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	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 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

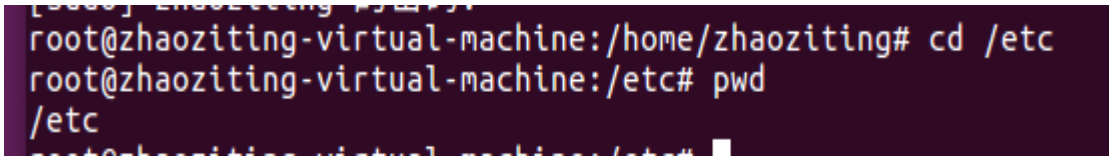
二、 实验环境

（1）计算机的硬件配置 PC 系列微机。

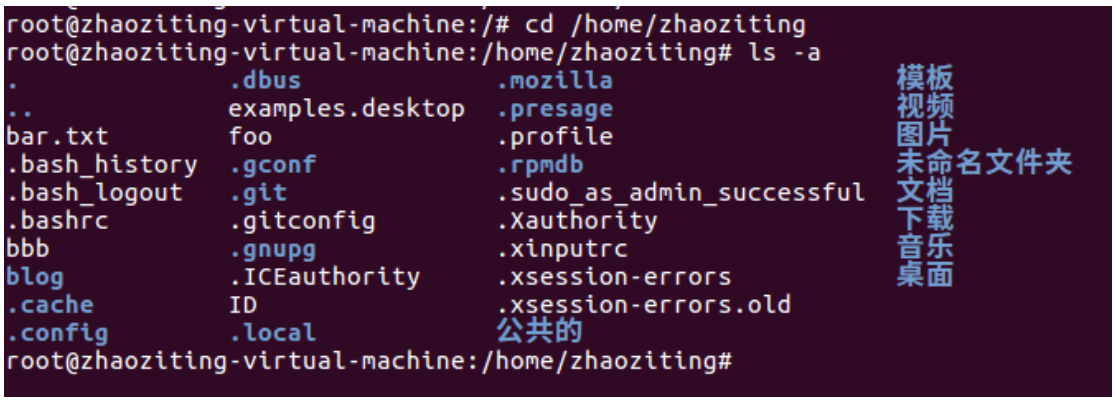
（2）计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

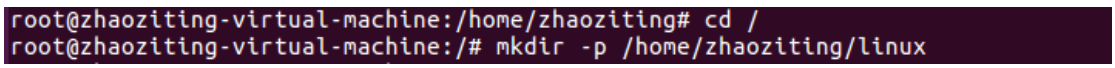
1. 使用命令切换到/etc 目录，并显示当前工作目录路径

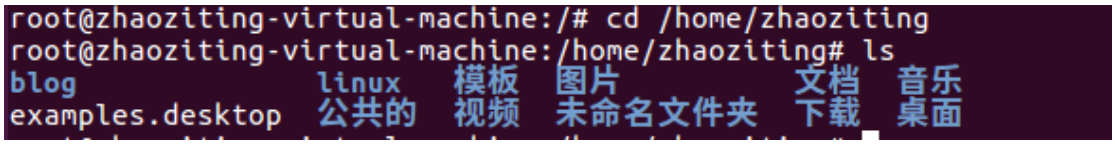


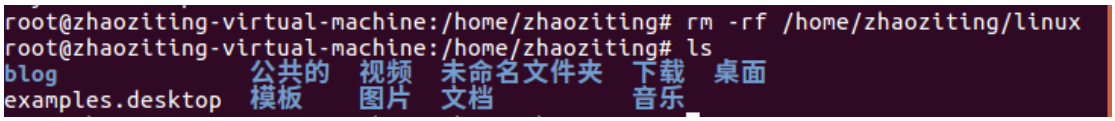
2、使用命令显示/home/lyj 目录下所有文件目录的详细信息，包括隐藏文件。



3、使用命令创建目录/home/lyj/linux，然后删除该目录。







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

```
root@zhaoziting-virtual-machine:/home/zhaoziting# cat > abc
hello linux
```

```
root@zhaoziting-virtual-machine:/home/zhaoziting# cat abc
hello linux
```

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

```
root@zhaoziting-virtual-machine:/home/zhaoziting# mkdir ak
root@zhaoziting-virtual-machine:/home/zhaoziting# cp -r abc ak
root@zhaoziting-virtual-machine:/home/zhaoziting# ls
abc  blog              公共的  视频  未命名文件夹  下载  桌面
ak   examples.desktop  模板    图片  文档          音乐
```

```
root@zhaoziting-virtual-machine:/home/zhaoziting/ak# rm abc
```

```
root@zhaoziting-virtual-machine:/home/zhaoziting# rm -rf 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。

```
root@zhaoziting-virtual-machine:/# cat /etc/adduser.conf | head -n 3
# /etc/adduser.conf: `adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.

root@zhaoziting-virtual-machine:/#
```

```
root@zhaoziting-virtual-machine:/# cat /etc/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
root@zhaoziting-virtual-machine:/#
```

7、分屏查看文件/etc/adduser.conf 的内容。

```
root@zhaoziting-virtual-machine:/# more /etc/adduser.conf
# /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.
--更多--(25%)
```

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

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 。

```
root@zhaoziting-virtual-machine:/home/zhaoziting# cat > bar.txt
google 110 5000
baidu 100 5000
guge 50 3000
sohu 100 4500
```

利用sort命令完成下列排序：

（1） 按公司字母顺序排序

```
root@zhaoziting-virtual-machine:/home/zhaoziting# sort -t ' ' -k 1 bar.txt
baidu 100 5000
google 110 5000
guge 50 3000
sohu 100 4500
```

（2） 按公司人数排

```
root@zhaoziting-virtual-machine:/home/zhaoziting# sort -n -t ' ' -k 2 bar.txt
guge 50 3000
baidu 100 5000
sohu 100 4500
google 110 5000
```

（3） 按公司人数排序，人数相同的按照员工平均工资升序排序

```
root@zhaoziting-virtual-machine:/home/zhaoziting# sort -n -t ' ' -k 2 -k 3 bar.t
xt
guge 50 3000
sohu 100 4500
baidu 100 5000
google 110 5000
```

（4） 按员工工资降序排序，如工资相同，则按公司人数升序排序

```
root@zhaoziting-virtual-machine:/home/zhaoziting# sort -n -t ' ' -k 3r -k 2 bar
txt
baidu 100 5000
google 110 5000
sohu 100 4500
guge 50 3000
```

（5）从公司英文名称的第2个字母开始进行排序。

```
root@zhaoziting-virtual-machine:/home/zhaoziting# sort -t ' ' -k 1.2 bar.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	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 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；

```
root@zhaoziting-virtual-machine:/home/zhaoziting# mkdir -p baz
root@zhaoziting-virtual-machine:/home/zhaoziting# ls
abc       baz   examples.desktop  模板  图片          文档  音乐
bar.txt   blog  公共的            视频  未命名文件夹  下载  桌面
root@zhaoziting-virtual-machine:/home/zhaoziting/baz# cat > qux
zzt
20021101
2021210815
^C
```

（2）在用户目录下查找文件qux，并显示该文件位置信息；

```
root@zhaoziting-virtual-machine:/home/zhaoziting# find ~ / -name "qux"
/home/zhaoziting/baz/qux
```

（3）统计文件qux中所包含内容的行数、字数和字节数；

```
root@zhaoziting-virtual-machine:/home/zhaoziting/baz# wc qux
 3  3 24 qux
```

（4）在用户目录下查找文件qux，并删除该文件；

```
root@zhaoziting-virtual-machine:/home/zhaoziting/baz# find ~ / -name "qux" -type
 f -delete
```

（5）查看文件夹baz内容，看一下是否删除了文件qux。

```
root@zhaoziting-virtual-machine:/home/zhaoziting/baz# ls
root@zhaoziting-virtual-machine:/home/zhaoziting/baz# cat qux
cat: qux: 没有那个文件或目录
```

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；

```
root@zhaoziting-virtual-machine:/home/zhaoziting# mkdir -p path1
```

```
root@zhaoziting-virtual-machine:/home/zhaoziting# cd ~/path1
root@zhaoziting-virtual-machine:~/path1# touch file1 file2
root@zhaoziting-virtual-machine:~/path1# ls
file1  file2
```

（2）在用户目录下新建文件夹path2，在path2下新建文件file3；

```
root@zhaoziting-virtual-machine:~# mkdir -p path2
root@zhaoziting-virtual-machine:~# cd path2
root@zhaoziting-virtual-machine:~/path2# touch file3
root@zhaoziting-virtual-machine:~/path2# ls
file3
```

（3）在用户目录下新建文件file4；

```
root@zhaoziting-virtual-machine:~# touch file4
root@zhaoziting-virtual-machine:~# ls
file4  path1  path2
```

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

```
root@zhaoziting-virtual-machine:~# tar -cf  ./package.tar ./path1 ./file4
```

（5）查看包package.tar的内容；

```
root@zhaoziting-virtual-machine:~# 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；

```
root@zhaoziting-virtual-machine:~# tar -rf package.tar path2
root@zhaoziting-virtual-machine:~# tar -tf package.tar
./path1/
./path1/file2
./path1/file1
./file4
path2/
path2/file3
```

（7）将包package.tar复制到用户目录下的新建文件夹path3中；

```
root@zhaoziting-virtual-machine:~# mkdir ./path3
root@zhaoziting-virtual-machine:~# cp package.tar ./path3
```

（8）进入path3文件夹，并还原包package.tar的内容。

```
root@zhaoziting-virtual-machine:~# cd path3
root@zhaoziting-virtual-machine:~/path3# tar -xf package.tar
```

3. 符号链接内容

（1）新建文件foo.txt，内容为123；

```
root@zhaoziting-virtual-machine:~# echo "123" > 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；

```
root@zhaoziting-virtual-machine:~# ln foo.txt bar.txt
root@zhaoziting-virtual-machine:~# comm foo.txt bar.txt
                123
root@zhaoziting-virtual-machine:~# diff foo.txt bar.txt
```

（3）查看foo.txt和bar.txt的i节点号（inode）是否相同；

```
root@zhaoziting-virtual-machine:~# ls -i foo.txt bar.txt
417837 bar.txt  417837 foo.txt
```

（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；

```
root@zhaoziting-virtual-machine:~# echo "abc" > bar.txt
root@zhaoziting-virtual-machine:~# cmp -s foo.txt bar.txt
```

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

```
root@zhaoziting-virtual-machine:~# rm foo.txt
root@zhaoziting-virtual-machine:~# ls -i bar.txt
417837 bar.txt
root@zhaoziting-virtual-machine:~# cat bar.txt
123
```

（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；

```
root@zhaoziting-virtual-machine:~# ln -s bar.txt baz.txt
root@zhaoziting-virtual-machine:~#
root@zhaoziting-virtual-machine:~# ls -i bar.txt baz.txt
417837 bar.txt  417874 baz.txt
root@zhaoziting-virtual-machine:~# cmp -s bar.txt baz.txt
```

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

```
root@zhaoziting-virtual-machine:~# rm bar.txt
root@zhaoziting-virtual-machine:~# cat baz.txt
cat: baz.txt: 没有那个文件或目录
```

4. 权限管理

（1）新建文件qux.txt；

```
root@zhaoziting-virtual-machine:~# touch qux.txt
root@zhaoziting-virtual-machine:~# ls
baz.txt  file4  ~package.tar  package.tar  path1  path2  path3  qux.txt
```

（2）为文件qux.txt增加执行权限（所有用户都可以执行）。

```
root@zhaoziting-virtual-machine:~# chmod +x qux.txt
```

四、 实验过程分析与讨论

遇到的困难在 3 那个实验，没有搞清楚链接与符文链接的区别，查询 csdn 后搞清楚了。

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验三 vim 编辑器及 gcc 编译器的使用		
实验教室	丹青 922	实验日期	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 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。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；

```
root@zhaoziting-virtual-machine:~# mkdir ./workspace1
```

（2）进入目录workspace1；

```
root@zhaoziting-virtual-machine:~# cd workspace1
root@zhaoziting-virtual-machine:~/workspace1#
```

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

```
#include <stdio.h>
intmain( )
{
```

printf(“helloworld!\n”);

return0;

}

```
root@zhaoziting-virtual-machine:~/workspace1# vim text.c
```

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

（4）保存test.c的内容，并退出；

```
:wq
```

（5）编译test.c文件，生成可执行文件test，并执行，查看执行结果。

编译test.c文件，生成可执行文件test，并执行，查看执行结果。

```
root@zhaoziting-virtual-machine:~/workspace1# gcc text.c -o text
root@zhaoziting-virtual-machine:~/workspace1# ./text
hello world!
```

2. vim编辑器的详细使用：

（1）在用户目录下创建一个名为workspace2的目录；

```
root@zhaoziting-virtual-machine:~/workspace1# mkdir ./workspace2
```

（2）进入workspace2目录；

```
root@zhaoziting-virtual-machine:~/workspace1# cd workspace2
root@zhaoziting-virtual-machine:~/workspace1/workspace2#
```

（3）使用以下命令：cat /etc/gai.conf > ./gai.conf

将文件/etc/gai.conf的内容复制到当前目录下的新建文件

gai.conf中；

```
root@zhaoziting-virtual-machine:~/workspace1/workspace2# cat /etc/gai.conf > ./g
au.conf
```

（4）使用vim编辑当前目录下的gai.conf；

```
root@zhaoziting-virtual-machine:~/workspace1/workspace2# vim gai.conf
```

（5）将光标移到第18行；

```
# label   <mask>   <value>
#    Add another rule to the RFC 3484 label table.  See section 2.1 in
#    RFC 3484.  The default is:
#
#label ::1/128       0
#label ::/0          1
:18                                                    18,1          顶端
```

（6）复制该行内容；

yy

（7）将光标移到最后一行行首；

```
#scopev4 ::ffff:0.0.0.0/96       14
                                                       65,1          底端
```

G

（8）粘贴复制行的内容；

```
#scopev4 ::ffff:127.0.0.0/104    2
#scopev4 ::ffff:0.0.0.0/96       14
#scopev4 ::ffff:0.0.0.0/96       14
                                                       66,1          底端
```

（9）撤销第8步的动作；

```
#scopev4 ::ffff:127.0.0.0/104    2
#scopev4 ::ffff:0.0.0.0/96       14
~
1 行被去掉；before #1  55 秒之前                       65,1          底端
```

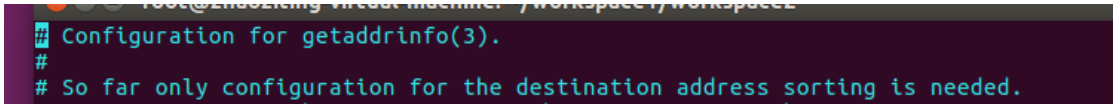
（10）存盘但不退出；

:w

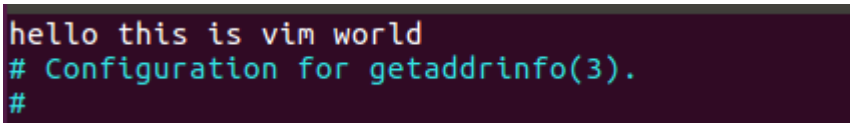
```
#scopev4 ::ffff:0.0.0.0/96       14
~
"gai.conf" 65L, 2584C 已写入                           65,1          底端
```

（11）将光标移到首行；

gg



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



（13）删除字符串"this"；

:%s/this//g

（14）强制退出vim，不存盘。

四、 实验过程分析与讨论

在 vim 中删除命令不太熟悉，查询后发现在正常模式下，使用命令删除字符串：diw

上述命令中的 d 是删除命令，iw 是文本对象，表示删除光标所在位置的一个单词。

使用替换命令删除所有匹配项：

在正常模式下，使用以下命令删除所有的字符串 "this"：

:%s/this//g

上述命令中的 :%s/this//g 是替换命令的格式。其中 :%s 表示在整个文件范围内进行替换，this 是要替换的字符串，// 表示将字符串替换

为空，g 表示全局替换，即删除所有匹配项。

这样可以把全文的 this 删除

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	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 班		
指导教师	卢洋		

东北林业大学

信息与计算机科学技术实验中心

一、 实验目的

1. 掌握用户管理命令，包括命令 useradd、usermod、userdel、newusers；

2. 掌握用户组管理命令，包括命令 groupadd、groupdel、gpasswd；

3. 掌握用户和用户组维护命令，包括命令 passwd、su、sudo。

二、 实验环境

（1）计算机的硬件配置 PC 系列微机。

（2）计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

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

```
root@zhaoziting-virtual-machine:~# sudo useradd -m -d /home/foo -s /bin/sh -c "b
ar" -p$(openssl passwd -1 123456) foo
```

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@zhaoziting-virtual-machine:~# sudo su - foo
```

3. 从用户foo切换到root；

```
sudo su -
```

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@zhaoziting-virtual-machine:/home/zhaoziting# sudo 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@zhaoziting-virtual-machine:/home/zhaoziting# sudo nano users.txt
root@zhaoziting-virtual-machine:/home/zhaoziting# sudo newusers < users.txt
```

```
root@zhaoziting-virtual-machine:/home/zhaoziting# sudo nano users.txt
root@zhaoziting-virtual-machine:/home/zhaoziting# sudo newusers < users.txt
```

```
root@zhaoziting-virtual-machine:/home/zhaoziting# sudo chpasswd < users.txt
```

```
root@zhaoziting-virtual-machine:/home/zhaoziting# cat /etc/password
```

6. 创建用户组group1，并在创建时设置其GID为3000；

```
root@zhaoziting-virtual-machine:/# sudo groupadd -g 3000 group1
```

7. 在用户组group1中添加两个之前批量创建的用户；

```
root@zhaoziting-virtual-machine:/# sudo usermod -a -G group1 user1
root@zhaoziting-virtual-machine:/# sudo usermod -a -G group1 user2
```

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

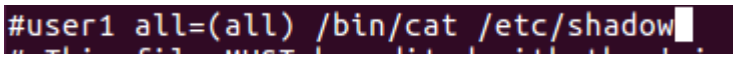
```
root@zhaoziting-virtual-machine:/# sudo su - user1
```

```
sudo cat /etc/shadow
```

若不行的话

说明目标用户没有足够的权限。为了允许该用户通过 sudo 查看 /etc/shadow 文件，可以修改 sudoers 文件。

sudo visudo命令会打开 sudoers 文件，并使用默认的文本编辑器进行编辑。在文件中添加以下行：



四、 实验过程分析与讨论

在批量创建用户的时候不会这个操作，查询资料后发现首先要

创建一个包含用户信息的文本文件 users.txt，每行包含一个用户的信息，格式为 <username>:<password>，按照以下步骤进行操作：

使用任意文本编辑器（如 vi、nano、gedit 等）创建并打开 users.txt 文件。

在文件中，每行表示一个用户的信息，按照 <username>:<password> 的格式输入用户名和密码，并确保每个用户的信息占据单独的一行。

例如，下面是一个示例的 users.txt 文件内容：

user1:password1

user2:password2

user3:password3

可根据需要添加更多的用户信息。

保存并关闭文件。

确保在每行中使用冒号 ：分隔用户名和密码，并确保用户名和密码不包含冒号。

完成上述步骤后，就可以使用 newusers 命令和 chpasswd 命令来批量创建用户并设置密码。一开始不清楚这一步，所以导致无法进行。

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	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 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）；

```
foo=200
root@zhaoziting-virtual-machine:/# 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）；

```
bar=100
root@zhaoziting-virtual-machine:/# test $bar -gt 150
root@zhaoziting-virtual-machine:/# echo $?
2
```

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@zhaoziting-virtual-machine:/# vim display_info.sh
```

```
#!/bin/bash
hostname=$(hostname)
current_date=$(date)
echo "Hostname: $hostname"
echo "System Time: $current_date"
~
~
```

```
root@zhaoziting-virtual-machine:/# chmod +x display_info.sh
```

```
root@zhaoziting-virtual-machine:/# ./display_info.sh
Hostname: zhaoziting-virtual-machine
System Time: 2023年 05月 23日 星期二 19:57:03 CST
```

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

所谓水仙花数是指一个3位数，该数字每位数字的3次幂之和等于其本身，例如：

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。

```
root@zhaoziting-virtual-machine:/# vim check_n_number.sh
```

```
#!/bin/bash
if [$# -eq 0]; then
echo "mycs"
exit 1
fi
for num in "$@"; do
if [ ${#num} -ne 3 ]; then
echo "$num bushiyigesanweishu"
continue
fi

digit1=$((num / 100))
digit2=$(((num / 10) % 10 ))
digit3=$((num % 10))
sum=$((digit1 ** 3 + digit2 ** 3 + digit3 **
if [ $sum -eq $num ]; then
echo "$num shi"
else
echo "$num buhsi"
fi
done

~
```

```
root@zhaoziting-virtual-machine:/# chmod +x check_n_number.sh
root@zhaoziting-virtual-machine:/# ./check_n_number.sh 153 124 370
```

```
153 shi
124 buhsi
370 shi
```

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；

```
root@zhaoziting-virtual-machine:/# vim sum.sh
```

```
#!/bin/bash
if [$# -ne 3]; then
   echo " shu ru san ge can shu"
   exit 1
fi

num1=$1
num2=$2
num3=$3
sum=$((num1 + num2 + num3))

echo "hewei: $sum"
~
```

```
root@zhaoziting-virtual-machine:/# bash sum.sh 10 20 30
sum.sh: 行 2: [3: 未找到命令
hewei: 60
```

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。

if

elif

else

fi

```
root@zhaoziting-virtual-machine:/# vim grade.sh
```

```
#!/bin/bash
read -p score
score=$(echo "$score" | bc)
if ((score >= 90)); then
   grade="A"
elif ((score >= 80)); then
   grade="B"
elif ((score >= 70)); then
   grade="C"
elif ((score >= 60)); then
   grade="D"
else
   grade="E"
fi
echo "$score $grade"
```

```
root@zhaoziting-virtual-machine:/# bash grade.sh
score50
 E
```

四、 实验过程分析与讨论

在最后一个实验里一直输出错误，发现是因为在 Shell 脚本中，用户输入的数值是以字符串的形式读取的，需要将其转换为整数类型才能进行比较。在修正后的代码中，使用了正确的逻辑运算符 (( )) 来进行数值比较才能得出正确的结果。

五、指导教师意见

指导教师签字：卢洋

实验报告

实验名称	实验六 Shell 循环控制语句		
实验教室	丹青 922	实验日期	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 班		
指导教师	卢洋		

东北林业大学

信息与计算机科学技术实验中心

一、 实验目的

1. 熟练掌握 Shell 循环语句：for、while、until；

2. 熟练掌握 Shell 循环控制语句：break、continue。

二、 实验环境

（1）计算机的硬件配置 PC 系列微机。

（2）计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 编写一个Shell脚本，利用for循环把当前目录下的所有*.c文件复制到指定的目录中（如~/workspace）；

可以事先在当前目录下建立若干*.c文件用于测试。

```
root@zhaoziting-virtual-machine:/# vim  copy_c_files.sh
```

```
#!/bin/bash
destination_dir=~/workspace
for file in *.c; do
 if [[ -f "$file" ]]; then
cp "$file" "$destination_dir"
fi
done
echo "wancheng"
```

```
root@zhaoziting-virtual-machine:/# bash  copy_c_files.sh
wancheng
```

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=1   # 当前偶数计数
sum=0     # 偶数之和

while [[ $count -le 10 ]]; do
    # 判断当前数字是否为偶数
    if (( count % 2 == 0 )); then
        sum=$((sum + count))  # 累加偶数到总和中
    fi

    count=$((count + 1))  # 增加当前偶数计数
done

echo "前10个偶数之和为: $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；

```bash
#!/bin/bash

sum=0
num=1

until [ $num -gt 10 ]; do
    square=$((num * num))
    sum=$((sum + square))
    num=$((num + 1))
done

echo "1到10的平方和为: $sum"
```

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

2、continue、continue 2，并观察四种情况下的实验结果。

```
#!/bin/bash
for i in a b c d; do
echo-n$i
for j in12345678910; do
if [[ $j-eq5 ]]; then
---
fi
echo -n $j
```

done

echo''

done

替换为 break：

程序在内部循环遍历到 j 等于 5 时，执行 break 语句，终止内部循环。因此，输出结果为：

```
a1234
b1234
c1234
d1234
```

替换为 break 2：

程序在内部循环遍历到 j 等于 5 时，执行 break 2 语句，终止外部和内部循环。因此，输出结果为：

```
a
```

替换为 continue：

程序在内部循环遍历到 j 等于 5 时，执行 continue 语句，跳过该次内部循环的剩余部分，继续下一次迭代。因此，输出结果为：

```
a1234678910
b1234678910
c1234678910
d1234678910
```

替换为 continue 2：

程序在内部循环遍历到 j 等于 5 时，执行 continue 2 语句，跳过该次内部循环的剩余部分，并跳过外部循环的剩余部分，继续下一次迭代。因此，输出结果为：

```
a
```

四、 实验过程分析与讨论

对于最后的问题观察结果得出：

根据不同的替换，程序的执行结果会有所不同。break 语句用于终止当前循环，break 2 语句用于终止外部和内部循环，continue 语句用于跳过当前迭代的剩余部分，continue 2 语句用于跳过外部和内部循环的剩余部分。

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	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 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；

```shell
#!/bin/bash

# 定义一个函数，接收两个参数，并计算它们的和
calculate_sum() {
    local num1=$1
    local num2=$2
    local sum=$((num1 + num2))
    echo "Sum: $sum"
}

# 调用函数，传递两个数值参数
calculate_sum 10 20
```

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；

```shell
#!/bin/bash

# 定义递归函数，计算n的阶乘
factorial() {
    local n=$1
    if [ $n -eq 0 ]; then
        echo 1
    else
        local prev=$((n - 1))
        local prev_factorial=$(factorial $prev)
        local result=$((n * prev_factorial))
        echo $result
    fi
}

# 调用函数计算阶乘，并输出结果
result=$(factorial 5)
echo "Factorial of 5 is: $result"
```

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 is in the first func."
    second
}
echo "starting..."
first
```

以上的Shell脚本定义了三个嵌套的函数first、second和third，并在主程序中依次调用这些函数。程序运行时会输出一系列信息来表示当前所在的函数。

当脚本运行时，它会按照函数的嵌套顺序逐步执行，从first函数开始，然后进入second函数，最后进入third函数。每次进入一个新的函数时，都会输出相应的提示信息。

输出结果如下：

```
starting...
-1- here is in the first func.
-2- here is in the second func.
-3- here is in the third func.
```

四、 实验过程分析与讨论

做实验时对嵌套函数不是很熟悉，查阅资料更深入的了解了嵌套函数。

函数嵌套是指在一个函数内部定义另一个函数的过程，即函数内部包含了另一个函数的定义。这样的设计允许在一个函数中调用其他函数，并可以创建多层次的函数调用关系。

函数嵌套的含义和作用包括以下几点：

代码结构清晰：函数嵌套可以将复杂的程序逻辑分解为多个小的函数模块，使代码结构更加清晰和可读。每个函数负责完成特定的任务，提高了代码的可维护性和可扩展性。

代码复用：通过函数嵌套，可以将通用的功能封装在独立的函数中，并在不同的上下文中重复使用。这样可以减少代码的重复编写，提高开发效率。

封装性和作用域：函数嵌套可以实现变量和函数的封装，确保内部函数中的变量和函数不会与外部函数或全局作用域中的同名变量和函数产生冲突。内部函数可以访问外部函数的变量和函数，但外部函数无法直接访问内部函数的变量和函数。

递归调用：函数嵌套还可以用于实现递归调用，即函数在自身内部调用自身。这种方式可以解决一些需要重复执行相似操作的问题，如计算阶乘、斐波那契数列等。

需要注意的是，在函数嵌套中，内部函数的作用范围仅限于外部函数的内部，外部函数无法直接调用内部函数。要调用内部函数，通常需

要通过外部函数的调用间接访问。

函数嵌套是 Shell 脚本中常用的技巧，它可以提高代码的模块化和可读性，并支持更灵活的程序设计。

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	年 月 日
学　　号	2021210815	姓　　名	赵梓廷
专业班级	计算机科学与技术 03 班		
指导教师	卢洋		

东北林业大学

信息与计算机科学技术实验中心

一、 实验目的

1. 掌握 sed 基本编辑命令的使用方法；

2. 掌握 sed 与 Shell 变量的交互方法；

3. 掌握 awk 命令的使用方法；

4. 掌握 awk 与 Shell 变量的交互方法。

二、 实验环境

（1）计算机的硬件配置 PC 系列微机。

（2）计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

三、 实验内容及结果

1. 文件quote.txt的内容如下所示：

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

It was an evening of splendid music and company.

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

The local nurse Miss P.Neave was in attendance.

试使用sed命令实现如下功能：

（1） 删除$符号；

```
sed 's/\$//g' quote.txt
```

（2） 显示包含music文字的行内容及行号；

```
grep -n "music" quote.txt
```

（3） 在第4行后面追加内容："hello world!"；

```
sed -i '4a\hello world!' quote.txt
```

（4） 将文本"The"替换为"Quod"；

```
sed 's/The/Quod/g' quote.txt
```

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

```
sed '3s/.*/This is the third line./' quote.txt
```

（6） 删除第2行内容；

```
sed -i '2d' quote.txt
```

（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。

```
var="evening"
sed -n "/$var/p" quote.txt | sed '/$var/{s//\033[1;31m&\033[0m/g}'
```

2. 文件numbers.txt的内容如下所示：

one : two : three

four : five : six

注：每个冒号前后都有空格。

试使用awk命令实现如下功能：分别以空格和冒号做分隔符，显示第2列的内容，观察两者的区别；

以空格为分隔符：

```
awk '{print $2}' numbers.txt
```

以冒号为分隔符：

```
awk -F: '{print $2}' numbers.txt
```

这两个命令分别以空格和冒号作为字段分隔符，然后打印每行的第2个字段内容。第2列是从1开始计数的，所以使用$2来表示第2个字段。

两者的区别在于分隔符的不同，第一个命令使用默认的空格作为分隔符，而第二个命令通过 -F 参数指定冒号作为分隔符。根据输入文件的格式，选择适当的分隔符来获取第2列的内容。

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

foo.txt中的所有偶数进行打印，并输出偶数的个数。

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

例如：foo.txt内容为：

2 4 3

15 46 79

则输出为：

even:

2

4

46

numbers:

3

```
#!/bin/bash

even_count=0  # 用于统计偶数的个数

while read -r line; do  # 逐行读取文件内容
    numbers=($line)  # 将每行内容拆分成数组

    for num in "${numbers[@]}"; do  # 遍历数组中的数字
        remainder=$(($num % 2))  # 计算数字的余数

        if [[ $remainder -eq 0 ]]; then  # 判断余数是否为0，即偶数
            echo "even:"
            echo $num
            even_count=$((even_count + 1))  # 偶数计数器加1
        fi
    done

    echo "numbers:"
    echo "${numbers[@]}"  # 输出当前行的所有数字
done < foo.txt

echo "Total even numbers: $even_count"  # 输出偶数的总个数
```

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。

#!/bin/bash

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

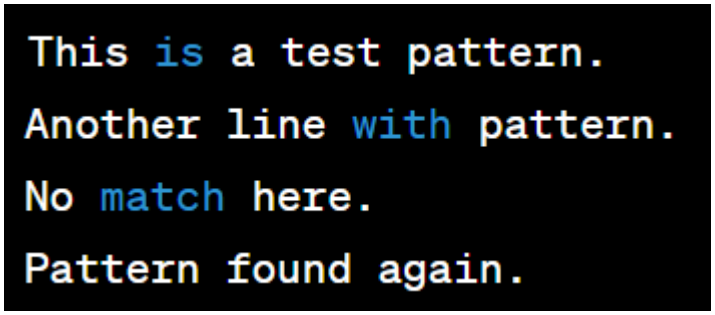
这段脚本实现了在文件`info.txt`中搜索指定模式的文本，并输出匹配到的行以及匹配行的总数。

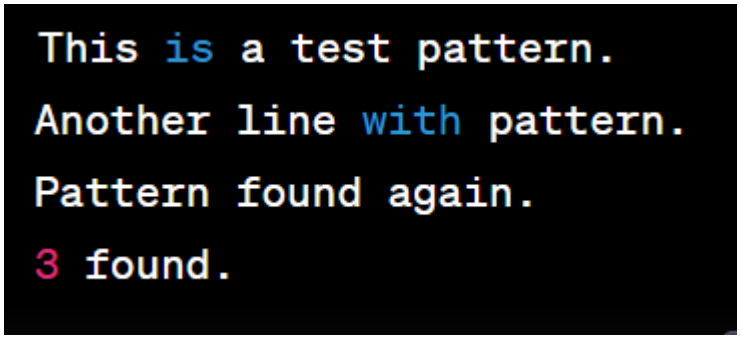
脚本的执行步骤如下：

1. 使用`read`命令提示用户输入搜索模式，并将用户输入的值保存在变量`pattern`中。
2. 使用`awk`命令对文件`info.txt`进行处理。
3. 在`awk`命令中使用双引号括起来的部分`"/$pattern/"`表示匹配模式，即搜索用户输入的模式。
4. 当匹配到的行被处理时，递增变量`nmatches`的值，并打印匹配到的行。
5. 在`END`部分，输出变量`nmatches`的值以及字符串`"found."`，表示匹配到的行的总数。

用户可以在脚本运行时输入自己想要搜索的模式，脚本会在文件`info.txt`中查找包含该模式的行，并输出这些行以及匹配行的总数。

注意：脚本中的`info.txt`是一个示例文件名，实际运行时需要替换为有效的文件名，并确保该文件存在且具有读取权限。





四、 实验过程分析与讨论

对于 set 的和 awk 的理解不够，查阅资料对它们产生更加清楚的理解。

在 Shell 脚本中，`set`命令用于设置或显示 Shell 的内部变量的值，而`awk`命令是一种强大的文本处理工具。这两者可以通过使用管道（`|`）实现交互。

当使用`set`命令设置 Shell 内部变量的值时，可以使用管道将变量的值传递给`awk`命令进行进一步的处理和操作。`awk`命令可以通过读取标准输入（stdin）来获取输入数据，并根据指定的模式和动作来处理和操作这些数据。

下面是一个示例，展示了如何使用`set`、`awk`和 Shell 变量的交互：

```
#!/bin/bash

# 设置Shell变量
name="John"
age=30

# 使用set命令显示Shell变量的值，并通过管道将其传递给awk命令进行处理
set | awk -F '=' '{print "Variable:", $1, "Value:", $2}'
```

在上面的示例中，首先通过`set`命令显示了当前 Shell 中的所有变量及其值。然后使用管道将`set`命令的输出传递给`awk`命令。在`awk`命令中，使用等号（`=`）作为字段分隔符（`-F '='`），并打印每个变量的名称和对应的值。

执行上述脚本后，将输出当前 Shell 中所有变量的名称和值，类似于以下内容：

```
Variable: name Value: John
Variable: age Value: 30
```

通过这种方式，我们可以在 Shell 脚本中将变量的值传递给`awk`命令进行进一步处理，例如根据特定条件筛选数据、进行计算等操作。

五、指导教师意见

指导教师签字：卢洋