

# 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 5 月 22 日
学 号	2021213037	姓 名	白明予
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

```
jade@550: /etc

jade@550:/etc$ cd
jade@550:~$ cd /etc
jade@550:/etc$ pwd
/etc
jade@550:/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。

```
jade@550:/etc$ cd ../home
jade@550:/home$ ls
jade  lost+found
jade@550:/home$ cd jade
jade@550:~$ ls -a
.          桌面          .mozilla
..         .bash_history .profile
公共的    .bash_logout snap
模板      .bashrc      .ssh
实验报告模板.doc .cache       .sudo_as_admin_successful
视频      .config      .SWM
图片      .gnupg       .SWN
文档      linuxtest    .SWO
下载      .local       .SWP
音乐      ~/.lock.实验报告模板.doc# .viminfo
```

3、使用命令创建目录 /home/lyj/linux , 然后删除该目录。

```
jade@550: ~  
jade@550:~$ ls  
公共的 实验报告模板.doc 图片 下载 桌面 snap  
模板 视频 文档 音乐 linuxtest  
jade@550:~$ mkdir linux  
jade@550:~$ ls  
公共的 实验报告模板.doc 图片 下载 桌面 linuxtest  
模板 视频 文档 音乐 linux snap  
jade@550:~$ rm linux  
rm: 无法删除 'linux': 是一个目录  
jade@550:~$ rmdir linux  
jade@550:~$ ls  
公共的 实验报告模板.doc 图片 下载 桌面 snap  
模板 视频 文档 音乐 linuxtest  
jade@550:~$
```

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

```
jade@550: ~  
jade@550:~$ cat > abc  
hello linux  
^C  
jade@550:~$ ls  
公共的 实验报告模板.doc 图片 下载 桌面 linuxtest  
模板 视频 文档 音乐 abc snap  
jade@550:~$ cat abc  
hello linux  
jade@550:~$
```

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

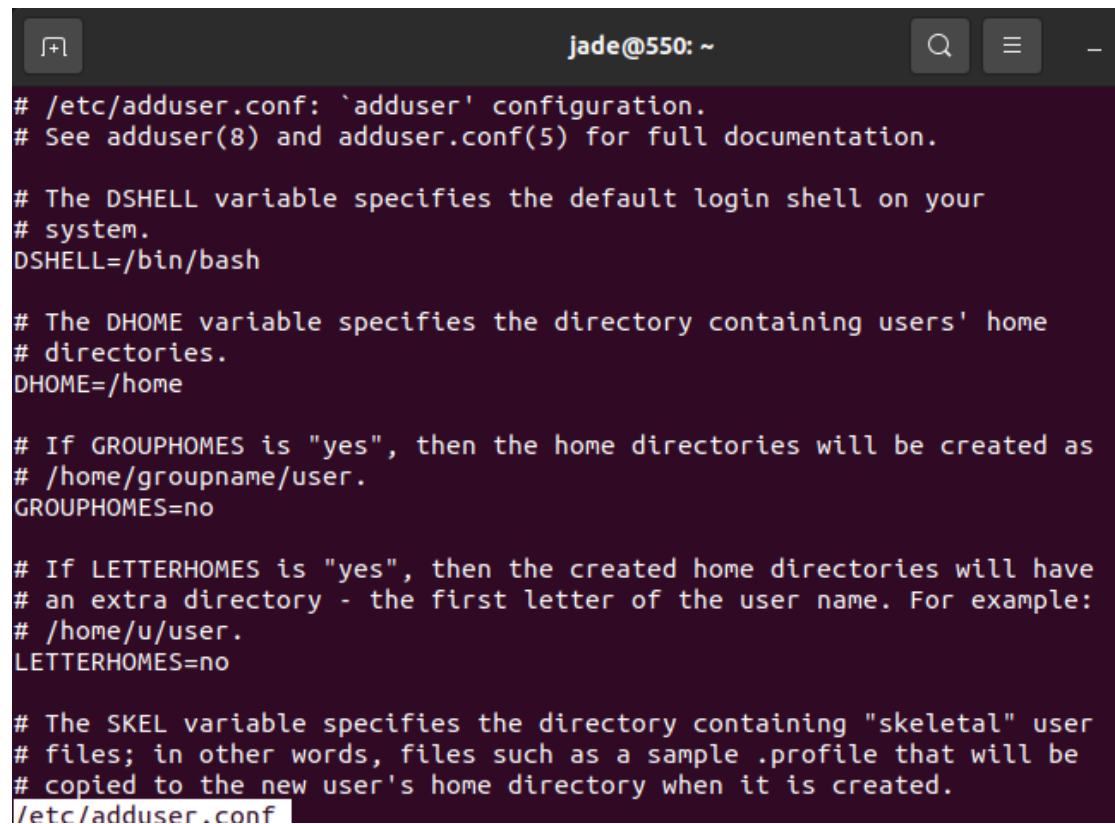
除。

```
jade@550: ~  
jade@550:~$ mkdir ak  
jade@550:~$ ls  
公共的 实验报告模板.doc 图片 下载 桌面 ak snap  
模板 视频 文档 音乐 abc linuxtest  
jade@550:~$ cp -r abc ak  
jade@550:~$ ls  
公共的 实验报告模板.doc 图片 下载 桌面 ak snap  
模板 视频 文档 音乐 abc linuxtest  
jade@550:~$ cd ak  
jade@550:~/ak$ ls  
abc  
jade@550:~/ak$ rm -i abc  
rm: 是否删除普通文件 'abc'? y  
jade@550:~/ak$ cd ..  
jade@550:~$ rmdir ak  
  
jade@550:~$ rmdir ak  
jade@550:~$
```

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。

```
jade@550: ~  
jade@550:~$ head -3 /etc/adduser.conf  
# /etc/adduser.conf: `adduser' configuration.  
# See adduser(8) and adduser.conf(5) for full documentation.  
  
jade@550:~$ 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  
jade@550:~$
```

## 7、分屏查看文件 /etc/adduser.conf 的内容。

A terminal window with a dark background and light-colored text. The title bar shows 'jade@550: ~'. The terminal displays the contents of the file /etc/adduser.conf. The text is as follows:

```
# /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.
/etc/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

```
jade@550: ~  
jade@550:~$ cat >facebook.txt  
goole 110 5000  
baidu 100 5000  
guge 50 3000  
sohu 100 4500  
^C  
jade@550:~$ cat facebook.txt  
goole 110 5000  
baidu 100 5000  
guge 50 3000  
sohu 100 4500  
jade@550:~$
```

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
- (2) 按公司人数排序
- (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
- (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
- (5) 从公司英文名称的第 2 个字母开始进行排序。

```
jade@550: ~  
jade@550:~$ sort facebook.txt  
baidu 100 5000  
goole 110 5000  
guge 50 3000  
sohu 100 4500  
jade@550:~$
```

```
jade@550:~$ sort -n -k2 facebook.txt  
guge 50 3000  
baidu 100 5000  
sohu 100 4500  
goole 110 5000  
jade@550:~$
```

```
jade@550:~$ sort -n -k2 -k3 facebook.txt  
guge 50 3000  
sohu 100 4500  
baidu 100 5000  
goole 110 5000  
jade@550:~$
```

```
jade@550:~$ sort -t ' ' -k1.2 facebook.txt  
baidu 100 5000  
sohu 100 4500  
goole 110 5000  
guge 50 3000  
jade@550:~$
```



#### 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 5 月 22 日
学 号	2021213037	姓 名	白明予
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) ubuntu 20.04 LTS 系统测试版。

(3) libreoffice。

### 三、实验内容及结果

#### 1、查找指定文件

(1) 在用户主目录下新建目录 baz，在 baz 下新建文件 qu  
x，内容随意写几行。




```
jade@550:~$ mkdir baz
jade@550:~$ cd baz
jade@550:~/baz$ cat > qux
baibai
jade@550:~/baz$
```

（2）在用户主目录下查找文件 `qux`，并显示该文件位置信息。

```
jade@550:~/baz$ find ../ -name "qux"
../baz/qux
jade@550:~/baz$
```

（3）统计 `qux` 文件中所包含的行数、字数和字节数。



```
jade@550: ~/baz
jade@550:~/baz$ wc -l qux
1 qux
jade@550:~/baz$ wc -w qux
1 qux
jade@550:~/baz$ wc -c qux
7 qux
jade@550:~/baz$
```

（4）在用户主目录下查找文件 `qux`，并删除该文件。

```
jade@550: ~  
jade@550:~$ find -name "qux"  
./baz/qux  
jade@550:~$ rm -rf /baz/qux  
jade@550:~$
```

(5) 查看文件夹 baz 内容，看一下是否删除了文件 qux。

```
jade@550: ~/baz  
jade@550:~$ rm ./baz/qux  
jade@550:~$ cd baz  
jade@550:~/baz$ cat qux  
cat: qux: 没有那个文件或目录  
jade@550:~/baz$
```

## 2、文件打包

(1) 在用户主目录下新建文件夹 path1，在 path1 下新建文件 file1 和 file2。

```
jade@550: ~/path1  
jade@550:~/baz$ cd  
jade@550:~$ mkdir path1  
jade@550:~$ cd path1  
jade@550:~/path1$ touch file1  
jade@550:~/path1$ touch file2  
jade@550:~/path1$ ls  
file1 file2  
jade@550:~/path1$
```

(2) 在用户主目录下新建文件夹 path2，在 path2 下新建文件 file3。

```
jade@550: ~/path1/path2

jade@550:~/path1$ mkdir path2
jade@550:~/path1$ cd path2
jade@550:~/path1/path2$ touch file3
jade@550:~/path1/path2$ ls
file3
jade@550:~/path1/path2$
```

(3) 在用户主目录下新建文件 file4。

```
jade@550:~/path1/path2$ cd
jade@550:~$ touch file4
jade@550:~$
```

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

```
jade@550:~$ tar -cvf package.tar path1 file4
path1/
path1/file2
path1/file1
path1/path2/
path1/path2/file3
file4
jade@550:~$
```

(5) 查看包 package.tar 的内容。

```
jade@550:~$ tar -tvf package.tar
drwxrwxr-x  jade/jade      0 2023-05-23 17:17 path1/
-rw-rw-r--  jade/jade      0 2023-05-23 17:16 path1/file2
-rw-rw-r--  jade/jade      0 2023-05-23 17:16 path1/file1
drwxrwxr-x  jade/jade      0 2023-05-23 17:17 path1/path2/
-rw-rw-r--  jade/jade      0 2023-05-23 17:17 path1/path2/file3
-rw-rw-r--  jade/jade      0 2023-05-23 17:18 file4
jade@550:~$
```

（6）向包 package.tar 里添加文件夹 path2 的内容。

```
jade@550:~$ tar -rvf package.tar ./path1/path2
./path1/path2/
./path1/path2/file3
jade@550:~$
```

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

```
jade@550: ~
jade@550:~$ mkdir path3
jade@550:~$ cp package.tar path3
jade@550:~$
```

（8）进入 path3 文件夹，并还原包 package.tar 的内容。

```
jade@550: ~/path3

jade@550:~$ cd path3
jade@550:~/path3$ ls
package.tar
jade@550:~/path3$ tar -xvf package.tar
path1/
path1/file2
path1/file1
path1/path2/
path1/path2/file3
file4
./path1/path2/
./path1/path2/file3
jade@550:~/path3$
```

### 3、符号链接内容

(1) 新建文件 foo.txt, 内容为 123。

```
jade@550: ~

jade@550:~$ cat > foo.txt
123
jade@550:~$
```

(2) 建立 foo.txt 得硬链接文件 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。

```
jade@550: ~

jade@550:~$ ln foo.txt bar.txt
ln: 无法创建硬链接 'bar.txt': 文件已存在
jade@550:~$ diff foo.txt bar.txt
jade@550:~$
```



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

```
jade@550:~$ ls -li foo.txt bar.txt
791388 bar.txt 791388 foo.txt
jade@550:~$
```

(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。

```
jade@550: ~
jade@550:~$ cat > bar.txt
abc
jade@550:~$ diff foo.txt bar.txt
jade@550:~$ cat foo.txt
abc
jade@550:~$
```

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

```
jade@550: ~
jade@550:~$ rm foo.txt
jade@550:~$ cat bar.txt
abc
jade@550:~$ ls -li bar.txt
791388 bar.txt
jade@550:~$
```

(6) 建立文件 bar.txt 的符号链接文件 baz.txt，然后查看 bar.txt 和 baz.txt 的 inode 号，观察两者是否相同，比较 b

ar.txt 和 baz.txt 的文件内容是否相同。

```
jade@550: ~  
jade@550:~$ ln -s bar.txt baz.txt  
jade@550:~$ ls -li bar.txt baz.txt  
791388 bar.txt 808890 baz.txt  
jade@550:~$ diff bar.txt baz.txt  
jade@550:~$
```

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

```
jade@550:~$ rm bar.txt  
jade@550:~$ ls  
公共的 实验报告 视频 文档 音乐 baz.txt linuxtest  
模板 实验报告模板.doc 图片 下载 桌面 facebook.txt snap  
jade@550:~$ cat baz.txt  
cat: baz.txt: 没有那个文件或目录  
jade@550:~$
```

#### 4、权限管理

(1) 新建文件 qux.txt

```
jade@550: ~  
jade@550:~$ touch qux.txt
```

(2) 为文件 qux.txt 增加执行权限（所有用户都可以执行）

```
jade@550: ~  
jade@550:~$ touch qux.txt  
jade@550:~$ chmod a+x qux.txt  
jade@550:~$ ls -l qux.txt  
-rwxrwxr-x 1 jade jade 0 5月 23 17:32 qux.txt  
jade@550:~$
```

#### 四、实验过程分析与讨论

遇到的困难在最后那个实验，chmod 部分，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 5 月 22 日
学 号	2021213037	姓 名	白明予
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

vim 编辑器及 gcc 编译器的使用

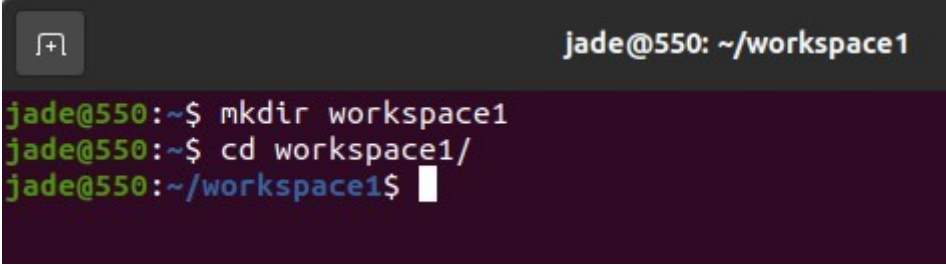
## 二、实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) ubuntu 20.04 LTS 系统测试版。
- (3) libreoffice。

## 三、实验内容及结果

### 1、vim 编辑器和 gcc 编译器的简单使用

(1) 在用户主目录下新建一个目录，命名为 workspace1



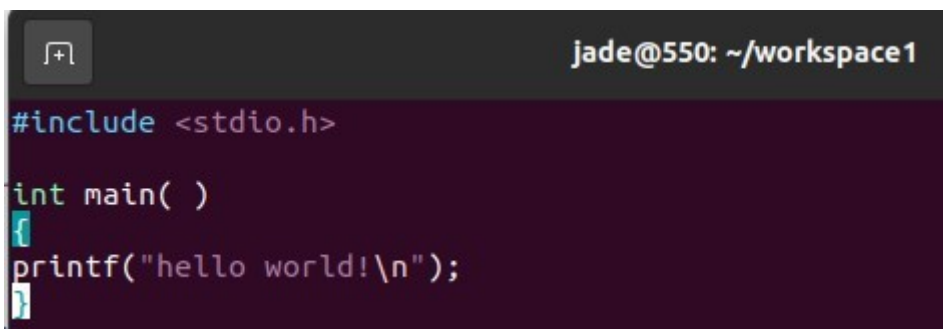
```
jade@550: ~/workspace1  
jade@550:~$ mkdir workspace1  
jade@550:~$ cd workspace1/  
jade@550:~/workspace1$
```

(2) 进入目录 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

test.c 文件内容为:

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

A screenshot of a terminal window with a dark background. The window title is 'jade@550: ~/workspace1'. The terminal shows the following C code:

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

```
}
```

(4) 保存 test.c 的内容, 并退出



(2) 进入 workspace2 目录。

(3) 使用命令 cat

```
jade@550: ~/workspace2

jade@550:~/workspace1$ cd
jade@550:~$ mkdir workspace2
jade@550:~$ cd workspace2
jade@550:~/workspace2$ cat /etcgai.conf > ./gai.conf
cat: /etcgai.conf: 没有那个文件或目录
jade@550:~/workspace2$ cat /etc/gai.conf > ./gai.conf
jade@550:~/workspace2$
```

(4) 使用 vim 编辑当前目录下的 gai.conf。

```
jade@550: ~/workspace2

Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address
# RFC 3484 governs the sorting. But the RFC also says
# administrators should be able to overwrite the default
# achieved here.
#
# All lines have an initial identifier specifying the
# up to two values. Information specified in this file
# default information. Complete absence of data of or
# appropriate default information to be used. The sup
#
# reload <yes|no>
#   If set to yes, each getaddrinfo(3) call will check
#   changed and if necessary reload. This option should
#   used. There are possible runtime problems. The
#
# label <mask> <value>
#   Add another rule to the RFC 3484 label table. See
#   RFC 3484. The default is:
#
#label ::1/128      0
#label ::/0         1
#label 2002::/16    2
"gai.conf" 65L, 2584C
```



( 5 ) 将光标移到第 18 行。



```
jade@550: ~/workspace2
# Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address sorting is
# 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
# up to two values. Information specified in this file replaces the
# default information. Complete absence of data of one kind causes
# appropriate default information to be used. The supported commands
# are:
# reload <yes|no>
#   If set to yes, each getaddrinfo(3) call will check whether this
#   configuration has 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
#   of RFC 3484. The default is:
#
#label ::1/128      0
#label ::/0         1
#label 2002::/16    2
/b
```

命令行模式输入 18gg

( 6 ) 复制该行内容。

```
jade@550: ~/workspace2

# Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address sorting is
# 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
# up to two values. Information specified in this file replaces the
# default information. Complete absence of data of one kind causes
# appropriate default information to be used. The supported commands
# are:
# reload <yes|no>
#   If set to yes, each getaddrinfo(3) call will check whether the
#   configuration has 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
#   of RFC 3484. The default is:
#
#label ::1/128      0
#label ::/0         1
#label 2002::/16    2
#
/b
```

命令行模式输入 y\$

( 7 ) 将光标移到最后一行行首。

```
jade@550: ~/workspace2

# precedence <mask> <value>
#   Add another rule to the RFC 3484 precedence table.  See section 10.2
#   and 10.3 in RFC 3484.  The default is:
#
#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
```

命令行模式输入 G

( 8 ) 粘贴复制行的内容。

```
jade@550: ~/workspace2

#   Add another rule to the RFC 3484 precedence table.  See
#   and 10.3 in RFC 3484.  The default is:
#
#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
#   precedence value to 100.
#precedence  ::ffff:0:0/96    100
#
# scopev4  <mask>  <value>
#   Add another rule to the RFC 6724 scope table for IPv4
#   By default the scope IDs described in section 3.2 in
#   used.  Changing these defaults should hardly ever be
#   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
# label  <mask>  <value>
```

命令行模式输入 p

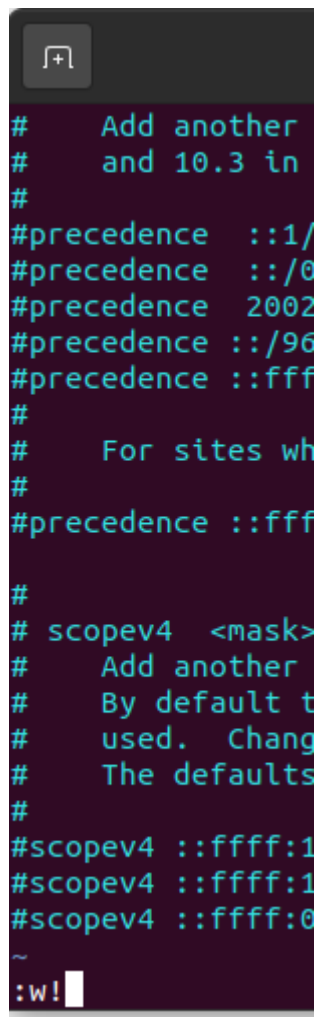
(9) 销第 8 步的动作。

```
jade@550: ~/workspace2

#   Add another rule to the RFC 3484 precedence table
#   and 10.3 in RFC 3484.  The default is:
#
#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
#   precedence of the ::ffff:0:0/96 rule to 100
#precedence  ::ffff:0:0/96 100
#
# scopev4 <mask> <value>
#   Add another rule to the RFC 6724 scope table for
#   By default the scope IDs described in section 3.
#   used.  Changing these defaults should hardly ever
#   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
~
1 行被去掉; before #1 28 秒前
```

命令行模式输入 u

( 10 ) 存盘但不退出。

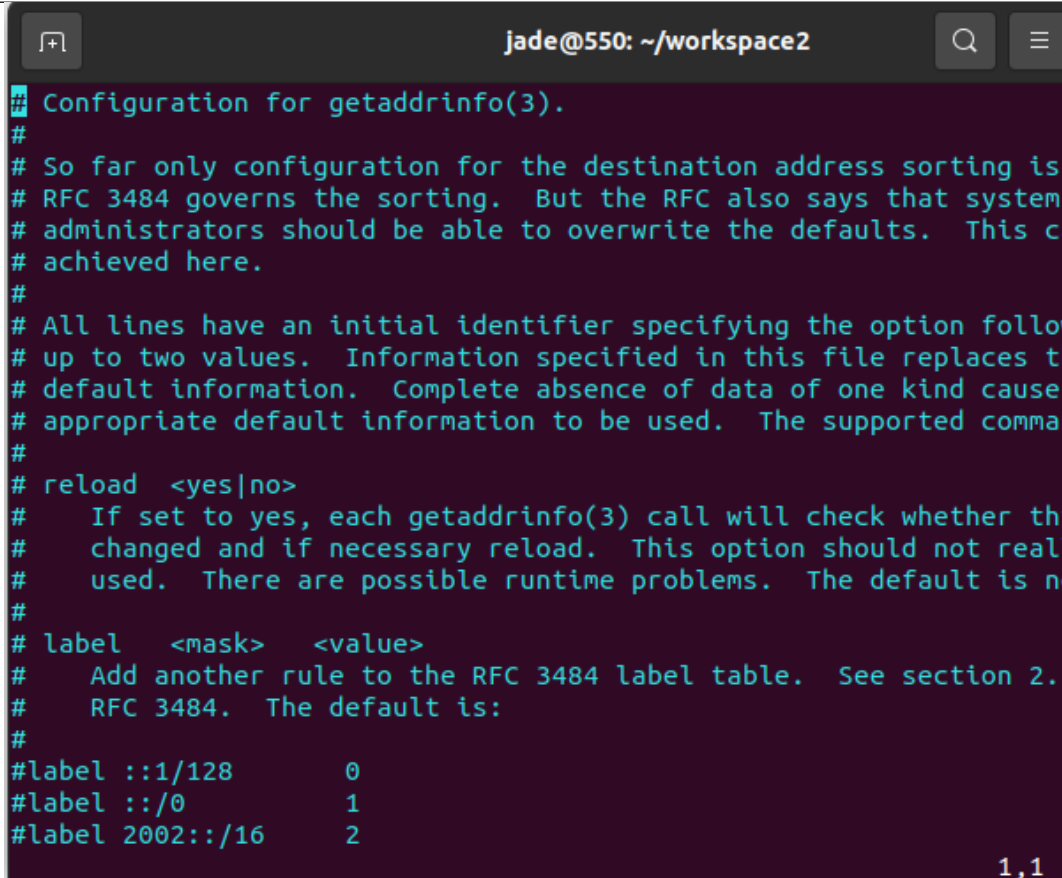
A terminal window with a dark background and light green text. The text shows various network configuration commands, including 'precedence' and 'scopev4'. The cursor is at the bottom of the screen, after the command ':w!'.

```
# Add another
# and 10.3 in
#
#precedence ::1/
#precedence ::/0
#precedence 2002
#precedence ::/96
#precedence ::fff
#
# For sites wh
#
#precedence ::fff

#
# scopev4 <mask>
# Add another
# By default t
# used. Chang
# The defaults
#
#scopev4 ::ffff:1
#scopev4 ::ffff:1
#scopev4 ::ffff:0
~
:w!
```

底行模式输入 :w!

( 11 ) 将光标移到首行。



The screenshot shows a terminal window with a dark background. The title bar at the top reads "jade@550: ~/workspace2". The terminal content is a configuration file for getaddrinfo(3), starting with a multi-line comment explaining RFC 3484 and the purpose of the configuration. It includes a "reload" option and a "label" section with three entries. The cursor is at the end of the last line, and the status bar at the bottom right shows "1,1".

```
## Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address sorting is
# RFC 3484 governs the sorting. But the RFC also says that system
# administrators should be able to overwrite the defaults. This ca
# achieved here.
#
# All lines have an initial identifier specifying the option follow
# up to two values. Information specified in this file replaces th
# default information. Complete absence of data of one kind causes
# appropriate default information to be used. The supported comman
#
# reload <yes|no>
#   If set to yes, each getaddrinfo(3) call will check whether th
#   changed and if necessary reload. This option should not real
#   used. There are possible runtime problems. The default is no
#
# label <mask> <value>
#   Add another rule to the RFC 3484 label table. See section 2.3
#   RFC 3484. The default is:
#
#label ::1/128      0
#label ::/0         1
#label 2002::/16    2
```

命令行模式输入 gg

( 12 ) 插入模式下输入 “ Hello, this is vim world!” 。

```
jade@550: ~/workspace2
Hello, this is vim world!
# Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address sorting is
# 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
# up to two values. Information specified in this file replaces the
# default information. Complete absence of data of one kind causes
# appropriate default information to be used. The supported commands
# are:
# reload <yes|no>
#   If set to yes, each getaddrinfo(3) call will check whether the
#   configuration has 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
#   of RFC 3484. The default is:
#
#label ::1/128 0
#label ::/0 1
-- 插入 --
```

底行模式输入 i

(13) 删除字符串 “this”。



```
jade@550: ~/workspace2
Hello, this is vim world!
# Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address sorting is
# RFC 3484 governs the sorting. But the RFC also says that system
# administrators should be able to overwrite the defaults. This c
# achieved here.
#
# All lines have an initial identifier specifying the option follo
# up to two values. Information specified in this file replaces t
# default information. Complete absence of data of one kind cause
# appropriate default information to be used. The supported comma
#
# reload <yes|no>
#   If set to yes, each getaddrinfo(3) call will check whether th
#   changed and if necessary reload. This option should not real
#   used. There are possible runtime problems. The default is n
#
# label <mask> <value>
#   Add another rule to the RFC 3484 label table. See section 2.
#   RFC 3484. The default is:
#
#label ::1/128      0
#label ::/0         1
-- 插入 --
```

( 14 ) 强制退出 vim , 不存盘。



```
Hello, is vim world!  
# Configuration for getaddrinfo  
#  
# So far only configuration  
# RFC 3484 governs the sort  
# administrators should be  
# achieved here.  
#  
# All lines have an initial  
# up to two values. Inform  
# default information. Com  
# appropriate default infor  
#  
# reload <yes|no>  
#   If set to yes, each ge  
#   changed and if necessa  
#   used. There are possi  
#  
# label <mask> <value>  
#   Add another rule to th  
#   RFC 3484. The default  
#  
#label ::1/128      0  
#label ::/0         1  
:q!
```

#### 四、实验过程分析与讨论

vim 的操作熟练后阅读、查询与筛选信息的能力将会大大加强，多使用有益于快速上手。

#### 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 5 月 22 日
学 号	2021213037	姓 名	白明予
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) ubuntu 20.04 LTS系统测试版。
- (3) libreoffice。

## 三、实验内容及结果

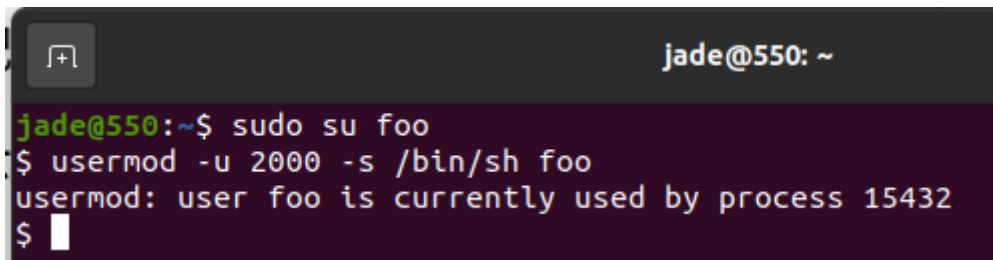
- 1、建立一个用户名为 `foo`，描述信息为 `bar`，登录 shell 为 `/bin/sh`，登录主目录为 `/home/foo` 的用户，并设置口令为 `123456`。



```
jade@550: ~  
jade@550:~$ useradd foo -m -c "bar" -s /bin/sh -d /home/foo -p 123456  
useradd: Permission denied.  
useradd: 无法锁定 /etc/passwd, 请稍后再试。  
jade@550:~$ sudo useradd foo -m -c "bar" -s /bin/sh -d /home/foo -p 123456  
[sudo] jade 的密码:  
jade@550:~$
```

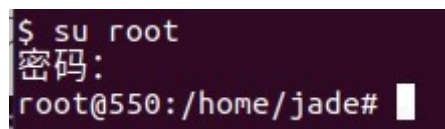
- 2、使用命令从用户 `root` 切换到用户 `foo`，修改 `foo` 的 `UID` 为 `2000`，

其 shell 类型为/bin/sh

A terminal window with a dark background. The title bar shows a window icon and the text 'jade@550: ~'. The terminal content shows a user prompt 'jade@550:~\$' followed by the command 'sudo su foo'. The next line shows a root prompt '\$' followed by the command 'usermod -u 2000 -s /bin/sh foo'. The following line shows the output 'usermod: user foo is currently used by process 15432'. The final line shows a root prompt '\$' followed by a blank line.

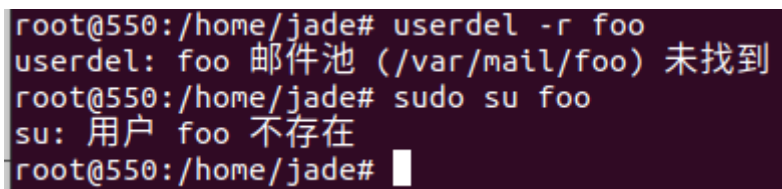
```
jade@550:~$ sudo su foo
$ usermod -u 2000 -s /bin/sh foo
usermod: user foo is currently used by process 15432
$
```

3、使用命令从用户 foo 切换到 root。

A terminal window with a dark background. The terminal content shows a root prompt '\$' followed by the command 'su root'. The next line shows the prompt '密码:' (Password:). The following line shows the output 'root@550:/home/jade#' followed by a cursor.

```
$ su root
密码:
root@550:/home/jade#
```

4、使用命令删除 foo 用户，并且在删除该用户的同时一起删除其主目录。

A terminal window with a dark background. The terminal content shows a root prompt 'root@550:/home/jade#' followed by the command 'userdel -r foo'. The next line shows the output 'userdel: foo 邮件池 (/var/mail/foo) 未找到'. The following line shows a root prompt 'root@550:/home/jade#' followed by the command 'sudo su foo'. The next line shows the output 'su: 用户 foo 不存在'. The final line shows a root prompt 'root@550:/home/jade#' followed by a cursor.

```
root@550:/home/jade# userdel -r foo
userdel: foo 邮件池 (/var/mail/foo) 未找到
root@550:/home/jade# sudo su foo
su: 用户 foo 不存在
root@550:/home/jade#
```

5、使用命令 newusers 批量创建用户，并使用命令 chpasswd 为这个批量用户创建密码（密码也是批量创建的），查看/etc/passwd 文件确认是否创建成功。

```
root@550:/home/jade# vi username.txt
root@550:/home/jade# newusers < username.txt
root@550:/home/jade# vi pw
root@550:/home/jade# chpasswd <pw
chpasswd: 第 4 行: 缺少新密码
root@550:/home/jade# cat /etc/passwd | tail -3
ahri:x:600:100:user:/home/ahri:/bin/bash
bai:x:601:100:user:/home/bai:/bin/bash
chen:x:602:100:user:/home/chen:/bin/bash
root@550:/home/jade#
```

6、使用命令创建用户组 group1，并在创建时设置其 GID 为 3000。

```
root@550:/home/jade# groupadd group1 -g 3000
root@550:/home/jade#
```

7、在用户组 group1 中添加两个之前批量创建的用户。

```
root@550:/home/jade# groupadd group1 -g 3000
root@550:/home/jade# gpasswd -a ahri group1
正在将用户“ahri”加入到“group1”组中
root@550:/home/jade# gpasswd -a bai group1
正在将用户“bai”加入到“group1”组中
root@550:/home/jade#
```

8、切换到 group1 组中的某个用户，在该用户下使用 sudo 命令查看/etc/shadow 文件，看一下是否可以执行。若不能执行，修改

sudoers 文件使得该用户可以查看/etc/shadow 文件内容。

```
root@550: /ho
root@550:/home/jade# su ahri
ahri@550:/home/jade$ sudo cat /etc/shadow
[sudo] ahri 的密码:
ahri 不在 sudoers 文件中。此事将被报告。
ahri@550:/home/jade$
```

```
ahri@550:/home/jade$ vi /etc/sudoers
ahri@550:/home/jade$ su root
密码:
root@550:/home/jade# vi /etc/sudoers
root@550:/home/jade# su ahri
ahri@550:/home/jade$ sudo cat /etc/shadow
[sudo] ahri 的密码:
root:$6$tz3LurJFCTRs6Da1$QDiBnnbKutLk4EDcInj05W6hofgXULOS2Nh89FEsQu2jnZ.
XszmpwpmNkr5QKPhKXjGrY0qugkyfB.:19432:0:99999:7:::
daemon*:19235:0:99999:7:::
bin*:19235:0:99999:7:::
sys*:19235:0:99999:7:::
sync*:19235:0:99999:7:::
games*:19235:0:99999:7:::
man*:19235:0:99999:7:::
lp*:19235:0:99999:7:::
```



#### 四、实验过程分析与讨论

明白组与组，用户与用户之间的关系，将有助与今后小组合作，安全保护的重要方面。

#### 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 5 月 22 日
学 号	2021213037	姓 名	白明予
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

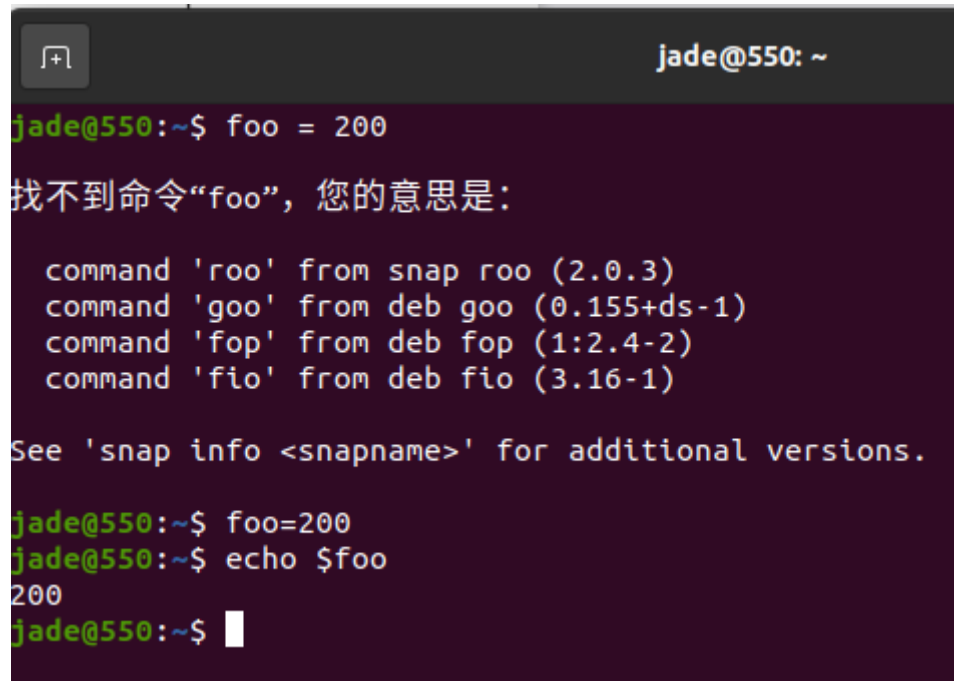
- 1、掌握 Shell 程序的创建过程及 Shell 程序的执行方法。
- 2、掌握 Shell 变量的定义方法，及用户定义变量、参数位置等。
- 3、掌握变量表达式，包括字符串比较、数字比较、逻辑测试、文件测试。
- 4、掌握条件判断语句，如 if 语句、case 语句。

### 二、实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) ubuntu 20.04 LTS 系统测试版。
- (3) libreoffice。

### 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）

A terminal window titled 'jade@550: ~' with a window icon in the top-left corner. The terminal shows the following commands and output:

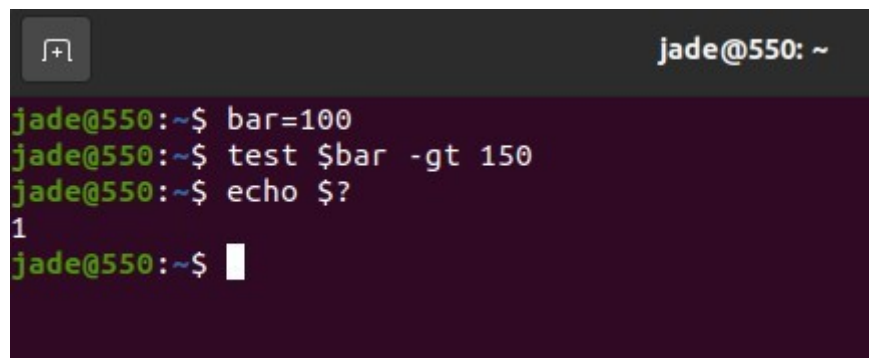
```
jade@550:~$ foo = 200
找不到命令“foo”，您的意思是：

command 'roo' from snap roo (2.0.3)
command 'goo' from deb goo (0.155+ds-1)
command 'fop' from deb fop (1:2.4-2)
command 'fio' from deb fio (3.16-1)

See 'snap info <snapname>' for additional versions.

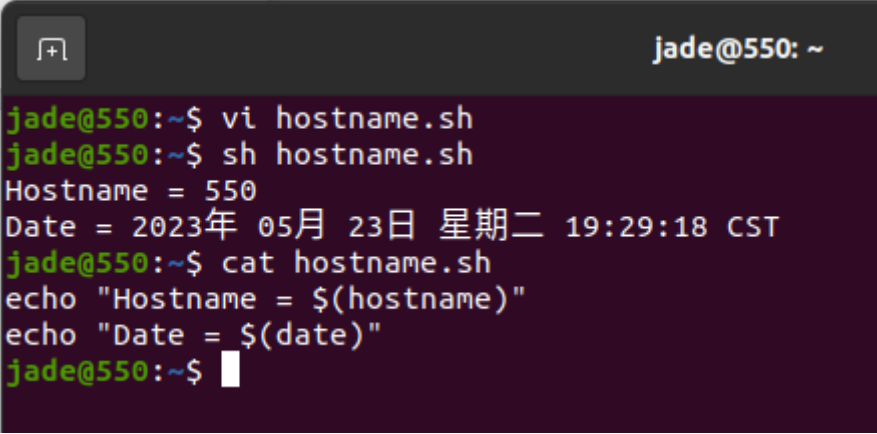
jade@550:~$ foo=200
jade@550:~$ echo $foo
200
jade@550:~$
```

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）

A terminal window titled 'jade@550: ~' with a window icon in the top-left corner. The terminal shows the following commands and output:

```
jade@550:~$ bar=100
jade@550:~$ test $bar -gt 150
jade@550:~$ echo $?
1
jade@550:~$
```

3、创建一个简单的 Shell 程序，其功能为显示计算机主机名（hostname）和系统时间（date）。



```
jade@550: ~  
jade@550:~$ vi hostname.sh  
jade@550:~$ sh hostname.sh  
Hostname = 550  
Date = 2023年 05月 23日 星期二 19:29:18 CST  
jade@550:~$ cat hostname.sh  
echo "Hostname = $(hostname)"  
echo "Date = $(date)"  
jade@550:~$
```

4、创建一个简单的 Shell 程序，要求带一个参数，判断参数是否是水仙花数。所谓水仙花数是指一个 3 位数，它的每个位上的数字的 3 次幂之和等于它本身。例如  $153=1^3+5^3+3^3$ ，153 是水仙花数。编写程序时要求首先进行参数个数判断，判断是否带了一个参数，如果没有参数则给出提示信息，否则给出该数是否是水仙花数。要求对 153，124，370 分别进行测试判断。

```
jade@550: ~  
jade@550:~$ vi flower.sh  
jade@550:~$ sh flower.sh 153  
yes  
jade@550:~$ sh flower.sh 124  
no  
jade@550:~$ sh flower.sh 370  
yes  
jade@550:~$
```

```
#!/bin/bash  
  
n=$1  
s=$n  
tot=0  
  
while [ $s -gt 0 ]  
do  
    m=$((s % 10))  
    tot=$((tot + m * m * m))  
    s=$((s / 10))  
    #echo $m $s $tot  
done  
  
#echo $tot $n  
if [ $tot = $n ]  
then  
    echo "yes"  
else  
    echo "no"  
fi  
~  
-- 插入 --
```

4、 创建一个简单的 shell 程序，输入 3 个数，输出这 3 个

数的和。

```
jade@550: ~  
jade@550:~$ vi sum.sh  
jade@550:~$ sh sum.sh 1 2 3  
6  
jade@550:~$ sh sum.sh 3 6 9  
18  
jade@550:~$
```

```
jade@550: ~  
#!/bin/bash  
  
n1=$1  
n2=$2  
n3=$3  
  
echo $(n1+n2+n3)
```

6、创建一个简单的 shell 程序，输入学生的成绩，给出该成绩对应的等级



jade@550: ~

```
jade@550:~$ vi grade.sh
jade@550:~$ sh grade.sh 98
98
98 A
jade@550:~$ sh grade.sh 85
85
85 B
jade@550:~$ sh grade.sh 77
77
C
jade@550:~$ sh grade.sh 60
60
C
```



jade@550: ~

```
#!/bin/bash

grade=$1

echo $grade

if [ "$grade" -ge "90" ]; then
    echo $grade "A"
elif [ "$grade" -ge "80" ]; then
    echo $grade "B"
elif [ "$grade" -ge "80" ]; then
    echo $grade "C"
elif [ "$grade" -ge "60" ]; then
    echo "D"
else
    echo "E"
fi

~
~
~
~
~
:wq
```



#### 四、实验过程分析与讨论

```
jade@550: ~  
jade@550:~$ foo = 200  
找不到命令“foo”，您的意思是：  
  
command 'roo' from snap roo (2.0.3)  
command 'goo' from deb goo (0.155+ds-1)  
command 'fop' from deb fop (1:2.4-2)  
command 'fio' from deb fio (3.16-1)  
  
See 'snap info <snapname>' for additional versions.  
jade@550:~$ foo=200  
jade@550:~$ echo $foo  
200  
jade@550:~$
```

变量定义中间不能有空格，否则会被误认为是命令

```
jade@550: ~  
jade@550:~$ vi grade.sh  
jade@550:~$ sh grade.sh 73  
grade.sh: 7: [: -ge: unexpected operator  
C  
jade@550:~$ bash grade.sh 73  
grade.sh: 第 7 行: [: -ge: 需要一元表达式  
C
```

判断与循环语句注意内部的符号，中括号与条件间必须有空格，没有空格会报错，数值加双引号若没有空格会被判定为空格。

shell 编程快速方便，运行快速，在手机开发等领域也有重要作用。

## 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 5 月 22 日
学 号	2021213037	姓 名	白明予
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) 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) ubuntu 20.04 LTS 系统测试版。
- (3) libreoffice。

## 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。(可以在当前目录下先建立几个 \*.c 文件, 用来测试, 复制到的指定目录可以自己建立一个)

```
jade@550: ~/linuxtest/target

jade@550:~/linuxtest/target$ ls
jade@550:~/linuxtest/target$ cd ..
jade@550:~/linuxtest$ ls
1.c 2.c 3.c move.sh target
jade@550:~/linuxtest$ sh
move.sh target/
jade@550:~/linuxtest$ sh move.sh
target
jade@550:~/linuxtest$ cd target/
jade@550:~/linuxtest/target$ ls
1.c 2.c 3.c
jade@550:~/linuxtest/target$
```

```
jade@550: ~/linuxtest

#!/bin/bash

read path

a=$(ls *.c)

for i in $a
do
    cp $i $path
done
```

2. 编写 shell 脚本，利用 while 循环求前 10 个偶数之和。

```
jade@550: ~/linuxtest

jade@550:~/linuxtest$ vi sum.sh
jade@550:~/linuxtest$ sh sum.sh
110
jade@550:~/linuxtest$
```

```
jade@550: ~/linuxtest

#!/bin/bash

i=1
sum=0

while [ $i -le 20 ]
do
    if [ $((i%2)) -eq 0 ]; then
        sum=$((sum+i))
    fi
    i=$((i+1))
done

echo $sum
```

3. 编写 shell 脚本，利用 until 循环求 1 到 10 的平方和。

```
jade@550: ~/linuxtest

jade@550:~/linuxtest$ vi sum.sh
jade@550:~/linuxtest$ sh sum.sh
385
jade@550:~/linuxtest$
```

```

jade@550: ~/linuxtest
#!/bin/bash

i=1
sum=0

while [ $i -le 10 ]
do
    sum=$((sum+i*i))
    i=$((i+1))
done

echo $sum

```

(4) 运行下列程序，观察程序的运行结果。break，break 2，continue，continue2，观察四种情况下的实验结果。

Break:

```

jade@550:~/linuxtest$ sh sum.sh
a1234
b1234
c1234
d1234
jade@550:~/linuxtest$

```

break 2:

```

jade@550:~/linuxtest$ sh sum.sh
a1234jade@550:~/linuxtest$

```

continue:

```
jade@550:~/linuxtest$ sh sum.sh  
a1234678910  
b1234678910  
c1234678910  
d1234678910  
jade@550:~/linuxtest$
```

continue 2:

```
jade@550:~/linuxtest$ sh sum.sh  
a1234b1234c1234d1234jade@550:~/linuxtest$  
jade@550:~/linuxtest$
```

#### 四、实验过程分析与讨论

循环是各种变成语言的基础，熟练运用是第一步。

shell 中的循环需要注意格式，较为严格，其次注意顺序即可。



## 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 5 月 22 日
学 号	2021213037	姓 名	白明予
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

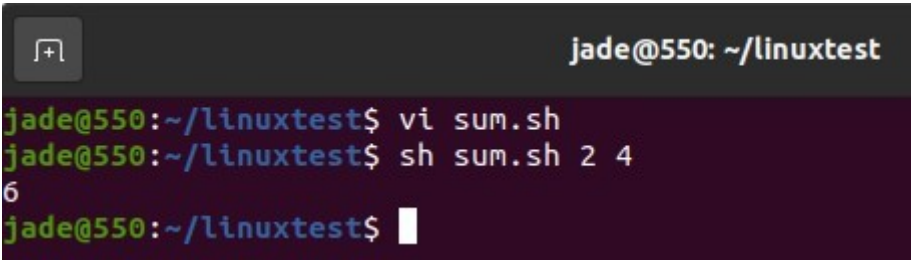
- 1、掌握 Shell 函数的定义方法
- 2、掌握 shell 函数的参数传递、调用和返回值
- 3、掌握 shell 函数的递归调用方法
- 4、理解 shell 函数的嵌套。

## 二、实验环境

- (1) 计算机的硬件配置 PC 系列微机。
- (2) ubuntu 20.04 LTS 系统测试版。
- (3) libreoffice。

## 三、实验内容及结果

1. 编写 shell 脚本，定义一个函数返回两个数的和。



```
jade@550: ~/linuxtest
jade@550:~/linuxtest$ vi sum.sh
jade@550:~/linuxtest$ sh sum.sh 2 4
6
jade@550:~/linuxtest$
```

```
jade@550: ~/linuxtest

#!/bin/bash

add(){
    return $((($1+$2))
}
a=$1
b=$2
add $a $b || sum=$?

echo $sum
```

2. 编写 shell 脚本，该脚本中定义一个递归函数，求  $n$  的阶乘。

```
jade@550: ~/linuxtest

jade@550:~/linuxtest$ vi jc.sh
jade@550:~/linuxtest$ sh jc.sh 4
24
jade@550:~/linuxtest$ sh jc.sh 5
120
jade@550:~/linuxtest$
```

```
jade@550: ~/linuxtest
#!/bin/bash

jc(){
    if [ $1 -gt 1 ]; then
        jc $((1-1))
        return $((1*$1))
    else
        return 1;
    fi
}

n=$1
jc $n
echo $?
```

3. 已知 shell 脚本 test.sh 内容如下所示，试运行下列程序，观察

程序运行结果，理解函数嵌套的含义。

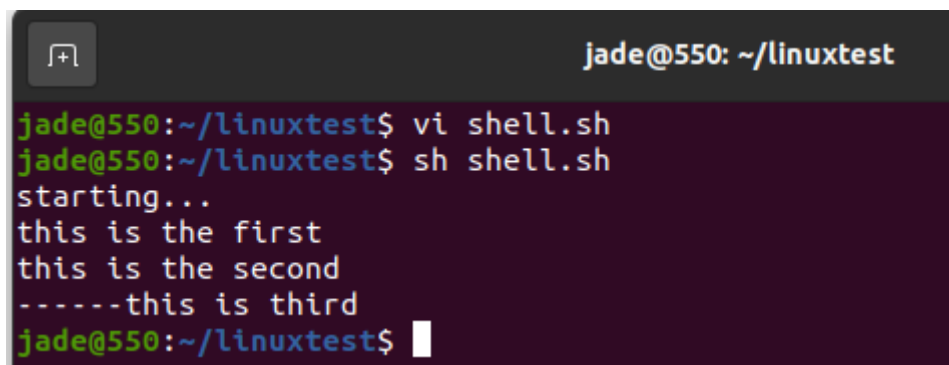
```
#!/bin/bash
function first() {
function second() {
function third() {
echo "-----this is third"
}
echo "this is the second"
third
}
echo "this is the first"
```

second

}

echo "starting..."

first



```
jade@550: ~/linuxtest
jade@550:~/linuxtest$ vi shell.sh
jade@550:~/linuxtest$ sh shell.sh
starting...
this is the first
this is the second
-----this is third
jade@550:~/linuxtest$
```



```
jade@550: ~/linuxtest
#!/bin/bash

first(){
    second(){
        third(){
            echo "-----this is third"
        }
        echo "this is the second"
        third
    }
    echo "this is the first"
    second
}
echo "starting..."
first
```

类似堆栈的效果，最外层的优先级最高。

#### 四、实验过程分析与讨论

函数是编程中非常重要的概念之一，它是一段可重复使用的代码块，通常接收一些输入参数并返回输出结果。函数可以帮助程序员把代码组织成更小、更可重用、更易于管理的模块化结构。模块化、可读性强、可维护性好的代码结构，提高代码复用性、可读性、可维护性和可扩展性。熟练掌握函数的概念和使用方法，能够使程序设计更加规范化和高效化，也是编程入门的基础之一。

#### 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 5 月 22 日
学 号	2021213037	姓 名	白明予
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 sed 基本编辑命令的使用方法
- 2、掌握 sed 与 shel 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) ubuntu 20.04 LTS 系统测试版。
- (3) libreoffice。

### 三、实验内容及结果

#### 1. 已知 quote.txt 文件内容如下

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.

试编写 sed 命令实现如下功能：

### (1) 删除 \$ 符号

```
jade@550: ~/linuxtest  
jade@550:~/linuxtest$ cat quote.txt  
The honeysuckle band played all night long for only $90.  
It was an evening of splendid music and company.  
Too bad the disco floor fell through at 23:10.  
The local nurse Miss P.Neave was in attendance.  
jade@550:~/linuxtest$ 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.  
jade@550:~/linuxtest$
```

### (2) 显示包含 music 文字的行内容及行号

```
jade@550: ~/linuxtest  
jade@550:~/linuxtest$ cat -n quote.txt  
1 The honeysuckle band played all night long for only $90.  
2 It was an evening of splendid music and company.  
3 Too bad the disco floor fell through at 23:10.  
4 The local nurse Miss P.Neave was in attendance.  
jade@550:~/linuxtest$ cat -n quote.txt | sed -n '/music/'  
sed: -e 表达式 #1, 字符 7: 遗漏命令  
jade@550:~/linuxtest$ cat -n quote.txt | sed -n '/music/p'  
2 It was an evening of splendid music and company.  
jade@550:~/linuxtest$
```

### (3) 在第 4 行后面追加文件 “hello world ! ”

```
jade@550: ~/linuxtest
jade@550:~/linuxtest$ cat quote.txt | sed 4a 'hello world'
sed: -e 表达式 #1, 字符 2: 期望在“a”, “c”, “i”之后有“\”
jade@550:~/linuxtest$ cat quote.txt | sed '4a hello world!'
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
hello world!
jade@550:~/linuxtest$
```

(4) 将文本 “The” 修改为 “Quod”

```
jade@550: ~/linuxtest
jade@550:~/linuxtest$ cat quote.txt | sed 's/The/Quod/g'
Quod honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
Quod local nurse Miss P.Neave was in attendance.
jade@550:~/linuxtest$
```

(5) 将第 3 行内容修改为 “This is the third line.”

```
jade@550: ~/linuxtest
jade@550:~/linuxtest$ cat quote.txt | sed '3c This is the third line.'
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
This is the third line.
The local nurse Miss P.Neave was in attendance.
jade@550:~/linuxtest$
```

(6) 删除第 2 行内容。

```
jade@550: ~/linuxtest

jade@550:~/linuxtest$ cat quote.txt | sed '2d'
The honeysuckle band played all night long for only $90.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
jade@550:~/linuxtest$
```

（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。

```
jade@550: ~/linuxtest

jade@550:~/linuxtest$ var=evening
jade@550:~/linuxtest$ cat quote.txt | sed -n '/$var/p'
jade@550:~/linuxtest$ cat quote.txt | sed -n "/$var/p"
It was an evening of splendid music and company.
jade@550:~/linuxtest$
```

2、已知文件 numbers.txt 内容如下：

one : two : three

four : five : six

（注：每个冒号前后都有空格）

试编写 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

```
jade@550:~/linuxtest$ cat > number.txt
one : two : tree
four : five : six
jade@550:~/linuxtest$ vi number.txt
jade@550:~/linuxtest$ cat number.txt | awk '{FS=":"}{print $2}'
:
five
jade@550:~/linuxtest$ cat number.txt | awk '{FS=" "}{print $2}'
:
:
jade@550:~/linuxtest$
```

3、已知文件 foo.txt 里面都是数字，且每行包含 3 个数字，数字之前以空格作为分隔符，试将 foo.txt 里的所有偶数输出，并输出偶数的个数。

例如：foo.txt 内容为：

2 4 3

15 46 79

```
jade@550:~/linuxtest$ cat foo.txt | awk '{for(i=1;i<=NF;i++) if($i%2==0){print $i;sum++}} END{print "sum=" sum}'
2
4
46
3
jade@550:~/linuxtest$
```

怪难的，人都是晕的

4、已知脚本的内容如下，试通过运行该脚本，理解该脚本实现的功能。

```
jade@550:~/linuxtest$ sh ep4.sh  
  
ep4.sh: 2: read: -p: bad variable name  
The honeysuckle band played all night long for only $90.  
It was an evening of splendid music and company.  
Too bad the disco floor fell through at 23:10.  
The local nurse Miss P.Neave was in attendance.  
4 found.  
jade@550:~/linuxtest$
```

传统字符串匹配，若能匹配，输出出现行编号与出现次数。

#### 四、实验过程分析与讨论

正则表达式的作用有：检索、替换那些符合某个模式（规则）的文本，可以通过一些设定的规则来匹配一些字符串。正则表达式，又称规则表达式，是一个强大的字符串匹配工具，是对字符串操作的一种逻辑公式。

与管道命令结合，将大大提高我们处理文本的能力。

#### 五、指导教师意见

指导教师签字：卢洋