

# 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 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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

```
coul@ubuntu: /etc
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

coul@ubuntu:~$ cd /etc
coul@ubuntu:/etc$ pwd
/etc
coul@ubuntu:/etc$
```

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

```
coul@ubuntu: ~
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

coul@ubuntu:~$ cd /etc
coul@ubuntu:/etc$ cd /home
coul@ubuntu:/home$ ls -a
.  ..  coul
coul@ubuntu:/home$ cd coul
coul@ubuntu:~$ ls-a
ls-a: command not found
coul@ubuntu:~$ ls -a
.          .cache      Downloads  .local     Public
..         .config     examples.desktop .mozilla   Templates
.bash_history Desktop    .gconf     Music      Videos
.bash_logout .dmrc     .gnupg     Pictures   .Xauthority
.bashrc      Documents .ICEauthority .profile   .xsession-errors
coul@ubuntu:~$
```

#### 3、使用命令创建目录/home/lyj/linux， 然后删除该目录。

```

coul@ubuntu:~$ mkdir linux
coul@ubuntu:~$ ls -a
.          Desktop          .ICEauthority  Public
..         .dmrc             linux          Templates
.bash_history Documents        .local         Videos
.bash_logout Downloads       .mozilla       .Xauthority
.bashrc    examples.desktop Music          .xsession-errors
.cache     .gconf         Pictures
.config    .gnupg         .profile
coul@ubuntu:~$ rmdir linux
coul@ubuntu:~$ ls -a
ls-a: command not found
coul@ubuntu:~$ ls -a
.          .cache      Downloads      .local      Public
..         .config     examples.desktop .mozilla    Templates
.bash_history Desktop      .gconf         Music       Videos
.bash_logout .dmrc       .gnupg         Pictures    .Xauthority
.bashrc      Documents  .ICEauthority  .profile    .xsession-errors

```

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

```

coul@ubuntu:~$ cat > abc
hello linux
coul@ubuntu:~$ 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。

```

coul@ubuntu:~$ mkdir ak
coul@ubuntu:~$ ls
abc Desktop Downloads Music Public Videos
ak Documents examples.desktop Pictures Templates
coul@ubuntu:~$ cp -r abc ak
coul@ubuntu:~$ cd ak
coul@ubuntu:~/ak$ ls
abc
coul@ubuntu:~/ak$ rm -i abc
rm: remove regular file 'abc'? y
coul@ubuntu:~/ak$ cd ..
coul@ubuntu:~$ rmdir ak

```

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。

```
coul@ubuntu:/etc$ cat adduser.conf | head -n 3
# /etc/adduser.conf: `adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.
```

```
coul@ubuntu:/etc$ cat adduser.conf | tail -n 5
# 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
--More-- (28%)
```

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

```
coul@ubuntu:/etc$ cd ..  
coul@ubuntu:/$ cd home  
coul@ubuntu:/home$ cd coul  
coul@ubuntu:~$ cat > facebook.txt  
google 110 5000  
baidu 100 5000  
guge 50 3000  
sohu 100 4500  
coul@ubuntu:~$
```

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

```
coul@ubuntu:~$ sort facebook.txt  
baidu 100 5000  
google 110 5000  
guge 50 3000  
sohu 100 4500
```

(2) 按公司人数排序

```
coul@ubuntu:~$ sort -n -k 2 -t' ' 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

```
coul@ubuntu:~$ sort -n -k2 -k3 -t' ' 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

```
coul@ubuntu:~$ sort -n -k3r -k2 -t' ' 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。

```
coul@ubuntu:~$ sort -k1.2 -t' ' facebook.txt
baidu 100 5000
sohu 100 4500
google 110 5000
guge 50 3000
```

#### 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 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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：chmod。

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；

```
coul@ubuntu:~/baz$ cat >qux
123456
coul@ubuntu:~/baz$
```

(2) 在用户目录下查找文件 qux，并显示该文件位置信息；

```
coul@ubuntu:~$ 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；

```
coul@ubuntu:~/baz$ wc qux
1 1 7 qux
```

(4) 在用户目录下查找文件 qux，并删除该文件；

```
coul@ubuntu:~$ 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。

```
coul@ubuntu:~$ cd baz
coul@ubuntu:~/baz$ ls
```

### 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；

```
coul@ubuntu:~$ mkdir path1
coul@ubuntu:~$ cd path1
coul@ubuntu:~/path1$ mkdir file1 file2
```

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

```
coul@ubuntu:~$ mkdir path2
coul@ubuntu:~$ cd path2
coul@ubuntu:~/path2$ mkdir file3
```

(3) 在用户目录下新建文件 file4;

```
coul@ubuntu:~$ mkdir file4
```

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

```
coul@ubuntu:~$ tar -cvf package.tar path1 file4
path1/
path1/file2/
path1/file1/
file4/
```

(5) 查看包 package.tar 的内容;

```
coul@ubuntu:~$ tar -tf package.tar
path1/
path1/file2/
path1/file1/
file4/
```

(6) 向包 package.tar 里添加文件夹 path2 的内容;

```
coul@ubuntu:~$ tar -rvf package.tar path2
path2/
path2/file3/
```

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

```
coul@ubuntu:~$ mkdir path3
coul@ubuntu:~$ cp package.tar path3
```

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

```
coul@ubuntu:~$ cd path3
coul@ubuntu:~/path3$ tar -xvf package.tar
path1/
path1/file2/
path1/file1/
file4/
path2/
path2/file3/
```

### 3. 符号链接内容

(1) 新建文件 foo.txt, 内容为 123;

```
coul@ubuntu:~$ touch foo.txt
coul@ubuntu:~$ echo '12345' > foo.txt
```

(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;

```
coul@ubuntu:~$ ln foo.txt bar.txt
coul@ubuntu:~$ diff bar.txt foo.txt
```

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

```
coul@ubuntu:~$ ls -li
839692 bar.txt          839718 facebook.txt  1055748 path2
1055744 baz             1055750 file4         1055751 path3
839300 Desktop         839692 foo.txt      839306 Pictures
839304 Documents       839305 Music         839303 Public
839301 Downloads       839703 package.tar     839302 Templates
839279 examples.desktop 1055745 path1       839307 Videos
```

(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;

```
coul@ubuntu:~$ 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;

```
coul@ubuntu:~$ rm foo.txt
coul@ubuntu:~$ car bar.txt
The program 'car' is currently not installed. You can install it by typing:
sudo apt install ucommon-utils
coul@ubuntu:~$ cat bar.txt
abc
```

(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;

```
coul@ubuntu:~$ ln -s bar.txt baz.txt
coul@ubuntu:~$ ls -li
839692 bar.txt      839279 examples.desktop  1055748 path2
1055744 baz          839718 facebook.txt   1055751 path3
839722 baz.txt      1055750 file4             839306 Pictures
839300 Desktop      839305 Music            839303 Public
839304 Documents    839703 package.tar       839302 Templates
839301 Downloads    1055745 path1             839307 Videos
```

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

```
coul@ubuntu:~$ rm bar.txt
coul@ubuntu:~$ cat baz.txt
cat: baz.txt: No such file or directory
```

#### 4. 权限管理

(1) 新建文件 qux.txt;

```
coul@ubuntu:~$ mkdir qux.txt
```

(2) 为文件 qux.txt 增加执行权限 (所有用户都可以执行)。

```
chmod: cannot access 'filename':
coul@ubuntu:~$ chmod 777 qux.txt
```

## 八、 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 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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、 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;

```
coul@ubuntu:~$ mkdir workspace1
```

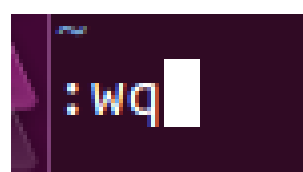
- (2) 进入目录workspace1;

```
coul@ubuntu:~$ cd workspace1  
coul@ubuntu:~/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:

```
coul@ubuntu:~$ vim.tiny test.c
```

- (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。

```
coul@ubuntu:~$ gcc -e test.c -o test
```

### 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;

```
coul@ubuntu:~$ mkdir worksapce2  
coul@ubuntu:~$ cd worksapce2
```

- (2) 进入workspace2目录;

- (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;



```
coul@ubuntu:~/workspace2$ cat /etc/gai.conf > ./gai.conf
```

(4) 使用vim编辑当前目录下的gai.conf;

```
coul@ubuntu:/$ vim.tiny pai.conf
```

(6) 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
"gai.conf" 651 2584C 18 1 Top
```

18gg

(7) 复制该行内容;

yy

(8) 将光标移到最后一行行首;

```
Terminal
coul@ubuntu: ~/workspace2
# 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
#
```

G

(9) 粘贴复制行的内容;

```
Terminal
coul@ubuntu: ~/workspace2
#
#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
#label <mask> <value>
```

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

```
intmain( )
```

```
{  
printf(" helloworld!\n" );  
return 0;  
}  
cat /etc/gai.conf > ./gai.conf
```

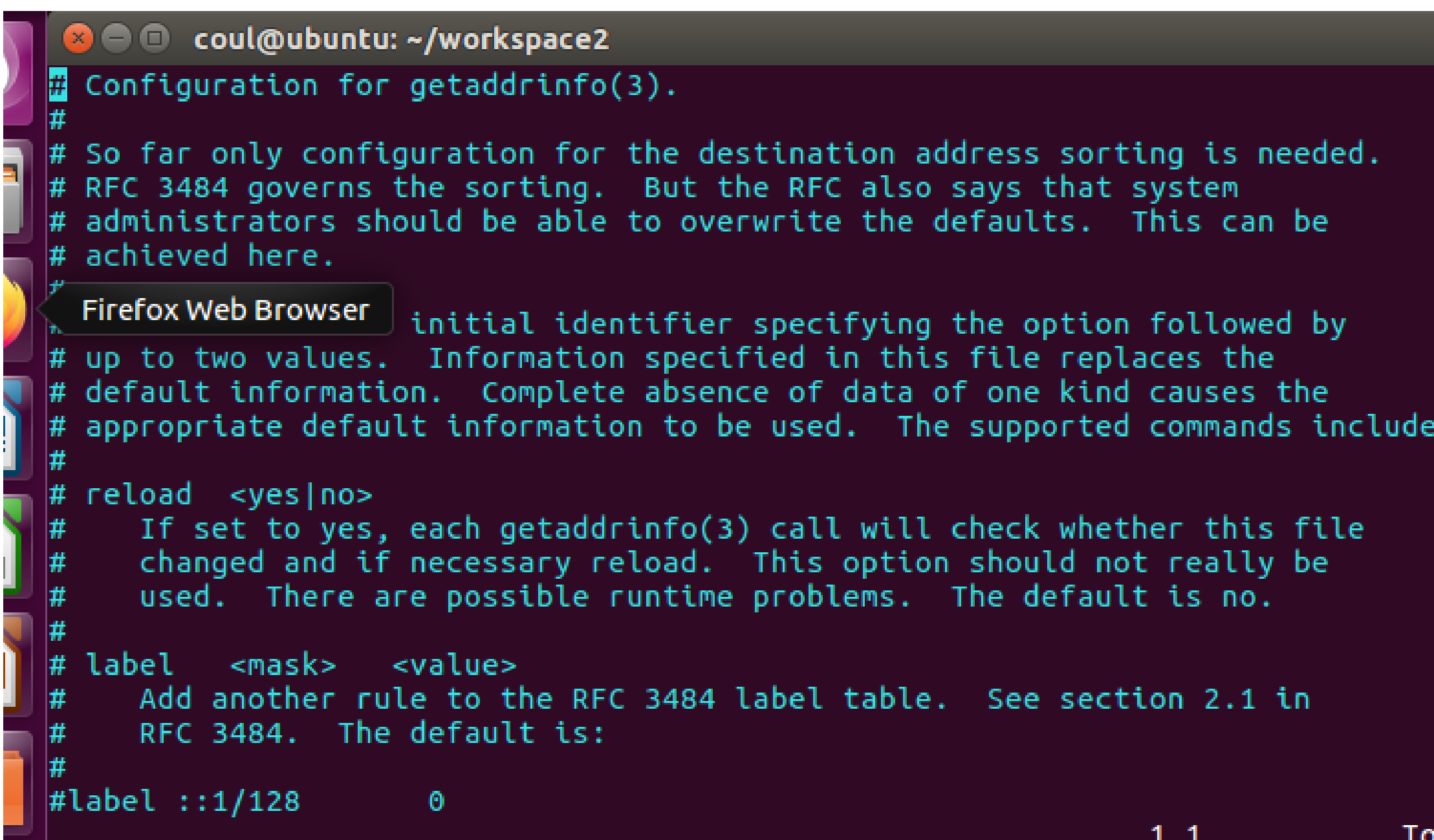
(10) 撤销第8步的动作;

u

(11) 存盘但不退出;



(12) 将光标移到首行;



```
coul@ubuntu: ~/workspace2  
## 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.  
# 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
```

gg

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

```

coul@ubuntu: ~/workspace2
hello, this is vim world# 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
-- INSERT --

```

(14) 删除字符串"this";

```

coul@ubuntu: ~/workspace2
hello, is vim world# 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

```

(15) 强制退出vim, 不存盘。

```

#label
:q!

```

十二、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 5 月 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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

十三、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。

## 十四、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。

## 十五、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@ubuntu:/home/coul# useradd foo -c "bar" -s /bin/sh -d /home/foo -u 1999
```

```
root@ubuntu:/home/coul# passwd foo
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

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@ubuntu:/home/coul# usermod foo -u 2000 -s /bin/csh
usermod: user foo is currently used by process 2689
```

```
root@ubuntu:/home/coul# su foo
```

```
root@ubuntu:/home/coul#
$ su root
Password:
```

3. 从用户foo切换到root;

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@ubuntu:/home/coul# 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@ubuntu:/home/coul# touch user123.txt
root@ubuntu:/home/coul# touch passwd123.txt
root@ubuntu:/home/coul# vim.tiny user123.txt
root@ubuntu:/home/coul# vim.tiny passwd123.txt
```

```
user1:x:2001:2001:user1:/home/user1:/bin/bash
user2:x:2002:2002:user2:/home/user2:/bin/bash
user3:x:2003:2003:user3:/home/user3:/bin/bash
```

```
User1:123456
user2:123456
user3:123456
```

```
root@ubuntu:/home/coul# newusers user123.txt
```

```
root@ubuntu:/home/coul# chpasswd < passwd123.txt
```

```
user1:x:2001:2001:user1:/home/jone:/bin/sh
user2:x:2002:2002:user2:/home/jone:/bin/sh
user3:x:2003:2003:user3:/home/jone:/bin/sh
```

6. 创建用户组group1，并在创建时设置其GID为3000；

```
root@ubuntu:/home/coul# groupadd group1 -g 3000
```

7. 在用户组group1中添加两个之前批量创建的用户；

```
root@ubuntu:/home/coul# gpasswd -a user1 group1
Adding user user1 to group group1
root@ubuntu:/home/coul# gpasswd -a user2 group1
Adding user user2 to group group1
```

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。

```
$ sudo cat /etc/shadow | tail -n 3
[sudo] password for user1:
user1 is not in the sudoers file. This incident will be reported.
```

```
root@ubuntu:/home/coul# vi /etc/sudoers
```



```
# User privilege specification
root    ALL=(ALL:ALL) ALL
user1   ALL=(ALL:ALL) ALL
```

```
root@ubuntu:/home/coul# su user1
$ sudo cat /etc/shadow | tail -n 3
[sudo] password for user1:
user1:$6$haqVha1G$CaAx5U2oAfNuqZfJLBHZBok8oAVtaauU05Q3oHQV5D5VX6Aw6HVKcSg6Dz2HW9IzUBRK
sxTim6MpVEPpcpXS3.:19501:0:99999:7:::
user2:$6$zzWSg00H$/1VzkNCpkHIwPQk/dW0q9KQYb52BZHsdH85gepT/CKHysKGm92fgsLgsQDv23EVkxxHq
AI4f972F0Bh5D9bN2.:19501:0:99999:7:::
user3:$6$9BZ1kfBv$zKK8xP8.Z2C34eDjn0/f9h7AbrNkp.P8LVt0AYf4mWgYE6EpY14k18KTaUoly/eWu7Wb
vEnwdXbVxAl8zV6W1.:19501:0:99999:7:::
$
```

## 十六、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 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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

十七、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。

十八、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。

## 十九、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);

```
coul@ubuntu:~$ foo=200
coul@ubuntu:~$ 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);

```
coul@ubuntu:~$ bar=100
coul@ubuntu:~$ test $bar -gt 150
coul@ubuntu:~$ 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);

```
coul@ubuntu:~$ vim.tiny 111.sh
```

```
#!/bin/bash

hostname
date
```

```
root@ubuntu:/home/coul# ./123.sh
ubuntu
Tue May 23 19:41:30 PDT 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;

```

coul@ubuntu: ~/workspace2
#!/bin/bash

n=$1
k=$n
sum=0

while [ $k -gt 0 ]
do
    m=$((k % 10))
    sum=$((sum + m * m * m))
    k=$((k/10))
    #echo $m $k $sum
done

#echo $sum $n
if [ $sum = $n ]
then
    echo "yes"
else
    echo "no"
fi

```

```

coul@ubuntu:~/workspace2$ sh shuixianhuashu.sh 153
no
coul@ubuntu:~/workspace2$ vim shuixianhuashu.sh
coul@ubuntu:~/workspace2$ sh shuixianhuashu.sh 153
yes
coul@ubuntu:~/workspace2$ sh shuixianhuashu.sh 124
no
coul@ubuntu:~/workspace2$ sh shuixianhuashu.sh 370
yes

```

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：

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。

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

sum=0

for x in $@
do
    #((sum=sum+x))
    let sum=sum+x
done

echo $sum
~
```

```
root@ubuntu:/home/coul# ./qiuhe.sh 10 11 12
33
root@ubuntu:/home/coul#
```

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

实现。

```
#!/bin/bash

read source
if [$source -ge 90]
then echo "A"
else if [$source -le 80]
then echo "B"
else if [$source -le 70]
then echo "C"
else if [$source -le 60]
then echo "D"
else if [$source -le 0]
then echo "E"
fi
fi
fi
fi
fi
~
```

7

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

if

elif

else

fi

## 二十、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	实验六 Shell 循环控制语句		
实验教室	丹青 922	实验日期	2023 年 5 月 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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。

二十二、 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。

## 二十三、 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）； 、

```
#!/bin/bash
for i in `ls *.c`
do
cp $i /workspace
done
```

可以事先在当前目录下建立若干\*.c文件用于测试。

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=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，并输出结果；

```
#!/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，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。

```
root@ubuntu:/home/coul# sh copy.sh
a1234
b1234
c1234
d1234
```

```
root@ubuntu:/home/coul# sh copy.sh
```

```
root@ubuntu:/home/coul# sh copy.sh
a1234678910
b1234678910
c1234678910
d1234678910
```

```
root@ubuntu:/home/coul# sh copy.sh
a1234b1234c1234d1234root@ubuntu:/home/coul#
```

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

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

```
echo-n$i
```

```
for j in12345678910; do
```

```
if [[ $j-eq5 ]]; then
```

```
---
```

```
fi
```

```
echo - n $j
```

```
done
```

```
echo''
```

```
done
```

## 二十四、 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	实验七 Shell 函数		
实验教室	丹青 922	实验日期	2023 年 5 月 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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

二十五、 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。

## 二十七、 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

read a
read b
sum=0
sum=$((a+b))
echo $sum
```

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 n
sum=1
i=1
while [ $i -le $n ]
do
sum=$((sum*i))

i=$((i+1))
done

echo $sum
```

3. 一个Shell脚本的内容如下所示：

试运行该程序，并观察程序运行结果，理解函数嵌套的含义。

```
root@ubuntu:/home/coul# sh copy.sh
starting...
-1- here is in the first func.
-2- here is in the second func.
-3- here is in the third func.
```

```
#!/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
```

## 二十八、 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	实验八 sed 和 awk		
实验教室	丹青 922	实验日期	2023 年 5 月 10 日
学 号	2021213075	姓 名	王晓波
专业班级	计算机科学与技术 05 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

<div>二十九、 实验目的</div> <div><div>1. 掌握 sed 基本编辑命令的使用方法；</div><div>2. 掌握 sed 与 Shell 变量的交互方法；</div><div>3. 掌握 awk 命令的使用方法；</div><div>4. 掌握 awk 与 Shell 变量的交互方法。</div></div>
<div>三十、实验环境</div> <div><div>(1) 计算机的硬件配置 PC 系列微机。</div><div>(2) 计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。</div></div>

## 三十一、实验内容及结果

1. 文件 quote.txt 的内容如下所示:

试使用 sed 命令实现如下功能:

(1) 删除\$符号;

```
root@ubuntu:/home/coul# cat -n quote.txt | sed 's/\$/g'
 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.
```

(2) 显示包含 music文字的行内容及行号;

```
root@ubuntu:/home/coul# 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@ubuntu:/home/coul# cat -n quote.txt | sed '4a hello world'
 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.
hello world
```

(4) 将文本"The"替换为"Quod";

```
root@ubuntu:/home/coul# cat -n quote.txt | sed 's/the/Quod/g'
 1 The honeysuckle band played all night long for only $90.
 2 It was an evening of splendid music and company.
 3 Too bad Quod disco floor fell through at 23:10.
 4 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@ubuntu:/home/coul# cat -n quote.txt | sed '3c This is the third line.'
 1 The honeysuckle band played all night long for only $90.
 2 It was an evening of splendid music and company.
This is the third line.
 4 the local nurse Miss P.Neave was in attendance.
```

(6) 删除第 2行内容;

```
root@ubuntu:/home/coul# cat -n quote.txt | sed '2d'
 1 The honeysuckle band played all night long for only $90.
 3 Too bad the disco floor fell through at 23:10.
 4 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@ubuntu:/home/coul# cat -n quote.txt | sed -n "/$var/p"
2 It was an evening of splendid music and company.
```

2. 文件 numbers.txt 的内容如下所示:

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@ubuntu:/home/coul# awk 'BEGIN{FS=" "}{print $2}' numbers.txt
:
```

```
root@ubuntu:/home/coul# awk 'BEGIN{FS=":"}{print $2}' numbers.txt
two
five
```

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。

```
root@ubuntu:/home/coul# awk '{for(i=1;i<=NF;i++)if($i%2==0){print $i;sum++}}END{print "sum=sum}" foo.txt
2
4
46
sum=3
```

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.

one : two : three

four : five : six

例如：foo.txt 内容为：

则输出为：

4. 脚本的内容如下所示：

试运行该脚本，并理解该脚本实现的功能。

```
root@ubuntu:/home/coul# sh cc.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 fell through at 23:10.
the local nurse Miss P.Neave was in attendance.
4 found.
```

243

154679

even:

2

4

46

numbers:

3

#!/bin/bash

read -p " enter search pattern: " pattern

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

三十二、 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