

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 03 月 08 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

```
lizhe@LAPTOP-PQJNJ06U:~$ cd /etc
lizhe@LAPTOP-PQJNJ06U:/etc$ pwd
/etc
lizhe@LAPTOP-PQJNJ06U:/etc$ █
```

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

```
lizhe@LAPTOP-PQJNJ06U:/etc$ cd /home/lizhe
lizhe@LAPTOP-PQJNJ06U:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .cache  .config  .profile  a
```

3、使用命令创建目录/home/lizhe/linux，然后删除该目录。

```
lizhe@LAPTOP-PQJNJ06U:~$ mkdir /home/lizhe/linux
lizhe@LAPTOP-PQJNJ06U:~$ ls
a linux
lizhe@LAPTOP-PQJNJ06U:~$ rmdir linux
lizhe@LAPTOP-PQJNJ06U:~$ ls
a
lizhe@LAPTOP-PQJNJ06U:~$
```

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

```
lizhe@LAPTOP-PQJNJ06U:~$ cat > foo
Hello, Linux!
lizhe@LAPTOP-PQJNJ06U:~$ ls
a foo
lizhe@LAPTOP-PQJNJ06U:~$ cat foo
Hello, Linux!
lizhe@LAPTOP-PQJNJ06U:~$
```

5、使用命令创建目录 `/home/lizhe/foo.bak`，然后将 `/home/lizhe/foo` 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。

```

lizhe@LAPTOP-PQJNJ06U:~$ mkdir /home/lizhe/foo.bak
lizhe@LAPTOP-PQJNJ06U:~$ ls
a foo foo.bak
lizhe@LAPTOP-PQJNJ06U:~$ cp -r foo foo.bak
lizhe@LAPTOP-PQJNJ06U:~$ cd foo.bak
lizhe@LAPTOP-PQJNJ06U:~/foo.bak$ ls
foo
lizhe@LAPTOP-PQJNJ06U:~/foo.bak$ rm -i foo
rm: remove regular file 'foo'? y
lizhe@LAPTOP-PQJNJ06U:~/foo.bak$ cd..
cd..: command not found
lizhe@LAPTOP-PQJNJ06U:~/foo.bak$ rmdir foo.bak
rmdir: failed to remove 'foo.bak': No such file or directory
lizhe@LAPTOP-PQJNJ06U:~/foo.bak$ _

```

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。

```

lizhe@LAPTOP-PQJNJ06U:~$ head -3 /etc/adduser.conf
# /etc/adduser.conf: `adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentat
lizhe@LAPTOP-PQJNJ06U:~$ _

```

```

lizhe@LAPTOP-PQJNJ06U:~$ tail -5 /etc/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
lizhe@LAPTOP-PQJNJ06U:~$ _

```

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 GROUPTHOMES is "yes", then the home directories will be created as
# /home/groupname/user.
GROUPTHOMES=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.
--More-- (36%)
```

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

```
lizhe@LAPTOP-PQJNJ06U:~$ cat > /home/lizhe/bar.txt
google 110 5000
baidu 100 5000
guge 50 3000
sohu 100 4500
lizhe@LAPTOP-PQJNJ06U:~$
```

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

```
lizhe@LAPTOP-PQJNJ06U:~$ sort -r bar.txt
sohu 100 4500
guge 50 3000
google 110 5000
baidu 100 5000
lizhe@LAPTOP-PQJNJ06U:~$ _
```

(2) 按公司人数排序

```
lizhe@LAPTOP-PQJNJ06U:~$ sort -n -k2 bar.txt
guge 50 3000
baidu 100 5000
sohu 100 4500
google 110 5000
lizhe@LAPTOP-PQJNJ06U:~$ _
```

(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

```
lizhe@LAPTOP-PQJNJ06U:~$ sort -n -t ' ' -k3r -k2 bar.txt
google 110 5000
baidu 100 5000
guge 50 3000
sohu 100 4500
```


(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

```
lizhe@LAPTOP-PQJNJ06U: ~$ sort -n -t ' ' -k3r -k2 bar.txt
google 110 5000
baidu 100 5000
guge 50 3000
sohu 100 4500
```

(5) 从公司英文名称的第2个字母开始进行排序。

```
lizhe@LAPTOP-PQJNJ06U: ~$ sort -t ' ' -k1.2 bar.txt
baidu 100 5000
sohu 100 4500
google 110 5000
guge 50 3000
lizhe@LAPTOP-PQJNJ06U: ~$
```

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 03 月 08 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

五、 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：chomd。

六、 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。

七、 实验内容及结果

1. 查找指定文件

- (1) 在用户目录下新建目录 baz ， 在 baz 下新建文件 qux ， 并写如任意几行内容；

```
lizhe@LAPTOP-PQJNJ06U:~$ mkdir baz
lizhe@LAPTOP-PQJNJ06U:~$ cd baz/
lizhe@LAPTOP-PQJNJ06U:~/baz$ touch qux
lizhe@LAPTOP-PQJNJ06U:~/baz$ echo 'Hello, Linux' > qux
```

- (2) 在用户目录下查找文件 qux ， 并显示该文件位置信息；

```
lizhe@LAPTOP-PQJNJ06U:~$ find -name qux
./baz/qux
```

- (3) 统计文件 qux 中所包含内容的行数、字数和字节数；

```
lizhe@LAPTOP-PQJNJ06U:~/baz$ wc qux
1 1 12 qux
lizhe@LAPTOP-PQJNJ06U:~/baz$
```

- (4) 在用户目录下查找文件 qux ， 并删除该文件；

```
lizhe@LAPTOP-PQJNJ06U:~$ find -name "qux" | xargs rm -rf
```

(5) 查看文件夹 `baz` 内容，看一下是否删除了文件 `qux` 。

```
lizhe@LAPTOP-PQJNJ06U:~$ cd baz/  
lizhe@LAPTOP-PQJNJ06U:~/baz$ ls  
lizhe@LAPTOP-PQJNJ06U:~/baz$
```

2. 文件打包

(1) 在用户目录下新建文件夹 `path1`，在 `path1` 下新建文件 `file1` 和 `file2`；

```
lizhe@LAPTOP-PQJNJ06U:~$ mkdir path1  
lizhe@LAPTOP-PQJNJ06U:~$ mkdir path1/file1 path1/file2  
lizhe@LAPTOP-PQJNJ06U:~$ _
```

(2) 在用户目录下新建文件夹 `path2`，在 `path2` 下新建文件 `file3`；

```
lizhe@LAPTOP-PQJNJ06U:~$ mkdir path2  
lizhe@LAPTOP-PQJNJ06U:~$ mkdir path2/file3  
lizhe@LAPTOP-PQJNJ06U:~$
```

(3) 在用户目录下新建文件 `file4`；

```
lizhe@LAPTOP-PQJNJ06U:~$ mkdir file4  
lizhe@LAPTOP-PQJNJ06U:~$
```

(4) 在用户目录下对文件夹 `path1` 和 `file4` 进行打包，生成文件 `package.tar`；

```
lizhe@LAPTOP-PQJNJ06U:~$ tar -cvf package.tar file4 path1/  
file4/  
path1/  
path1/file1/  
path1/file2/
```

(5) 查看包 `package.tar` 的内容；

```
lizhe@LAPTOP-PQJNJ06U:~$ tar -tvf package.tar
drwxr-xr-x lizhe/lizhe      0 2023-05-12 10:44 file4/
drwxr-xr-x lizhe/lizhe      0 2023-05-12 10:43 path1/
drwxr-xr-x lizhe/lizhe      0 2023-05-12 10:43 path1/file1/
drwxr-xr-x lizhe/lizhe      0 2023-05-12 10:43 path1/file2/
lizhe@LAPTOP-PQJNJ06U:~$
```

(6) 向包 package.tar 里添加文件夹 path2 的内容;

```
lizhe@LAPTOP-PQJNJ06U:~$ tar -rvf package.tar path2/
path2/
path2/file3/
lizhe@LAPTOP-PQJNJ06U:~$
```

(7) 将包 package.tar 复制到用户目录下的新建文件夹 path3 中;

```
lizhe@LAPTOP-PQJNJ06U:~$ mkdir path3/
lizhe@LAPTOP-PQJNJ06U:~$ cp package.tar path3/
lizhe@LAPTOP-PQJNJ06U:~$
```

(8) 进入 path3 文件夹，并还原包 package.tar 的内容。

```
lizhe@LAPTOP-PQJNJ06U:~/path3$ tar -xvf package.tar
file4/
path1/
path1/file1/
path1/file2/
path2/
path2/file3/
lizhe@LAPTOP-PQJNJ06U:~/path3$ _
```

3. 符号链接内容

(1) 新建文件 foo.txt ， 内容为 123 ；

```
lizhe@LAPTOP-PQJNJ06U:~$ touch foo.txt
lizhe@LAPTOP-PQJNJ06U:~$ echo '123' > foo.txt
lizhe@LAPTOP-PQJNJ06U:~$
```

(2) 建立 foo.txt 的硬链接文件 bar.txt ， 并比较 bar.txt 的内容和 foo.txt 是否相同，要求用 comm 或 diff 命令；

```
lizhe@LAPTOP-PQJNJ06U:~$ ln foo.txt bar.txt
lizhe@LAPTOP-PQJNJ06U:~$ diff foo.txt bar.txt
```

(3) 查看 foo.txt 和 bar.txt 的 i 节点号 (inode) 是否相同;

```
lizhe@LAPTOP-PQJNJ06U:~$ ls -li
30399297484841077 a 155374187144375760 file4 7318349394629894 foo.txt 167196136166187703 path1
7318349394629894 bar.txt 1970324837735160 foo 21392098230199364 locate 59672695062757581 path2
7318349395036460 baz 7599824371948336 foo.bak 77968568548944758 package.tar 46724846133969569 path3
lizhe@LAPTOP-PQJNJ06U:~$
```

(4) 修改 bar.txt 的内容为 abc，然后通过命令判断 foo.txt 与 bar.txt 是否相同；

```
lizhe@LAPTOP-PQJNJ06U:~$ echo 'abc' > bar.txt
lizhe@LAPTOP-PQJNJ06U:~$ diff foo.txt bar.txt
```

(5) 删除 foo.txt 文件，然后查看 bar.txt 文件的 inode 及内容；

```
lizhe@LAPTOP-PQJNJ06U:~$ rm foo.txt
lizhe@LAPTOP-PQJNJ06U:~$ ls -li
30399297484841077 a 155374187144375760 file4 21392098230199364 locate 59672695062757581 path2
7318349394629894 bar.txt 1970324837735160 foo 77968568548944758 package.tar 46724846133969569 path3
7318349395036460 baz 7599824371948336 foo.bak 167196136166187703 path1
lizhe@LAPTOP-PQJNJ06U:~$
```

(6) 创建文件 bar.txt 的符号链接文件 baz.txt，然后查看 bar.txt 和 baz.txt 的 inode 号，并观察两者是否相同，比较 bar.txt 和 baz.txt 的文件内容是否相同；

```
lizhe@LAPTOP-PQJNJ06U:~$ ln -s bar.txt baz.txt
lizhe@LAPTOP-PQJNJ06U:~$ ls -li
30399297484841077 a 9288674231876993 baz.txt 7599824371948336 foo.bak 167196136166187703 path1
7318349394629894 bar.txt 155374187144375760 file4 21392098230199364 locate 59672695062757581 path2
7318349395036460 baz 1970324837735160 foo 77968568548944758 package.tar 46724846133969569 path3
lizhe@LAPTOP-PQJNJ06U:~$ diff bar.txt baz.txt
lizhe@LAPTOP-PQJNJ06U:~$
```

(7) 删除 bar.txt，查看文件 baz.txt，观察系统给出什么提示信息。

```
lizhe@LAPTOP-PQJNJ06U:~$ rm bar.txt
lizhe@LAPTOP-PQJNJ06U:~$ cat baz.txt
cat: baz.txt: No such file or directory
lizhe@LAPTOP-PQJNJ06U:~$
```

4. 权限管理

(1) 新建文件 qux.txt；

```
lizhe@LAPTOP-PQJNJ06U:~$ touch qux.txt
```

(2) 为文件 qux.txt 增加执行权限（所有用户都可以执行）。

```
lizhe@LAPTOP-PQJNJ06U: ~$ chmod a+x qux.txt
lizhe@LAPTOP-PQJNJ06U: ~$
```

八、 实验过程分析与讨论

在进行权限管理的实验时，忘记权限增加的具体命令，后经过查询弄明白后才正确完成实验内容。。

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 3 月 22 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

九、 实验目的

掌握 vim 编辑器及 gcc 编译器的使用方法。

十、 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。

十一、 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 ；

```
lizhe@LAPTOP-PQJNJ06U:~$ mkdir workspace1
lizhe@LAPTOP-PQJNJ06U:~$ ls
workspace1
```

- (2) 进入目录 workspace1 ；

```
lizhe@LAPTOP-PQJNJ06U:~$ cd workspace1
lizhe@LAPTOP-PQJNJ06U:~/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;  
  
}
```

```
lizhe@LAPTOP-PQJNJ06U: ~/workspace1$ vim test.c
```

```
#include<stdio.h>  
  
int main()  
{  
    printf("hello world!\n");  
    return 0;  
}
```

(4) 保存 test.c 的内容，并退出；

```
~  
:wq_
```

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

```
lizhe@LAPTOP-PQJNJ06U: ~/workspace1$ gcc test.c -o test  
lizhe@LAPTOP-PQJNJ06U: ~/workspace1$ ./test  
hello world!  
lizhe@LAPTOP-PQJNJ06U: ~/workspace1$ _
```

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；

```
lizhe@LAPTOP-PQJNJ06U: ~$ mkdir workspace2  
lizhe@LAPTOP-PQJNJ06U: ~$ ls  
workspace1 workspace2  
lizhe@LAPTOP-PQJNJ06U: ~$
```

(2) 进入 workspace2 目录；

```
lizhe@LAPTOP-PQJNJ06U: ~$ cd workspace2  
lizhe@LAPTOP-PQJNJ06U: ~/workspace2$
```

(3) 使用以下命令：

```
cat /etc/gai.conf > ./gai.conf
```

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；

```
lizhe@LAPTOP-PQJNJ06U:~/workspace2$ cat /etc/gai.conf > ./gai.conf
lizhe@LAPTOP-PQJNJ06U:~/workspace2$
```

(4) 使用 `vim` 编辑当前目录下的 `gai.conf` ；

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

(5) 将光标移到第 18 行；

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

(6) 复制该行内容；

yy

(7) 将光标移到最后一行行首；

```
#precedence ::1/128      50
#precedence ::/0         40
#precedence 2002::/16    30
#precedence ::/96        20
#precedence ::ffff:0:0/96 10
#
#     For sites which prefer IPv4 connections change the last line to
#
#precedence ::ffff:0:0/96 100
#
# scopev4 <mask> <value>
#     Add another rule to the RFC 6724 scope table for IPv4 addresses.
#     By default the scope IDs described in section 3.2 in RFC 6724 are
#     used. Changing these defaults should hardly ever be necessary.
#     The defaults are equivalent to:
#
#scopev4 ::ffff:169.254.0.0/112 2
#scopev4 ::ffff:127.0.0.0/104    2
#scopev4 ::ffff:0.0.0.0/96      14
```

(8) 粘贴复制行的内容；

```
# label <mask> <value>
# label <mask> <value>
```

(9) 撤销第 8 步的动作；

```
# label <mask> <value>
# Add another rule to the RFC 3484 label table. See section
# RFC 3484. The default is:
```

(10) 存盘但不退出;

```
#scopev4 ::ffff:0.0.0.0/96 14
:w
```

(11) 将光标移到首行;

```
# Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address sorting is needed.
# RFC 3484 governs the sorting. But the RFC also says that system
# administrators should be able to overwrite the defaults. This can be
# achieved here.
#
# All lines have an initial identifier specifying the option followed by
# up to two values. Information specified in this file replaces the
# default information. Complete absence of data of one kind causes the
# appropriate default information to be used. The supported commands include:
#
# reload <yes|no>
# If set to yes, each getaddrinfo(3) call will check whether this file
# changed and if necessary reload. This option should not really be
# used. There are possible runtime problems. The default is no.
#
# label <mask> <value>
# Add another rule to the RFC 3484 label table. See section 2.1 in
# RFC 3484. The default is:
```

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

```
#scopev4 ::ffff:0.0.0.0/96 14
#scopev4 ::ffff:0.0.0.0/96 14
#Hello, this is vim world!
```

(13) 删除字符串 "this" ;

```
#Hello, is vim world!
```

(14) 强制退出 vim , 不存盘

```
#Hello, is vim world!
:q!
```


十二、 在进行 gcc 命令使用时提示没有 gcc 指令，用 `sudo apt` 语句下载 gcc 后成功实验

十三、 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	实验四 用户和用户组管理		
实验教室	丹青 922	实验日期	2023 年 3 月 28 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

十四、 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` 、 `groupmod` 、 `gpasswd` ；
3. 掌握用户和用户组维护命令，包括命令 `passwd` 、 `su` 、 `sudo` 。

十五、 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。

十六、 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@LAPTOP-PQJNJ06U:/home/lizhe# useradd foo -s /bin/sh -b /home/foo -c bar -p 123456
root@LAPTOP-PQJNJ06U:/home/lizhe#
```

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@LAPTOP-PQJNJ06U:/home/lizhe# usermod foo -u 2000 -s /bin.csh
root@LAPTOP-PQJNJ06U:/home/lizhe# _
```

3. 从用户 `foo` 切换到 `root` ；

```
su root
```

```
root@LAPTOP-PQJNJ06U:/home/lizhe#
```

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@LAPTOP-PQJNJ06U:/home/lizhe# userdel foo -r
userdel: foo mail spool (/var/mail/foo) not found
userdel: foo home directory (/home/foo/foo) not found
root@LAPTOP-PQJNJ06U:/home/lizhe# _
```

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@LAPTOP-PQJNJ06U:/home/lizhe# #newuser < u.txt
```

```
root@LAPTOP-PQJNJ06U:/home/lizhe# cat userpasswd | chpasswd
```

```

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
messagebus:x:102:105:/:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:103:106:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
syslog:x:104:111:/:/home/syslog:/usr/sbin/nologin
_apt:x:105:65534:/:/nonexistent:/usr/sbin/nologin
uuidd:x:106:112:/:/run/uuidd:/usr/sbin/nologin

```

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

```

root@LAPTOP-PQJNJ06U:/home/lizhe# groupadd group1 -g 3000
root@LAPTOP-PQJNJ06U:/home/lizhe#

```

7. 在用户组 `group1` 中添加两个之前批量创建的用户；

```

root@LAPTOP-PQJNJ06U:/home/lizhe# usermod -g group1 user1
root@LAPTOP-PQJNJ06U:/home/lizhe# usermod -g group1 user2

```

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@LAPTOP-PQJNJ06U:/home/lizhe# sudo vi /etc/shadow
root@LAPTOP-PQJNJ06U:/home/lizhe# sudo vi /etc/sudoers

```

十七、 实验过程分析与讨论

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时遇到问题，经过查询后得到解决。

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 5 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

十八、 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。

十九、 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。

二十、 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）；

```
lizhe@LAPTOP-PQJNJ06U:~$ foo=200
lizhe@LAPTOP-PQJNJ06U:~$ echo foo
foo
lizhe@LAPTOP-PQJNJ06U:~$ echo $foo
200
lizhe@LAPTOP-PQJNJ06U:~$ _
```

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）；

```
lizhe@LAPTOP-PQJNJ06U:~$ bar=100
lizhe@LAPTOP-PQJNJ06U:~$ test $bar -gt 150
lizhe@LAPTOP-PQJNJ06U:~$ echo $?
1
lizhe@LAPTOP-PQJNJ06U:~$
```

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`）；

```
#!/bin/bash
#name KPL
#time 2023.5.13
#function test
#version 1.0

echo $(hostname)
echo $(date)
```

```
lizhe@LAPTOP-PQJNJ06U:~$ vi test.sh
lizhe@LAPTOP-PQJNJ06U:~$ sh test.sh
LAPTOP-PQJNJ06U
Sat May 13 10:10:33 CST 2023
```

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
echo "total paramter are: $#"
```

```
test $# -eq 0 && echo "You don't give one paramter at least" && exit 0
```

```
for var in $@
do
    echo "the num is: $var"
    var0=$var
    var1=$((($var0/100))
    var0=$((($var0%100))
    var2=$((($var0/10))
    var3=$((($var0%10))
    if [ $((($var1*$var1*$var1+$var2*$var2*$var2+$var3*$var3*$var3)) -eq $var );then
        echo "$var is shuixianhua num!"
    else
        echo "$var is not a shuixianhua nhum."
    fi
done
```

```
lizhe@LAPTOP-PQJNJ06U:~$ sh shuixian.sh 153
total paramter are: 1
the num is: 153
shuixian.sh: 14: echo153 is shuixianhua num!
```

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/sh
a=$((($1+$2+$3))
echo "jiegua is $a"
```

```
lizhe@LAPTOP-PQJNJ06U:~$ vi jiegua.sh
lizhe@LAPTOP-PQJNJ06U:~$ sh jiegua.sh 1 2 3
jiegua is 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
echo "the parmater num are: $#"
```

```
for var in $@
do
    if [ $var -ge 90 ];then
        echo "A"
    elif [ "$var" -ge 80 -a "$var" -lt 90 ];then
        echo "B"
    elif [ "$var" -ge 70 -a "$var" -lt 80 ];then
        echo "C"
    elif [ "$var" -ge 60 -a "$var" -lt 70 ];then
        echo "D"
    else
        echo "E"
    fi
done
```

```
lizhe@LAPTOP-PQJNJ06U:~$ sh chengji.sh 91
the parmater num are: 1
A
```

二十一、 实验过程分析与讨论

编写 shell 程序时要用 `vim` 命令写入 shell 程序，然后用 `sh` 命令加上输入再输出结果。

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 5 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

二十二、 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 。

二十三、 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。

二十四、 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。

```
#!/bin/bash
read -p "please input a file name: "file
filelist=$(ls $file/*.sh)
echo "$filelist"
for filename in $filelist
do
    echo "$filename"
    if [ ! -x "$filename" ];then
        eval "chmod a+x $filename"
        eval "cp $filename movedir"
    fi
done
```

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

```
#!/bin/bash
i=0
sum=0
while [ "$i" -lt 20 ]
do
    if [ $((i%2)) -eq 0 ];then
        sum=$((sum+i))
    fi
    i=$((i+1))
done
echo "$sum"
```

```
lizhe@LAPTOP-PQJNJ06U: ~$ sh s.sh
90
```

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

```
#!/bin/bash
i=1
sum=0
until [ "$i" -gt 10 ]
do
    sum=$((sum+i*i))
    i=$((i+1))
done
echo "$sum"
```

```
lizhe@LAPTOP-PQJNJ06U: ~$ sh p.sh
385
lizhe@LAPTOP-PQJNJ06U: ~$
```

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` 、 `break2` 、 `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
```

```
---
```

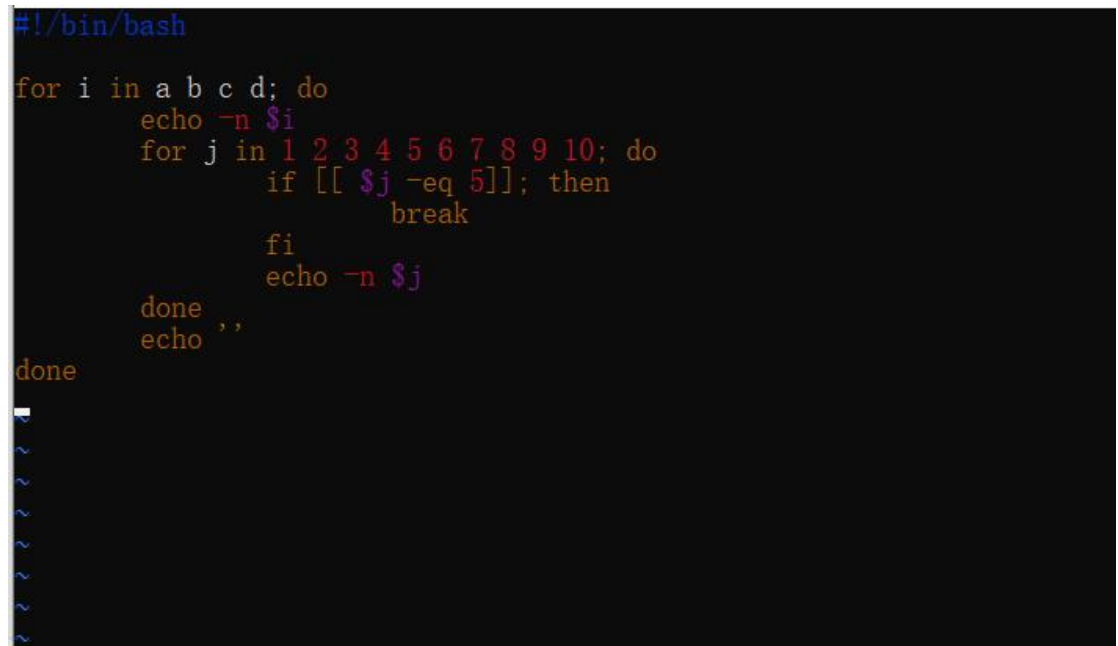
```
fi
```

```
echo -n $j
```

```
done
```

```
echo "
```

```
done
```



```
#!/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
```

The image shows a terminal window with a black background and syntax-highlighted text. The text represents a shell script with nested loops. The first loop iterates over 'a b c d'. Inside it, a second loop iterates over '1 2 3 4 5 6 7 8 9 10'. The inner loop has a conditional 'break' statement when 'j' equals 5. The script prints the characters of 'i' followed by the characters of 'j' separated by a comma and a space. The terminal shows the first few lines of the script being executed, with the output 'a, ' visible on the line following the 'done' of the inner loop.

```
#!/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

~
~
~
```

```
#!/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

~
~
~
~
~
~
~
~
~
~
~
```

```
#!/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
```

二十五、 实验过程分析与讨论

1、for 循环

(1) for 循环有三种结构：一种是列表 for 循环，第二种是不带列表 for 循环。第三种是类 C 风格的 for 循环。

while 循环

也称为前测试循环语句，重复次数是利用一个条件来控制是否继续重复执行这个语句。为了避免死循环，必须保证循环体中包含循环出口条件即表达式存在退出状态为非 0 的情况。

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 19 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

二十六、 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。

二十七、 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。

二十八、 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；

```
#!/bin/bash
sum() {
    return $((($1+$2))
}
sum $1 $2
echo $?
```

```
lizhe@LAPTOP-PQJNJ06U:~$ sh s1.sh 1 2
3
```

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

jiecheng() {
    local i=1
    local mul=1
    while [ $i -le $n ]
    do
        mul=$((i*$mul))
        i=$((i+1))
    done
    return $mul
}

n=$1

jiecheng $n
echo "$?"
```

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
```

```
start...
this is the first
this is the second
-----this is third
```

二十九、 实验过程分析与讨论

```
function name() {
```

```
statements
```

```
[return value]
```

```
}
```

对各个部分的说明：

function 是 Shell 中的关键字，专门用来定义函数；

name 是函数名；

statements 是函数要执行的代码，也就是一组语句；

return value 表示函数的返回值，其中 return 是 Shell 关键字，专门

用在函数中返回一个值；这一部分可以写也可以不写。

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 19 日
学 号	2021211575	姓 名	李哲
专业班级	计算机科学与技术 01 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

三十、 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。

三十一、 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。

三十二、 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；

```
lizhe@LAPTOP-PQJNJ06U:~$ 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.
lizhe@LAPTOP-PQJNJ06U:~$ cat quote.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
lizhe@LAPTOP-PQJNJ06U:~$
```

(2) 显示包含 music 文字的行内容及行号；

```
lizhe@LAPTOP-PQJNJ06U:~$ cat quote.txt | sed -n '/music/p'
It was an evening of splendid music and company.
lizhe@LAPTOP-PQJNJ06U:~$
```

(3) 在第 4 行后面追加内容: "hello world!" ;

```
lizhe@LAPTOP-PQJNJ06U:~$ 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 fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
hello world
```

(4) 将文本 "The" 替换为 "Quod" ;

```
lizhe@LAPTOP-PQJNJ06U:~$ 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 fell 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." ;

```
lizhe@LAPTOP-PQJNJ06U:~$ cat quote.txt | sed '2c This is the third line'
The honeysuckle band played all night long for only $90.
This is the third line
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
lizhe@LAPTOP-PQJNJ06U:~$
```

(6) 删除第 2 行内容；

```
lizhe@LAPTOP-PQJNJ06U:~$ cat quote.txt | sed '2d'
he honeysuckle band played all night long for only $90.
oo bad the disco floor fell through at 23:10.
he local nurse Miss P.Neave was in attendance.
lizhe@LAPTOP-PQJNJ06U:~$
```

(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。

```
lizhe@LAPTOP-PQJNJ06U: ~$ 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；

```
lizhe@LAPTOP-PQJNJ06U: ~$ cat numbers.txt | awk '{FS=":"}{print $2}'
:
five
lizhe@LAPTOP-PQJNJ06U: ~$ cat numbers.txt | awk '{FS=" "}{print $2}'
:
:
:
lizhe@LAPTOP-PQJNJ06U: ~$
```

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。

four : five : six 例如： `foo.txt` 内容为：

2 4 3

15 46 79

则输出为：

even:

2

4

46

numbers:

3

```
lizhe@LAPTOP-PQJNJ06U:~$ cat foo.txt | awk 'BEGIN{sum=0} {for(i=1;i<=NF;i++){if($i%2==0){printf $i;sum+=1}} }END{print sum}'  
24463
```

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

```
lizhe@LAPTOP-PQJNJ06U:~$ cat s4.sh  
#!/bin/bash  
  
read -p "enter search pattern: " pattern  
  
awk "/$pattern/" '{ nmatches++; print } END { print nmatches, "found." }' info.txt
```


三十三、 实验过程分析与讨论

1. 常见的 sed 命令选项包含以下几种：
2. -e 或-expression=: 表示用指定命令或者脚本来处理输入的文本文件
3. -f 或-file=: 表示用指定的脚本文件来处理输入的文件文件
4. -h 或--help: 显示帮助
5. -n、-quite 或 silent: 表示仅表示处理后的结果
6. -i: 直接编辑文本文件

。

五、指导教师意见

指导教师签字：卢洋