

# 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 6 日
学 号	2021214893	姓 名	任平实
专业班级	奥林 21 级计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学  
信息与计算机科学技术实验中心

## 一、 实验目的

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

## 二、 实验环境

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

### 三、 实验内容及结果

#### 1. 使用命令切换到/etc 目录，并显示当前工作目录路径

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[rocky@localhost ~]$ cd /etc
[rocky@localhost etc]$ pwd
/etc
[rocky@localhost etc]$
```

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

```
[rocky@localhost home]$ cd rocky
[rocky@localhost ~]$ ls -a
.          .bash_profile  .esd_auth  linux      test.txt    视频  桌面
..         .bashrc       facebook.txt .local     .viminfo    图片
1.txt      .cache        .ICEauthority locate     work.txt    文档
.bash_history .config      itcast     .mozilla   公共        下载
.bash_logout .dbus        .lessht    newfile1   模板        音乐
[rocky@localhost ~]$
```

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

```
[rocky@localhost ~]$ ls
1.txt      itcast  newfile1  work.txt  模板  图片  下载  桌面
facebook.txt locate  test.txt  公共      视频  文档  音乐
[rocky@localhost ~]$ mkdir linux
[rocky@localhost ~]$ ls
1.txt      itcast  locate    test.txt  公共  视频  文档  音乐
facebook.txt linux   newfile1  work.txt  模板  图片  下载  桌面
[rocky@localhost ~]$ rmdir linux
[rocky@localhost ~]$ ls
1.txt      itcast  newfile1  work.txt  模板  图片  下载  桌面
facebook.txt locate  test.txt  公共      视频  文档  音乐
[rocky@localhost ~]$
```

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

```
[rocky@localhost ~]$ cat > abc
Hello,linux!
[rocky@localhost ~]$ ls
1.txt  facebook.txt  locate  test.txt  公共  视频  文档  音乐
abc    itcast       newfile1  work.txt  模板  图片  下载  桌面
[rocky@localhost ~]$ cat abc
Hello,linux!
[rocky@localhost ~]$
```

5、使用命令创建目录/home/用户/ak，然后将/home/用户/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。

```
rocky@localhost ~]$ mkdir ak
rocky@localhost ~]$ cp abc ak
rocky@localhost ~]$ cd ak
rocky@localhost ak]$ ls
abc
rocky@localhost ak]$ cd
rocky@localhost ~]$ ls
1.txt  ak  itcast  newfile1  work.txt  模板  图片  下载  桌面
abc  facebook.txt  locate  test.txt  公共  视频  文档  音乐
rocky@localhost ~]$ rm -r ak
rocky@localhost ~]$ ls
1.txt  facebook.txt  locate  test.txt  公共  视频  文档  音乐
abc  itcast  newfile1  work.txt  模板  图片  下载  桌面
rocky@localhost ~]$
```

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。

```
rocky@localhost ~]$ head -3 /etc/adduser.conf
rocky@localhost etc]$ tail -5 /etc/adduser.conf
```

7、分屏查看文件/etc/adduser.conf 的内容。

```
rocky@localhost etc]$ more /etc/adduser.conf
```

8、使用命令cat用输出重定向在/home/用户 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

```
[rocky@localhost ~]$ cat > facebook.txt
google 110 5000
baidu 100 5000
guge 50 3000
sohu 100 4500
[rocky@localhost ~]$
```

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。

```
[rocky@localhost ~]$ sort facebook.txt
baidu 100 5000
google 110 5000
guge 50 3000
sohu 100 4500
[rocky@localhost ~]$ sort -n -t ' ' -k 2 facebook.txt
guge 50 3000
baidu 100 5000
sohu 100 4500
google 110 5000
```

```
[rocky@localhost ~]$ sort -n -t ' ' -k 2 -k 3 facebook.txt
guge 50 3000
sohu 100 4500
baidu 100 5000
google 110 5000
[rocky@localhost ~]$ sort -t ' ' -nrk3 -nk2 facebook.txt
google 110 5000
baidu 100 5000
sohu 100 4500
guge 50 3000
```

```
[rocky@localhost ~]$ sort -t ' ' -k1,2 facebook.txt
baidu 100 5000
sohu 100 4500
google 110 5000
guge 50 3000
```

#### 四、 实验过程分析与讨论

对 sort 方法使用遗忘，需要看 ppt 来写，仍需多加训练

#### 五、 指导教师意见

指导教师签字：卢洋

# 实验报告

实验名称	实验二 Linux 常用命令（二）		
实验教室	丹青 922	实验日期	2023 年 5 月 6 日
学 号	2021214893	姓 名	任平实
专业班级	奥林 21 级计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学  
信息与计算机科学技术实验中心

一、实验目的

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

## 二、实验环境

(1) 计算机的硬件配置 PC 系列微机。

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

## 三、实验内容及结果

### 1. 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；

```
[rocky@localhost ~]$ mkdir baz
[rocky@localhost ~]$ cd baz
[rocky@localhost baz]$ cat > qux
fgagfadfghfgdg
awetrarhgrsgdaes
dwacssssssssssssssss
asdadaasaas
dadsdsadsaad
fewgwgrge
rfqrefscs
qwrqedfgsa
sfgdhjygfsda
ghjhgfdsaa
dfsgjhffrdsgrht
fddfsdsfsd
[rocky@localhost baz]$
```

(2) 在用户目录下查找文件 qux ，并显示该文件位置信息；

```
[rocky@localhost ~]$ find /home/rocky/ -name qux
/home/rocky/baz/qux
[rocky@localhost ~]$
```

(3) 统计文件 qux 中所包含内容的行数、字数和字节数；

```
[rocky@localhost ~]$ wc -lwc ./baz/qux
12 12 161 ./baz/qux
[rocky@localhost ~]$
```

(4) 在用户目录下查找文件 qux ，并删除该文件；

```
[rocky@localhost ~]$ find /home/rocky/ -name qux -exec rm {} \;
```



(5) 查看文件夹 baz 内容，看一下是否删除了文件 qux 。

```
[rocky@localhost ~]$ cd ./baz/  
[rocky@localhost baz]$ ls  
[rocky@localhost 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 ；

```
[rocky@localhost ~]$ mkdir path1  
[rocky@localhost ~]$ cd path1  
[rocky@localhost path1]$ touch file1 file2
```

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

```
[rocky@localhost ~]$ mkdir path2  
[rocky@localhost ~]$ cd path2  
[rocky@localhost path2]$ touch file3  
[rocky@localhost path2]$
```

(3) 在用户目录下新建文件 file4 ；

```
[rocky@localhost ~]$ touch 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 ；

```
[rocky@localhost ~]$ tar -cvf package.tar path1 file4  
path1/  
path1/file1  
path1/file2  
file4
```

(5) 查看包 package.tar 的内容；

```
[rocky@localhost ~]$ tar -tf package.tar  
path1/  
path1/file1  
path1/file2  
file4
```

(6) 向包 package.tar 里添加文件夹 path2 的内容；

```
[rocky@localhost ~]$ tar -rf package.tar path2  
[rocky@localhost ~]$
```

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

```
[rocky@localhost ~]$ mkdir path3
[rocky@localhost ~]$ cp package.tar ./path3
[rocky@localhost ~]$
```

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

```
[rocky@localhost path3]$ tar -xvf package.tar
path1/
path1/file1
path1/file2
file4
path2/
path2/file3
[rocky@localhost path3]$
```

### 3. 符号链接内容

(1) 新建文件 foo.txt ，内容为 123 ；

```
[rocky@localhost ~]$ vim foo.txt
[rocky@localhost ~]$ ls
1.txt  facebook.txt  itcast  package.tar  path3  公共  图片  音乐
abc    file4          locate  path1        test.txt  模板  文档  桌面
baz    foo.txt        newfile1 path2        work.txt  视频  下载
[rocky@localhost ~]$ cat foo.txt
123
```

(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；

```
[rocky@localhost ~]$ ln foo.txt bar.txt
[rocky@localhost ~]$ diff foo.txt bar.txt
[rocky@localhost ~]$
```

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

```
[rocky@localhost ~]$ ls -li
78192 1.txt          77139 foo.txt      16786331 path2        33575029 视频
3163403 abc            16786325 itcast       52038390 path3        16786376 图片
77139 bar.txt       16786334 locate      78189 test.txt    51737664 文档
52038386 baz        77124 newfile1     78191 work.txt    3163405 下载
77122 facebook.txt  77136 package.tar  33575028 公共        3163406 音乐
77133 file4        3163386 path1        16786375 模板        51737663 桌面
[rocky@localhost ~]$
```

(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；

```
[rocky@localhost ~]$ vim bar.txt
[rocky@localhost ~]$ diff bar.txt foo.txt
[rocky@localhost ~]$
```

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

```
[rocky@localhost ~]$ rm foo.txt
[rocky@localhost ~]$ ls -li bar.txt
77139 bar.txt
[rocky@localhost ~]$ cat bar.txt
abc
[rocky@localhost ~]$
```

(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；

```
[rocky@localhost ~]$ ln -s bar.txt baz.txt
[rocky@localhost ~]$ ls -li
 78192 1.txt          77133 file4          16786331 path2          33575029 视频
3163403 abc          16786325 itcast          52038390 path3          16786376 图片
 77139 bar.txt      16786334 locate          78189 test.txt     51737664 文档
52038386 baz          77124 newfile1        78191 work.txt     3163405 下载
 77137 baz.txt       77136 package.tar  33575028 公共          3163406 音乐
 77122 facebook.txt 3163386 path1          16786375 模板          51737663 桌面
[rocky@localhost ~]$ diff bar.txt baz.txt
[rocky@localhost ~]$
```

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

```
[rocky@localhost ~]$ rm bar.txt
[rocky@localhost ~]$ cat baz.txt
cat: baz.txt: 没有那个文件或目录
[rocky@localhost ~]$
```

## 4. 权限管理

(1) 新建文件 qux.txt；

```
[rocky@localhost ~]$ touch qux.txt
[rocky@localhost ~]$
```

(2) 为文件 qux.txt 增加执行权限（所有用户都可以执行）

```
[rocky@localhost ~]$ chmod a+x qux.txt  
[rocky@localhost ~]$ █
```

#### 四、实验过程分析与讨论

知识点一段时间不看比较不熟悉，需要通过查询得以做出，仍需多加复习。

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 6 日
学 号	2021214893	姓 名	任平实
专业班级	奥林 21 级计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学  
信息与计算机科学技术实验中心

一、实验目的

掌握 vim 编辑器及 gcc 编译器的使用方法。

二、实验环境

(1) 计算机的硬件配置 PC 系列微机。

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

### 三、实验内容及结果

## 1. vim 编辑器和 gcc 编译器的简单使用:

(1) 在用户目录下新建一个目录，命名为 `workspace1` ；

```
[rocky@localhost ~]$ mkdir workspace1
[rocky@localhost ~]$
```

(2) 进入目录 workspace1 ;

```
[rocky@localhost ~]$ cd workspace1
[rocky@localhost 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：

```
[rocky@localhost workspace]$ vim test.c
[rocky@localhost workspace]$
```

(4) 保存 test.c 的内容, 并退出;

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
#include <stdio.h>  
  
int main( )  
{  
    printf("hello world!\n");  
    return 0;  
}
```

: 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。

```
[rocky@localhost workspace]$ gcc test.c -o test
[rocky@localhost workspace]$ ./test
hello world!
[rocky@localhost workspace]$
```

## 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;

```
rocky@localhost ~]$ mkdir workspace2
```

(2) 进入 workspace2 目录:

```
[rocky@localhost ~]$ cd workspace2
[rocky@localhost workspace2]$
```

(3) 使用以下命令： 将文件 /etc/gai.conf 的内容复制到当前目录下的新建文件 gai.conf 中；

```
[rocky@localhost workspace2]$ cat /etc/gai.conf > ./gai.conf  
[rocky@localhost workspace2]$
```

(4) 使用 vim 编辑当前目录下的 gai.conf ；

```
[rocky@localhost workspace2]$ vim gai.conf
```

(5) 将光标移到第 18 行；



```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
ewWWWERGTHYGJFFAW  
hdgtrseawet  
yjtrsegrnt  
54b64564y54  
6 47865ner  
34b7y46ervdf  
v34ertg34  
bretg5b  
r45tr  
rgh  
ryert  
rertre  
rtgft  
tehry  
nu565  
5n65ut  
5n6u65u  
5nur32  
rv2er  
k87uyk  
versb  
6n5  
wryuu  
:18 18,1
```

(6) 复制该行内容；

(7) 将光标移到最后一行行首；



```
rocky@localhost:~/workspace2
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
 87uyk
versb
6n5
wryuu
4wrvbnn
78muytfgr
34m,
4h5tydrf
28,0-1
```

(8) 粘贴复制行的内容;

```
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
 87uyk
versb
6n5
wryuu
4wrvbnn
78muytfgr
34m,
4h5tydrf
5nur32
28,1
```

(9) 撤销第 8 步的动作;

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
87uyk
versb
6n5
wryuu
4wrvbnn
78muytfgr
34m,
4h5tydrf
1 行被去掉 ; before #2 40 seconds ago 28,0-1 底
```

(10) 存盘但不退出;

```
rocky@localhost:~/workspace2
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
87uyk
versb
6n5
wryuu
4wrvbnn
78muytfgr
34m,
4h5tydrf
"gai.conf" 28L, 222C 已写入 28,0-1 底
```

(11) 将光标移到首行;

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
ewWWWWERGTHYGJFFAW
hdgtrseawet
yjrsegrnt
54b64564y54
6 47865ner
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
87uyk
versb
6n5
wryuu
1,1
```

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

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
Hello, this is vim world!
ewWWWWERGTHYGJFFAW
hdgtrseawet
yjrsegrnt
54b64564y54
6 47865ner
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
87uyk
versb
6n5
-- 插入 --
1,26
```

(13) 删除字符串 "this" ;

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
Hello, is vim world!
ewWWWWWERGTHYGJFFAW
hdgtrseawet
yjtrsegrnt
54b64564y54
6 47865ner
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
87uyk
versb
6n5
1,8
```

#### (14) 强制退出 vim , 不存盘

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(I) 帮助(H)
Hello, is vim world!
ewWWWWWERGTHYGJFFAW
hdgtrseawet
yjtrsegrnt
54b64564y54
6 47865ner
34b7y46ervdf
v34ertg34
bretg5b
r45tr
rgh
ryert
rertre
rtgft
tehry
nu565
5n65ut
5n6u65u
5nur32
rv2er
87uyk
versb
6n5
: q!
```

#### 四、实验过程分析与讨论

对 vim 编辑器相关命令仍需多熟悉

#### 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 6 日
学 号	2021214893	姓 名	任平实
专业班级	奥林 21 级计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学  
信息与计算机科学技术实验中心

一、实验目的

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

## 二、实验环境

(1) 计算机的硬件配置 PC 系列微机。

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

## 三、实验内容及结果

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

```
[root@localhost rocky] # useradd -d /home/foo -s /bin/sh -p 123456 foo
[root@localhost rocky] #
```

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@localhost rocky] # su foo
sh-4.2$ exit
exit
[root@localhost rocky] # usermod -u 2000 foo
```

```
[root@localhost rocky] # usermod -s /bin/csh foo
[root@localhost rocky] #
```

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@localhost rocky] # userdel -r foo
[root@localhost rocky] #
```

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；

6. 创建用户组 group1，并在创建时设置其 GID 为 3000；

7. 在用户组 group1 中添加两个之前批量创建的用户；

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@localhost rocky] # userdel -r foo
[root@localhost rocky] # newusers userlist.txt
newusers: userlist.txt: 没有那个文件或目录
[root@localhost rocky] # touch userlist.txt
[root@localhost rocky] # vi userlist.txt
[root@localhost rocky] # newusers userlist.txt
[root@localhost rocky] # touch mima
[root@localhost rocky] # vi mima
[root@localhost rocky] # cat mima | chpasswd
[root@localhost rocky] # cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
```

```
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
polkitd:x:999:998:User for polkitd:/:/sbin/nologin
unbound:x:998:997:Unbound DNS resolver:/etc/unbound:/sbin/nologin
libstoragemgmt:x:997:995:daemon account for libstoragemgmt:/var/run/lsm:/sbin/nologin
colord:x:996:994:User for colord:/var/lib/colord:/sbin/nologin
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
saned:x:995:993:SANE scanner daemon user:/usr/share/sane:/sbin/nologin
sasauthd:x:994:76:Sasauthd user:/run/sasauthd:/sbin/nologin
abrt:x:173:173::/etc/abrt:/sbin/nologin
setroubleshoot:x:993:990::/var/lib/setroubleshoot:/sbin/nologin
rtkit:x:172:172:RealtimeKit:/proc:/sbin/nologin
pulse:x:171:171:PulseAudio System Daemon:/var/run/pulse:/sbin/nologin
radvd:x:75:75:radvd user:/:/sbin/nologin
chrony:x:992:987::/var/lib/chrony:/sbin/nologin
qemu:x:107:107:qemu user:/:/sbin/nologin
tss:x:59:59:Account used by the trousers package to sandbox the tcsd daemon:/dev/null:/sbin/nologin
usbmuxd:x:113:113:usbmuxd user:/:/sbin/nologin
geoclue:x:991:985:User for geoclue:/var/lib/geoclue:/sbin/nologin
gluster:x:990:984:GlusterFS daemons:/run/gluster:/sbin/nologin
```

```
nisnobody:x:65534:65534:Anonymous NFS User:/var/lib/nis:/sbin/nologin
gnome-initial-setup:x:989:983::/run/gnome-initial-setup:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
avahi:x:70:70:Avahi mDNS/DNS-SD Stack:/var/run/avahi-daemon:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
tcpdump:x:72:72::/:/sbin/nologin
rocky:x:1000:1000:Rocky:/home/rocky:/bin/bash
[root@localhost rocky] # groupadd -g 3000 group1
[root@localhost rocky] # gpasswd -a user1 user2
gpasswd: 用户 'user1' 不存在
[root@localhost rocky] # su user1
su: user user1 does not exist
[root@localhost rocky] # sudo /etc/shadow
sudo: /etc/shadow: 找不到命令
[root@localhost rocky] #
```



#### 四、实验过程分析与讨论

对用户管理相关命令不熟，通过看 ppt 等得以解决。

#### 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 5 月 6 日
学 号	2021214893	姓 名	任平实
专业班级	奥林 21 级计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学  
信息与计算机科学技术实验中心

一、实验目的

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

## 二、实验环境

(1) 计算机的硬件配置 PC 系列微机。

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

### 三、实验内容及结果

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

```
[root@localhost rocky]# foo=200
```

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

```
[root@localhost rocky] # bar=100
[root@localhost rocky] # test ${bar} -gt 150
[root@localhost rocky] # echo $?
1
[root@localhost rocky] #
```

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

```
[root@localhost rocky] # vim getInfo.sh
[root@localhost rocky] # bash ./getInfo.sh
2023年 05月 15日 星期一 11:48:39 CST
localhost.localdomain
[root@localhost rocky] #
```

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

```
date
hostname
```

~~~~~

4.8 全部

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： 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。

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
#!/bin/bash

if [ $# = 0 ]
then
    echo "please give a number!"
else
    a=$(( $1 % 10 ))
    b=$(( $1 / 10 % 10 ))
    c=$(( $1 / 100 % 10 ))
    sum=$(( $a**3 + $b**3 + $c**3 ))
    if [ ${sum} = $1 ]
    then
        echo "true"
    else
        echo "false"
    fi
fi
```

```
[root@localhost rocky]# vim sxhNumber.sh
[root@localhost rocky]# bash sxhNumber.sh 153
true
```

```
[root@localhost rocky]# bash sxhNumber.sh 124
false
```

```
[root@localhost rocky]# bash sxhNumber.sh 370
true
```

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：

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
#!/bin/bash

numsum()
{
    read -p "please send me the first num: " num1
    read -p "please send me the second num: " num2
    read -p "please send me teh third num: " num3
    echo "the sum is $(( ${num1} + ${num2} + ${num3} ))"
    return $(( ${num1} + ${num2} + ${num3} ))
}

numsum
echo $?
```

```
[rocky@localhost ~]$ vim sum.sh
[rocky@localhost ~]$ bash sum.sh
please send me the first num: 1
please send me the second num: 2
please send me the third num: 3
the sum is 6
6
[rocky@localhost ~]$
```

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

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
#!/bin/bash

if [[ "$1" -gt "90" ]]
then
    echo A

elif [[ "$1" -gt "80" && "$1" -le "90" ]]
then
    echo B
elif [[ "$1" -gt "70" && "$1" -le "80" ]]
then
    echo C
elif [[ "$1" -gt "60" && "$1" -le "70" ]]
then
    echo D
else
    echo E
fi
~
~
~
~
-- 插入 --
```

#### 四、实验过程分析与讨论

C 语言事先没学过，通过看 ppt 和上网搜索得以解决

#### 五、指导教师意见

指导教师签字：卢洋

# 实验报告

|      |                      |      |                |
|------|----------------------|------|----------------|
| 实验名称 | 实验六 Shell 循环控制语句     |      |                |
| 实验教室 | 丹青 922               | 实验日期 | 2023 年 5 月 6 日 |
| 学 号  | 2021214893           | 姓 名  | 任平实            |
| 专业班级 | 奥林 21 级计算机科学与技术 04 班 |      |                |
| 指导教师 | 卢洋                   |      |                |

东北林业大学  
信息与计算机科学技术实验中心

一、实验目的

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

## 二、实验环境

(1) 计算机的硬件配置 PC 系列微机。

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

## 三、实验内容及结果

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

```
#!/bin/bash
mv *.c ~/abc/
~
~
~
~
~
~

[rocky@localhost ~]$ mkdir asa
[rocky@localhost ~]$ cd asa
[rocky@localhost asa]$ touch a.c
[rocky@localhost asa]$ touch b.c
[rocky@localhost asa]$ touch c.c
[rocky@localhost asa]$ ls
a.c b.c c.c
[rocky@localhost asa]$ vim cp1.sh
[rocky@localhost asa]$ cd

[rocky@localhost ~]$ mkdir abc
[rocky@localhost ~]$ cd asa
[rocky@localhost asa]$ bash cp1.sh
[rocky@localhost asa]$ ls
cp1.sh
[rocky@localhost asa]$ cd
[rocky@localhost ~]$ cd abc
[rocky@localhost abc]$ ls
a.c b.c c.c
[rocky@localhost abc]$ █
```

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
count=0
sum=0
num=1
while (( count<10 ))
do
    if (( num % 2 == 0))
    then
        sum=$(( ${sum} + ${num} ))
        count=$(( ${count} + 1 ))
    fi
    num=$(( ${num} + 1 ))
done
echo ${sum}
~
~
~
~

[rocky@localhost ~]$ vim evenSum.sh
[rocky@localhost ~]$ bash evenSum.sh
110
[rocky@localhost ~]$
```

## VIM

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

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
#!/bin/bash
num=1
count=0
until (( num > 10 ))
do
    count=$(( ${count} + num**2 ))
    num=$(( ${num} + 1 ))
done
echo $count
~
~
~
~
~
~

[rocky@localhost abc]$ vim pfSum.sh
[rocky@localhost abc]$ bash pfSum.sh \
>
385
[rocky@localhost abc]$
```

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。

文件(F) 编辑(E) 查看(V) 格式(O) 终端(T) 帮助(H)

#!/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
            break 2
        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
~
~
~
~
```

文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

```
#!/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
```

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

```
[rocky@localhost ~]$ vim evenSum.sh
[rocky@localhost ~]$ bash evenSum.sh
110
[rocky@localhost ~]$ vim q.sh
[rocky@localhost ~]$ bash evenSum.sh
110
[rocky@localhost ~]$ vim q.sh
[rocky@localhost ~]$ bash evenSum.sh
110
[rocky@localhost ~]$ vim q.sh
[rocky@localhost ~]$ bash evenSum.sh
110
[rocky@localhost ~]$ vim q.sh
[rocky@localhost ~]$ bash evenSum.sh
110
[rocky@localhost ~]$
```

#### 四、实验过程分析与讨论

C 语言事先没学过，通过看 ppt 和上网搜索得以解决

#### 五、指导教师意见

指导教师签字：卢洋

# 实验报告

|      |              |
|------|--------------|
| 实验名称 | 实验七 Shell 函数 |
|------|--------------|

|      |                      |      |                |
|------|----------------------|------|----------------|
| 实验教室 | 丹青 922               | 实验日期 | 2023 年 5 月 6 日 |
| 学 号  | 2021214893           | 姓 名  | 任平实            |
| 专业班级 | 奥林 21 级计算机科学与技术 04 班 |      |                |
| 指导教师 | 卢洋                   |      |                |

东北林业大学  
信息与计算机科学技术实验中心

一、实验目的

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

## 二、实验环境

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

## 三、实验内容及结果

1. 编写Shell脚本，实现一个函数，对两个数的和进行求解，并输出结果；

```
#!/bin/bash

numssum()
{
    read -p "please send me one num: " num1
    read -p "please send me other num: " num2
    echo "the sum is$(( ${num1} + ${num2} ))"
    return $(( ${num1} + ${num2} ))
}

numssum
echo $?
~
~
~
~
~

[rocky@localhost ~]$ vim sum0.sh
[rocky@localhost ~]$ bash sum0.sh
please send me one num: 2
please send me other num: 3
the sum is5
5
[rocky@localhost ~]$
```

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
echo "num"
read n
echo
func ()
{
    local i=$1
    if [ "$i" -eq 0 ]; then
        result=1
    else
        let "m=i-1"
        func "$m"
        let "result=$i * $"
    fi
    return $result
}
func "$n"
echo "$n jie cheng wei: $"
~
~
~

[rocky@localhost ~]$ vim jiecheng.sh
[rocky@localhost ~]$ bash jiecheng.sh
num
3
3 jie cheng wei: 6
```

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

```
#!/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 os om the first func."
    second
}

echo "starting..."
first
~
~
~

[rocky@localhost ~]$ vim www.sh
[rocky@localhost ~]$ bash www.sh
starting...
-1- here os om the first func.
-2- here is in the second func.
-3- here is in the third func.
[rocky@localhost ~]$
```

#### 四、实验过程分析与讨论

C 语言事先没学过，通过看 ppt 和上网搜索得以解决

#### 五、指导教师意见

指导教师签字：卢洋

# 实验报告

|      |               |      |                |
|------|---------------|------|----------------|
| 实验名称 | 实验八 sed 和 awk |      |                |
| 实验教室 | 丹青 922        | 实验日期 | 2023 年 5 月 6 日 |



|      |                      |     |     |
|------|----------------------|-----|-----|
| 学 号  | 2021214893           | 姓 名 | 任平实 |
| 专业班级 | 奥林 21 级计算机科学与技术 04 班 |     |     |
| 指导教师 | 卢洋                   |     |     |

东北林业大学  
信息与计算机科学技术实验中心

一、实验目的

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

## 二、实验环境

(1) 计算机的硬件配置 PC 系列微机。

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

## 三、实验内容及结果

1. 文件 quote.txt 的内容如下所示： 试使用 sed 命令实现如下功能：

```
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.
```

(1) 删除 \$ 符号；

```
[root@localhost rocky]# vim quote.txt  
[root@localhost rocky]# sed s?'$'?''?g 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.
```

```
[root@localhost rocky]# █
```

(2) 显示包含 music 文字的行内容及行号；

```
[root@localhost rocky]# nl quote.txt | sed -n '/music/p'  
2 It was an evening of splendid music and company.  
[root@localhost rocky]#
```

(3) 在第 4 行后面追加内容：“hello world!”；

```
[root@localhost rocky]# sed '4a hello world!' 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.  
hello world!
```

(4) 将文本“The”替换为“Quod”；

```
[root@localhost rocky]# sed 's/The/Quod/g' quote.txt  
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.
```

```
[root@localhost rocky]# █
```

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

```
[root@localhost rocky]# sed '3c This is the third line.' quote.txt
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.

[root@localhost rocky]#
```

(6) 删除第 2 行内容;

```
[root@localhost rocky]# sed '2d' quote.txt
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.

[root@localhost rocky]#
```

(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@localhost rocky]# sed -n "/${var}/p" quote.txt
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;

```
one : two : three
four : five : six
```

```
[root@localhost rocky]# awk -F ':' '{print $2}' numbers.txt
two
five
[root@localhost rocky]#
[root@localhost rocky]# awk -F ' ' '{print $2}' numbers.txt
:
:
[root@localhost rocky]#
```

4. 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。 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:

```
[root@localhost rocky]# vim foo.txt
[root@localhost rocky]# awk 'BEGIN{count=0} {for(i=1;i<=NF;i++) {if($i%2==0){print $i; count +=1}} } END{print count}' foo.txt
2
4
46
3
[root@localhost rocky]#
```

5. 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

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

56t
gfh
hyt
jyf
gsfr
jytg
kyyyy
ye

hfhdh
hdfh
bvc
t7yyyyyyyy
nmb
uyg
cfh
sgd
zcx
yjfg
rtyyyy
455e4terecvr
w3

:wq
```

```
[root@localhost rocky]# vim info.txt
[root@localhost rocky]# bash sss.sh
enter search pattern: 3
13
w3
2 found.
[root@localhost rocky]#
```

#### 四、实验过程分析与讨论

对 sed 和 awk 不熟悉，通过搜索解决，仍需多加记忆

#### 五、指导教师意见

指导教师签字：卢洋