is in the second func.
-3- here is in the third func.
```

四、 实验过程分析与讨论

对函数的返回值理解不到位

五、 指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验八 sed 和 awk		
实验教室	丹青 922	实验日期	2023 年 4 月 30 日
学 号	2021213160	姓 名	郭寒誠
专业班级	计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

1. 掌握 sed 基本编辑命令的使用方法；
2. 掌握 sed 与 Shell 变量的交互方法；
3. 掌握 awk 命令的使用方法；
4. 掌握 awk 与 Shell 变量的交互方法。

二、 实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 文件 quote.txt 的内容如下所示：

The honeysuckle band played all night long for only \$90.

It was an evening of splendid music and company.

Too bad the disco floor fell through at 23:10.

The local nurse Miss P. Neave was in attendance.

试使用 sed 命令实现如下功能：

- (1) 删除 \$ 符号；

```
root@ghc-virtual-machine:~# cat quote.txt | sed 's/\$/g'
The honeysuckle band played all night long for only 90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P. Neave was in attendance.
```

- (2) 显示包含 music 文字的行内容及行号；

```
root@ghc-virtual-machine:~# cat -n quote.txt | sed -n '/music/p'
2 It was an evening of splendid music and company.
```

(3) 在第 4 行后面追加内容: "hello world!";

```
root@ghc-virtual-machine:~# cat quote.txt | sed '4a hello world!'
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor through at 23:10.
The local nurse Miss P.Neave was in attendance.
hello world!
```

(4) 将文本 "The" 替换为 "Quod";

```
root@ghc-virtual-machine:~# cat quote.txt | sed 's/The/Quod/g'
Quod honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor through at 23:10.
Quod local nurse Miss P.Neave was in attendance.
```

(5) 将第 3 行内容修改为: "This is the third line.";

```
root@ghc-virtual-machine:~# cat quote.txt | sed '3c This is the third line.'
The honeysuckle band played all night long for only $90.
This is the third line.
The local nurse Miss P.Neave was in attendance.
```

(6) 删除第 2 行内容;

```
root@ghc-virtual-machine:~# cat quote.txt | sed '2d'
The honeysuckle band played all night long for only $90.
Too bad the disco floor through at 23:10.
The local nurse Miss P.Neave was in attendance.
```

(7) 设置Shell变量 var 的值为 evening , 用 sed 命令查找匹配 var 变量值的行。

```
root@ghc-virtual-machine:~# cat quote.txt | sed -n "/$var/p"
It was an evening of splendid music and company.
```

2. 文件 numbers.txt 的内容如下所示:

one : two : three

four : five : six

注: 每个冒号前后都有空格。

试使用 awk 命令实现如下功能: 分别以 空格 和 冒号 做分隔

符，显示第 2 列的内容，观察两者的区别：

```
root@ghc-virtual-machine:~# awk 'BEGIN{FS=":"}{print $2}' numbers.txt
two
five
root@ghc-virtual-machine:~# awk 'BEGIN{FS=" "}{print $2}' numbers.txt
:
```

3. 已知文件 `foo.txt` 中存储的都是数字，且每行都包含 3 个数字，数字之前以空格作为分隔符。试找出

`foo.txt` 中的所有偶数进行打印，并输出偶数的个数。

要求：判断每行的 3 个数字是否为偶数时用循环结果，即要求程序里包含循环和分支结构。

例如： `foo.txt` 内容为：

2 4 3

15 46 79

则输出为：

even:

2

4

46

numbers:

3

```
root@ghc-virtual-machine:~# awk 'BEGIN{sum=0;print "Even:"} {for(i=1;i<=NF;i++) if($i%2==0){print $i;sum++}}
END{print "numbers:\n" sum}' foo.txt
Even:
2
4
46
numbers:
3
```

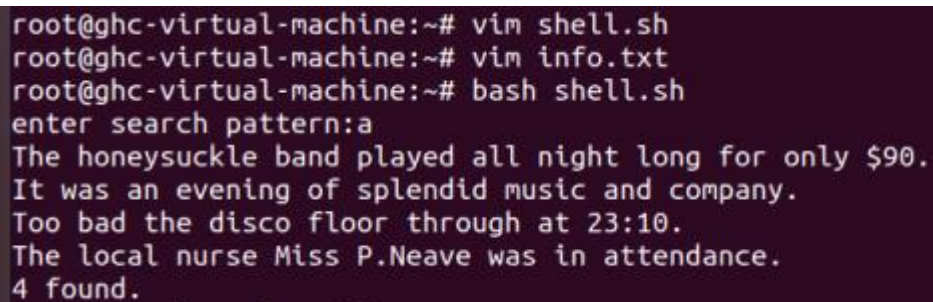
4. 脚本的内容如下所示：

```
#!/bin/bash
```

```
read -p "enter search pattern: " pattern
```

```
awk "/$pattern/" '{ nmatches++; print } END { print nmatches,  
"found." }' info.txt
```

试运行该脚本，并理解该脚本实现的功能。



```
root@ghc-virtual-machine:~# vim shell.sh  
root@ghc-virtual-machine:~# vim info.txt  
root@ghc-virtual-machine:~# bash shell.sh  
enter search pattern:a  
The honeysuckle band played all night long for only $90.  
It was an evening of splendid music and company.  
Too bad the disco floor through at 23:10.  
The local nurse Miss P.Neave was in attendance.  
4 found.
```

原语句有误，更改后可运行。功能为查找符合条件的行数量并显示数量。

四、 实验过程分析与讨论

在使用 sed 指令的过程中，因误用了单引号 ‘\$var’ 使得无论如何也无法显示所筛选的行内容，改用双引号后实验成功

akw 可使用复杂的条件语句，仍需加强练习

五、指导教师意见

指导教师签字：卢洋