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 3 月 22 日
学　　号	2021223124	姓　　名	张颖
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

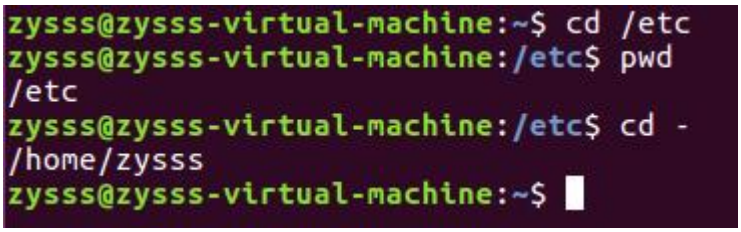
二、 实验环境

（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



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

```
zysss@zysss-virtual-machine:~$ ll
总用量 108
drwxr-xr-x 17 zysss zysss 4096 5月  22 10:30 ./
drwxr-xr-x  3 root  root  4096 4月  14 22:42 ../
-rw-r--r--  1 zysss zysss  220 4月  14 22:42 .bash_logout
-rw-r--r--  1 zysss zysss 3771 4月  14 22:42 .bashrc
drwx------ 12 zysss zysss 4096 5月  22 10:30 .cache/
drwx------ 19 zysss zysss 4096 4月  14 22:58 .config/
drwx------  3 zysss zysss 4096 4月  14 22:53 .dbus/
-rw-r--r--  1 zysss zysss 8980 4月  14 22:42 examples.desktop
drwx------  2 zysss zysss 4096 4月  15 18:28 .gconf/
drwx------  3 zysss zysss 4096 5月  22 09:52 .gnupg/
-rw-------  1 zysss zysss  756 5月  22 10:30 .ICEauthority
drwx------  3 zysss zysss 4096 4月  14 22:53 .local/
drwx------  2 zysss zysss 4096 4月  14 22:53 .presage/
-rw-r--r--  1 zysss zysss  655 4月  14 22:42 .profile
-rw-r--r--  1 zysss zysss    0 4月  14 22:59 .sudo_as_admin_successful
-rw-------  1 zysss zysss   66 5月  22 09:51 .Xauthority
-rw-------  1 zysss zysss   84 5月  22 09:52 .xsession-errors
-rw-------  1 zysss zysss   84 4月  14 22:55 .xsession-errors.old
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 公共的/
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 模板/
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 视频/
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 图片/
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 文档/
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 下载/
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 音乐/
drwxr-xr-x  2 zysss zysss 4096 4月  14 22:53 桌面/
zysss@zysss-virtual-machine:~$
```

3、使用命令创建目录/home/lyj/linux，然后删除该目录

```
zysss@zysss-virtual-machine:~$ ls
examples.desktop  公共的  模板  视频  图片  文档  下载  音乐  桌面
zysss@zysss-virtual-machine:~$ mkdir linux
zysss@zysss-virtual-machine:~$ ls
examples.desktop  linux  公共的  模板  视频  图片  文档  下载  音乐  桌面
zysss@zysss-virtual-machine:~$ rmdir linux
zysss@zysss-virtual-machine:~$ ls
examples.desktop  公共的  模板  视频  图片  文档  下载  音乐  桌面
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ cat >abc
hello linux!
zysss@zysss-virtual-machine:~$ cat abc
hello linux!
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ ls
abc  examples.desktop  公共的  模板  视频  图片  文档  下载  音乐  桌面
zysss@zysss-virtual-machine:~$ mv abc ak
zysss@zysss-virtual-machine:~$ cat ak
hello linux!
zysss@zysss-virtual-machine:~$
```

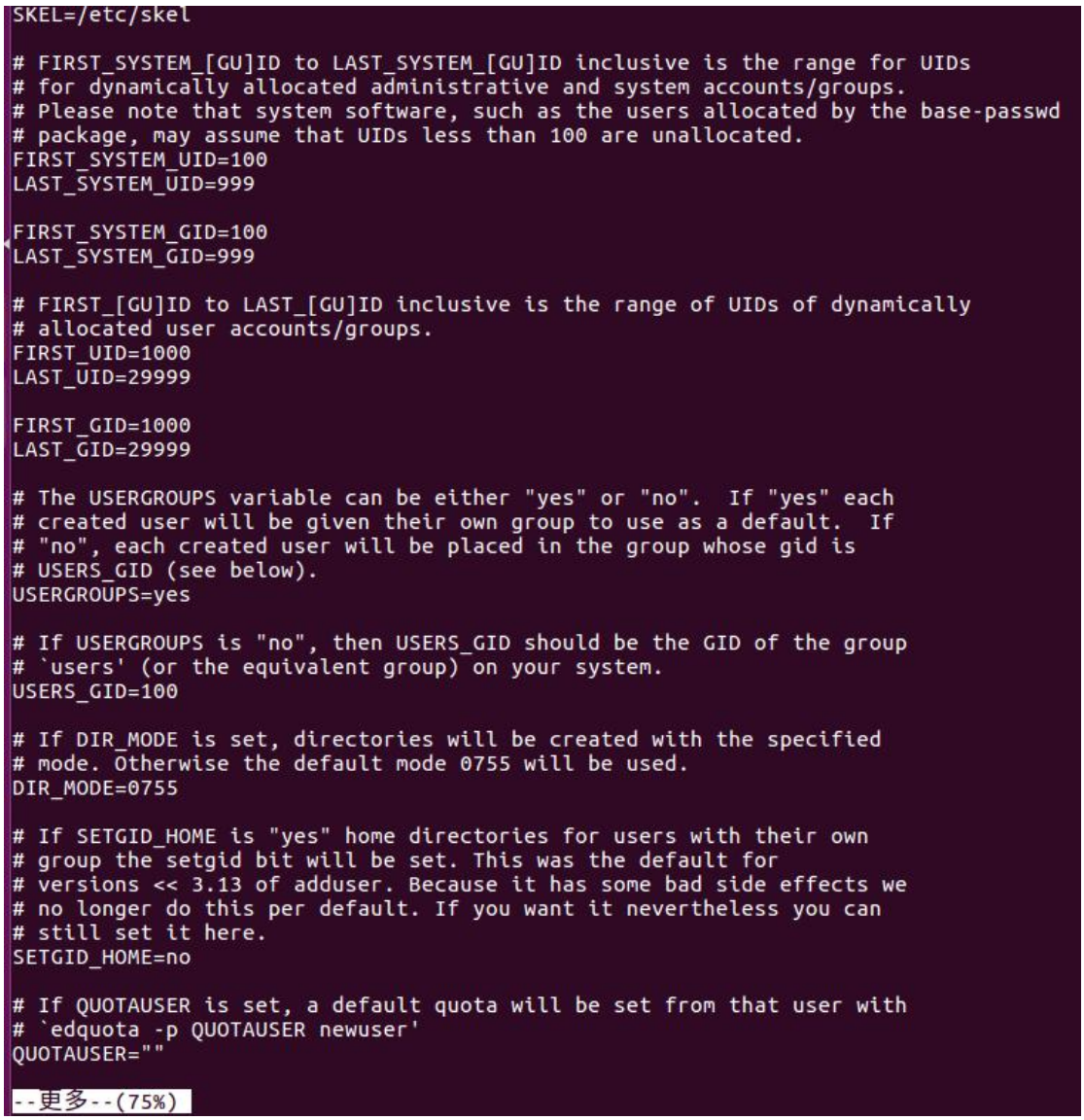
6、查看文件/etc/adduser.conf 的前 3 行内容，查看文/etc/adduser.conf 的最后 5 行内容

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

zysss@zysss-virtual-machine:~$ 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
zysss@zysss-virtual-machine:~$
```

7、分屏查看文件/etc/adduser.conf 的内容



```
SKEL=/etc/skel

# FIRST_SYSTEM_[GU]ID to LAST_SYSTEM_[GU]ID inclusive is the range for UIDs
# for dynamically allocated administrative and system accounts/groups.
# Please note that system software, such as the users allocated by the base-passwd
# package, may assume that UIDs less than 100 are unallocated.
FIRST_SYSTEM_UID=100
LAST_SYSTEM_UID=999

FIRST_SYSTEM_GID=100
LAST_SYSTEM_GID=999

# FIRST_[GU]ID to LAST_[GU]ID inclusive is the range of UIDs of dynamically
# allocated user accounts/groups.
FIRST_UID=1000
LAST_UID=29999

FIRST_GID=1000
LAST_GID=29999

# The USERGROUPS variable can be either "yes" or "no".  If "yes" each
# created user will be given their own group to use as a default.  If
# "no", each created user will be placed in the group whose gid is
# USERS_GID (see below).
USERGROUPS=yes

# If USERGROUPS is "no", then USERS_GID should be the GID of the group
# `users' (or the equivalent group) on your system.
USERS_GID=100

# If DIR_MODE is set, directories will be created with the specified
# mode. Otherwise the default mode 0755 will be used.
DIR_MODE=0755

# If SETGID_HOME is "yes" home directories for users with their own
# group the setgid bit will be set. This was the default for
# versions << 3.13 of adduser. Because it has some bad side effects we
# no longer do this per default. If you want it nevertheless you can
# still set it here.
SETGID_HOME=no

# If QUOTAUSER is set, a default quota will be set from that user with
# `edquota -p QUOTAUSER newuser'
QUOTAUSER=""

--更多--(75%)
```

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

```
zysss@zysss-virtual-machine:~$ cat >facebook.txt
google 110 5000
baidu 100 5000
guge 50 3000
suho 100 4500
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ sort facebook.txt
baidu 100 5000
google 110 5000
guge 50 3000
suho 100 4500
zysss@zysss-virtual-machine:~$
```

（2）按公司人数排序

```
zysss@zysss-virtual-machine:~$ sort -n -t' ' -k 2 facebook.txt
guge 50 3000
baidu 100 5000
suho 100 4500
google 110 5000
zysss@zysss-virtual-machine:~$
```

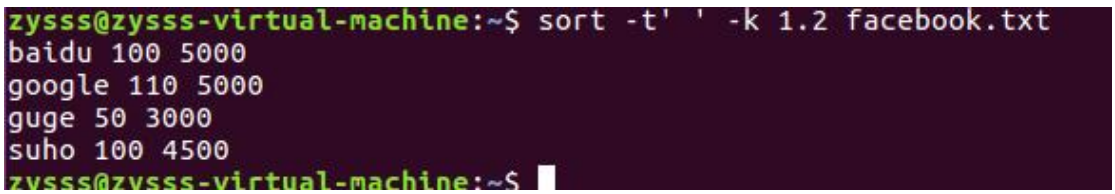
（3）按公司人数排序，人数相同的按照员工平均工资升序排序

```
zysss@zysss-virtual-machine:~$ sort -n -t' ' -k 3 facebook.txt
guge 50 3000
suho 100 4500
baidu 100 5000
google 110 5000
zysss@zysss-virtual-machine:~$
```

（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

```
zysss@zysss-virtual-machine:~$ sort -n -t' ' -k 3r -k 2 facebook.txt
baidu 100 5000
google 110 5000
suho 100 4500
guge 50 3000
zysss@zysss-virtual-machine:~$
```

（5）从公司英文名称的第2个字母开始进行排序。



四、 实验过程分析与讨论

Sort 命令

- 功能说明：将文本文件内容加以排序,sort 可针对文本文件的内容，以行为单位来排序。
- 格式：sort [选项] filename
- -m 将已排序的输入文件，合并为一个排序后的输出数据流。
- -n 以整数类型比较字段
- -o outfile 将输入写到指定文件，而非标准输出。如果该文件为输入文件之一，则 sort 在进行排序写到输入文件之前，会先将它复制到一个临时文件
- -r 倒置排序的顺序为 由大至小（descending）,而非默认的由小至大（ascending）
- -t char 使用单个字符 char 作为默认的字段分割字符，取代默认的空白字符。
- -u 只有唯一的记录，丢弃所有具有相同键值的记录，只留其中的第一条。只有键值字段是重要的，也就是说：被丢弃的记录其他部分可能是不同值。

- 行为模式：sort 会读取指定的文件，如果未给定文件，则读取标准输入，在将排序好的数据写至标准输出。
- -b　忽略开头的空白
- -c　检查输入是否已正确排序，如输入未经排序，但退出码(exit code)为非零值，则不会有任何输出
- -d　字典顺序：仅文字数字与空白才有意义
- -g　一般数值：以浮点数字类型比较字段。这个选项的运作有点类似 -n.差别仅在于这个选项的数字可能有小数点及指数。（仅 GNU 版本提供此功能）
- -f　以不管字母大小写的方式排序
- -i　忽略无法打印的字符

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 3 月 15 日
学　　号	2021223124	姓　　名	张颖
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 下查找文件和统计文件行数、字数和字节数命令：find、locate 、wc

2、掌握 Linux 下文件打包、压缩命令：tar　gzip

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　chown

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 ，并写入任意内容

```
zysss@zysss-virtual-machine:~$ mkdir baz
zysss@zysss-virtual-machine:~$ cd baz
zysss@zysss-virtual-machine:~/baz$ cat >qux <<EOF
> 123
> 456
> abc
> EOF
zysss@zysss-virtual-machine:~/baz$
```

（2）在用户目录下查找文件 qux ，并显示该文件位置信息

```
zysss@zysss-virtual-machine:~$ find ~ -name qux
/home/zysss/baz/qux
zysss@zysss-virtual-machine:~$
```

（3）统计文件 qux 中所包含内容的行数、字数和字节数

```
zysss@zysss-virtual-machine:~/baz$ wc qux
 3  3 12 qux
zysss@zysss-virtual-machine:~/baz$
```

（4）在用户目录下查找文件 qux ，并删除该文件

```
zysss@zysss-virtual-machine:~/baz$ find ~ -name qux -delete
zysss@zysss-virtual-machine:~/baz$ ll
总用量 8
drwxrwxr-x  2 zysss zysss 4096 5月  22 13:03 ./
drwxr-xr-x 19 zysss zysss 4096 5月  22 12:48 ../
zysss@zysss-virtual-machine:~/baz$ ls
zysss@zysss-virtual-machine:~/baz$
```

（5）查看文件夹 baz 内容，看一下是否删除了文件 qux

```
zysss@zysss-virtual-machine:~/baz$ ls
zysss@zysss-virtual-machine:~/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

```
zysss@zysss-virtual-machine:~$ mkdir path1
zysss@zysss-virtual-machine:~$ cd path1/
zysss@zysss-virtual-machine:~/path1$ touch file1 file2
zysss@zysss-virtual-machine:~/path1$ ls
file1  file2
zysss@zysss-virtual-machine:~/path1$
```

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

```
zysss@zysss-virtual-machine:~$ mkdir path2
zysss@zysss-virtual-machine:~$ cd path2
zysss@zysss-virtual-machine:~/path2$ touch file3
zysss@zysss-virtual-machine:~/path2$ ls
file3
zysss@zysss-virtual-machine:~/path2$
```

（3）在用户主目录下新建文件 file4

```
zysss@zysss-virtual-machine:~$ touch file4
zysss@zysss-virtual-machine:~$ ls
ak   d.txt             f1            file4   path1  公共的  视频  文档  音乐
baz  examples.desktop  facebook.txt  locate  path2  模板    图片  下载  桌面
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ tar -cvf package.tar path1 file4
path1/
path1/file2
path1/file1
file4
zysss@zysss-virtual-machine:~$
```

（5）查看包 package.tar 的内容

```
zysss@zysss-virtual-machine:~$ tar -tvf package.tar
drwxrwxr-x zysss/zysss       0 2023-05-22 13:09 path1/
-rw-rw-r-- zysss/zysss       0 2023-05-22 13:09 path1/file2
-rw-rw-r-- zysss/zysss       0 2023-05-22 13:09 path1/file1
-rw-rw-r-- zysss/zysss       0 2023-05-22 13:13 file4
zysss@zysss-virtual-machine:~$
```

（6）向包 package.tar 里添加文件夹 path2 的内容

```
zysss@zysss-virtual-machine:~$ tar -rvf package.tar path2
path2/
path2/file3
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ mkdir path3
zysss@zysss-virtual-machine:~$ cp package.tar path3
zysss@zysss-virtual-machine:~$
```

（8）进入path3 文件夹，并还原包 package.tar 的内容

```
zysss@zysss-virtual-machine:~$ cd path3
zysss@zysss-virtual-machine:~/path3$ tar -xvf package.tar
path1/
path1/file2
path1/file1
file4
path2/
path2/file3
zysss@zysss-virtual-machine:~/path3$
```

3、符号链接内容

（1）新建文件 foo.txt ，内容为 123

```
zysss@zysss-virtual-machine:~$ echo "123" >foo.txt
zysss@zysss-virtual-machine:~$
```

（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；

```
zysss@zysss-virtual-machine:~$ echo "123" >foo.txt
zysss@zysss-virtual-machine:~$ ln foo.txt bar.txt
zysss@zysss-virtual-machine:~$ comm foo.txt bar.txt
                123
zysss@zysss-virtual-machine:~$ diff foo.txt bar.txt
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ ll -i foo.txt bar.txt
1102819 -rw-rw-r-- 2 zysss zysss 4 5月  22 13:23 bar.txt
1102819 -rw-rw-r-- 2 zysss zysss 4 5月  22 13:23 foo.txt
zysss@zysss-virtual-machine:~$
```

（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

```
zysss@zysss-virtual-machine:~$ echo "abc" >bar.txt
zysss@zysss-virtual-machine:~$ diff foo.txt bar.txt
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ rm foo.txt
zysss@zysss-virtual-machine:~$ ll -i bar.txt
1102819 -rw-rw-r-- 1 zysss zysss 4 5月  22 13:27 bar.txt
zysss@zysss-virtual-machine:~$
```

（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

```
zysss@zysss-virtual-machine:~$ ln -s bar.txt baz.txt
zysss@zysss-virtual-machine:~$ ls -i bar.txt baz
1102819 bar.txt

baz:
```

```
zysss@zysss-virtual-machine:~$ df -i bar.txt baz.txt
文件系统         Inode 已用(I) 可用(I) 已用(I)% 挂载点
/dev/sda1      1248480  231117 1017363      19% /
/dev/sda1      1248480  231117 1017363      19% /
zysss@zysss-virtual-machine:~$
```

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

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

4、权限管理

（1）新建文件 qux.txt

```
zysss@zysss-virtual-machine:~$ touch qux.txt
```

（2）增加写权限，创建文件 qux.txt 并为该文件增加执行权限（所有用户都可以执行）

```
zysss@zysss-virtual-machine:~$ chmod +x qux.txt
zysss@zysss-virtual-machine:~$ ll
总用量 160
drwxr-xr-x 22 zysss zysss  4096 5月  22 13:41 ./
drwxr-xr-x  3 root  root   4096 4月  14 22:42 ../
-rw-rw-r--  1 zysss zysss    13 5月  22 11:07 ak
-rw-------  1 zysss zysss  1331 5月  22 13:23 .bash_history
-rw-r--r--  1 zysss zysss   220 4月  14 22:42 .bash_logout
-rw-r--r--  1 zysss zysss  3771 4月  14 22:42 .bashrc
drwxrwxr-x  2 zysss zysss  4096 5月  22 13:03 baz/
lrwxrwxrwx  1 zysss zysss     7 5月  22 13:30 baz.txt -> bar.txt
drwx------ 12 zysss zysss  4096 5月  22 10:30 .cache/
drwx------ 19 zysss zysss  4096 4月  14 22:58 .config/
drwx------  3 zysss zysss  4096 4月  14 22:53 .dbus/
-rw-rw-r--  1 zysss zysss    65 5月  22 11:31 d.txt
-rw-r--r--  1 zysss zysss  8980 4月  14 22:42 examples.desktop
-rw-rw-r--  1 zysss zysss     5 5月  22 12:28 f1
-rw-rw-r--  1 zysss zysss    58 5月  22 11:34 facebook.txt
-rw-rw-r--  1 zysss zysss     0 5月  22 13:13 file4
drwx------  2 zysss zysss  4096 4月  15 18:28 .gconf/
drwx------  3 zysss zysss  4096 5月  22 09:52 .gnupg/
-rw-------  1 zysss zysss   756 5月  22 10:30 .ICEauthority
drwx------  3 zysss zysss  4096 4月  14 22:53 .local/
drwxrwxr-x  2 zysss zysss  4096 5月  22 12:17 locate/
-rw-rw-r--  1 zysss zysss 10240 5月  22 13:17 package.tar
drwxrwxr-x  2 zysss zysss  4096 5月  22 13:09 path1/
drwxrwxr-x  2 zysss zysss  4096 5月  22 13:12 path2/
drwxrwxr-x  4 zysss zysss  4096 5月  22 13:20 path3/
drwx------  2 zysss zysss  4096 4月  14 22:53 .presage/
-rw-r--r--  1 zysss zysss   655 4月  14 22:42 .profile
-rwxrwxr-x  1 zysss zysss     0 5月  22 13:41 qux.txt*
-rw-r--r--  1 zysss zysss     0 4月  14 22:59 .sudo_as_admin_successf
```

四、实验过程分析与讨论

本次实验进行了 Linux 系统中的查找、压缩、链接文件和修改权限命令的练习。

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 vi 编辑器及 gcc 编译器的使用		
实验教室	丹青 922	实验日期	2023 年 3 月 22 日
学　　号	2021223124	姓　　名	张颖
专业班级	计算机科学与技术 5 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 vi 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

```
zysss@zysss-virtual-machine:~$ mkdir workspace1
zysss@zysss-virtual-machine:~$ cd workspace1
zysss@zysss-virtual-machine:~/workspace1$
```

（2）进入目录 workspace1

```
zysss@zysss-virtual-machine:~$ mkdir workspace1
zysss@zysss-virtual-machine:~$ cd workspace1
zysss@zysss-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 int main( ) { printf(“helloworl d!\n”); return 0; }

```
#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

```
zysss@zysss-virtual-machine:~/workspace1$ gcc test.c
zysss@zysss-virtual-machine:~/workspace1$ ./a.out
hello world!
zysss@zysss-virtual-machine:~/workspace1$
```

2. vim 编辑器的详细使用

（1）在用户目录下创建一个名为 workspace2 的目录

```
zysss@zysss-virtual-machine:~/workspace1$ mkdir workspace2
zysss@zysss-virtual-machine:~/workspace1$
```

（2）进入workspace2 目录

```
zysss@zysss-virtual-machine:~/workspace1$ mkdir workspace2
zysss@zysss-virtual-machine:~/workspace1$
```

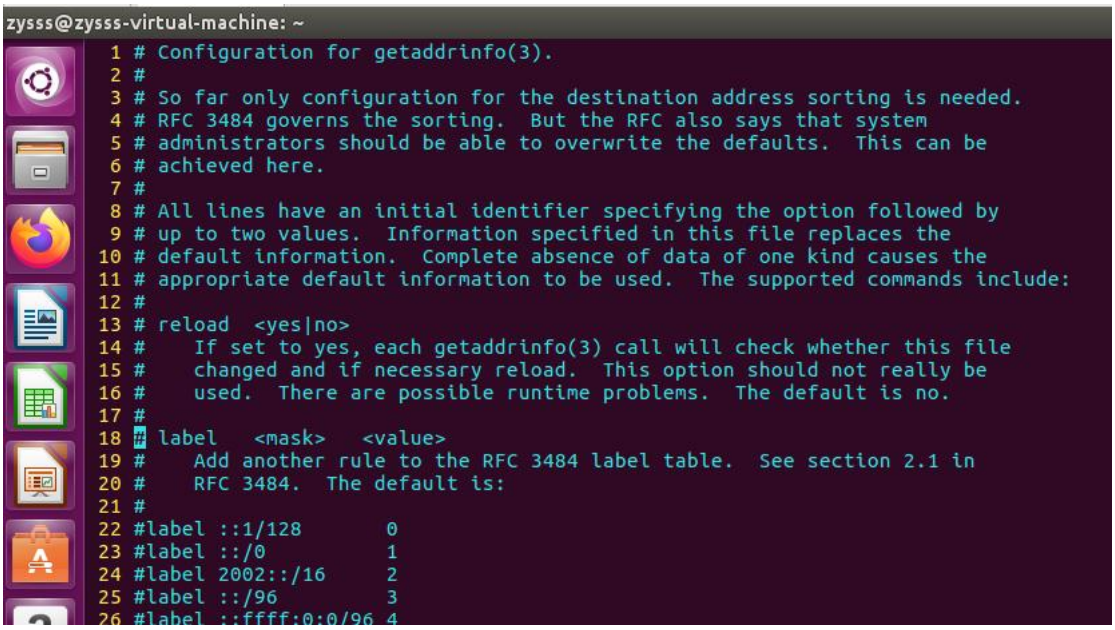
（3）使用以下命令：将文件/etc/gai.conf 的内容复制到当前目录下的新建文件 gai.conf 中

```
zysss@zysss-virtual-machine:~/workspace1$ cat /etc/gai.conf > ./gai.conf
zysss@zysss-virtual-machine:~/workspace1$
```

（4）使用vim 编辑当前目录下的 gai.conf

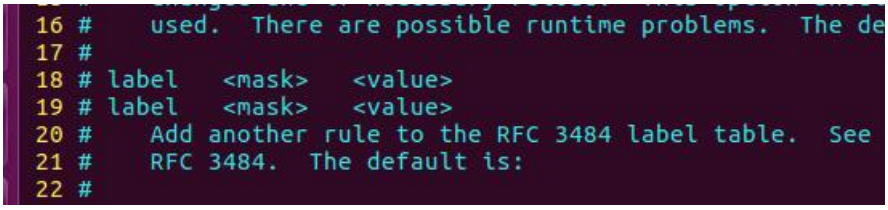
```
zysss@zysss-virtual-machine:~/workspace1$ vim gai.conf
```

（5）将光标移到第 18 行

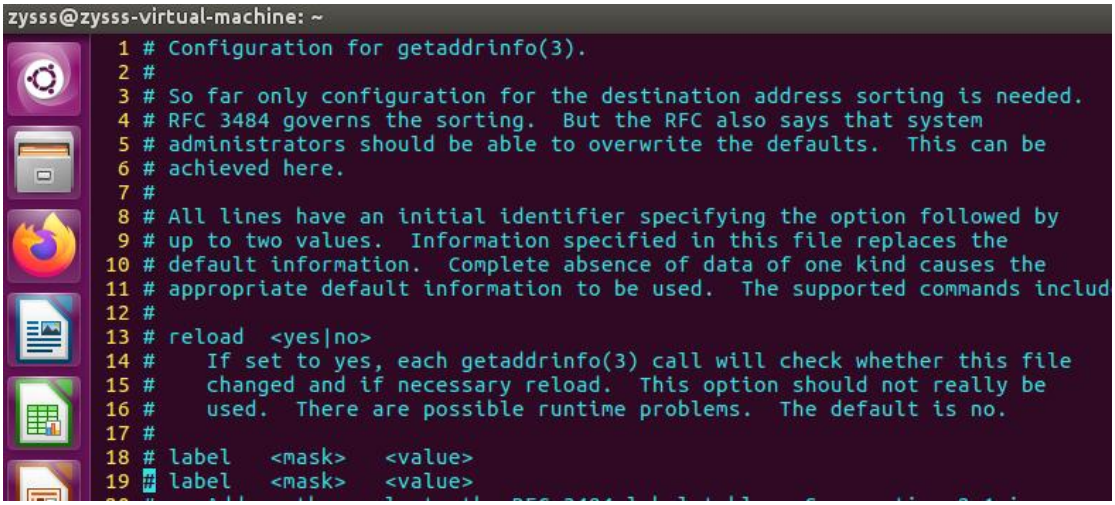


（6）复制该行内容

yy p

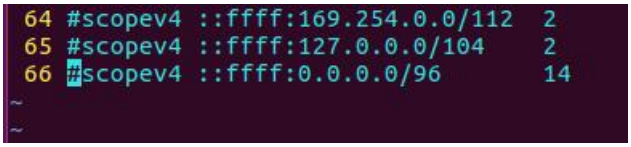


（8）将光标移到最后一行行首



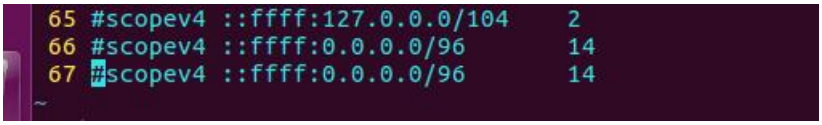
（9）将光标移到最后一行行首

G

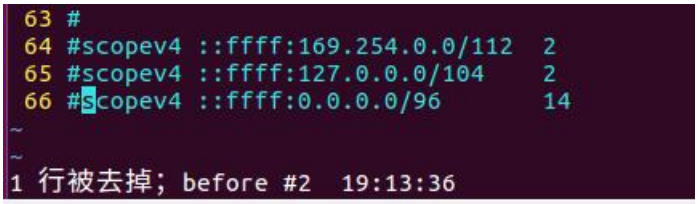


（10）粘贴复制行的内容

yy p



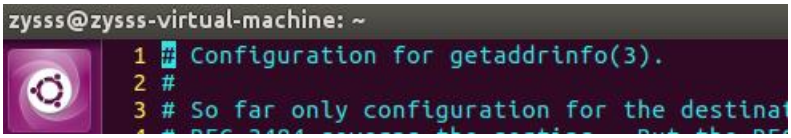
（9）撤销第 8 步的动作



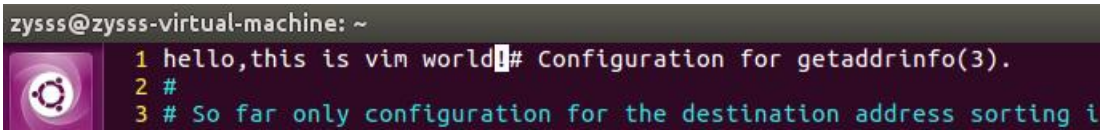
（10）存盘但不退出

（11）将光标移到首行

gg



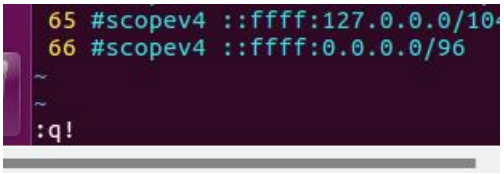
（12）插入模式下输入"Hello, this is vim world!"



（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

在运行 vim gai.conf 遇到了“vim 已包含在下列软件中”，输入 sudo apt-get install exuberant-ctags 解决了问题。

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	2022 年 3 月 29 日
学　　号	2021223124	姓　　名	张颖
专业班级	计算机科学与技术 5 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

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

```
zysss@zysss-virtual-machine:~$ sudo useradd -m -s /bin/sh -p $(openssl passwd -1 123456) -c "bar" foo
[sudo] zysss 的密码:
zysss@zysss-virtual-machine:~$
```

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@zysss-virtual-machine:/home/zysss# usermod -u 2000 -s /bin/csh foo
root@zysss-virtual-machine:/home/zysss# su -foo
```

3.从用户 foo 切换到 root

```
root@zysss-virtual-machine:/home/zysss# su - root
root@zysss-virtual-machine:~#
```

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@zysss-virtual-machine:~# 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@zysss-virtual-machine:~# vim user.txt
root@zysss-virtual-machine:~# vim pass.txt
root@zysss-virtual-machine:~#
root@zysss-virtual-machine:~# newusers user.txt
root@zysss-virtual-machine:~# tail /etc/passwd -n 3
user001:x:2001:2002:user001:/home/user001:/bin/bash
user002:x:2002:2002:user002:/home/user002:/bin/bash
user003:x:2003:2002:user003:/home/user003:/bin/bash
root@zysss-virtual-machine:~# pwunconv
```

```
root@zysss-virtual-machine:~# chpasswd < passwds.txt
root@zysss-virtual-machine:~# pwconv
root@zysss-virtual-machine:~#
```

```
root@zysss-virtual-machine:~# sudo chpasswd < user.txt
root@zysss-virtual-machine:~# ll /home
总用量 24
drwxr-xr-x  6 root    root    4096 5月  22 21:14 ./
drwxr-xr-x 24 root    root    4096 4月  15 18:33 ../
drwxr-xr-x  2 user001 user001 4096 5月  22 21:14 user001/
drwxr-xr-x  2 user002 user001 4096 5月  22 21:14 user002/
drwxr-xr-x  2 user003 user001 4096 5月  22 21:14 user003/
drwxr-xr-x 23 zysss   zysss   4096 5月  22 21:29 zysss/
root@zysss-virtual-machine:~#
```

6.创建用户组 group1，并在创建时设置其 GID 为 3000.

```
root@zysss-virtual-machine:~# sudo groupadd -g 3000 group1
root@zysss-virtual-machine:~# cat /etc/group | grep group1
group1:x:3000:
root@zysss-virtual-machine:~#
```

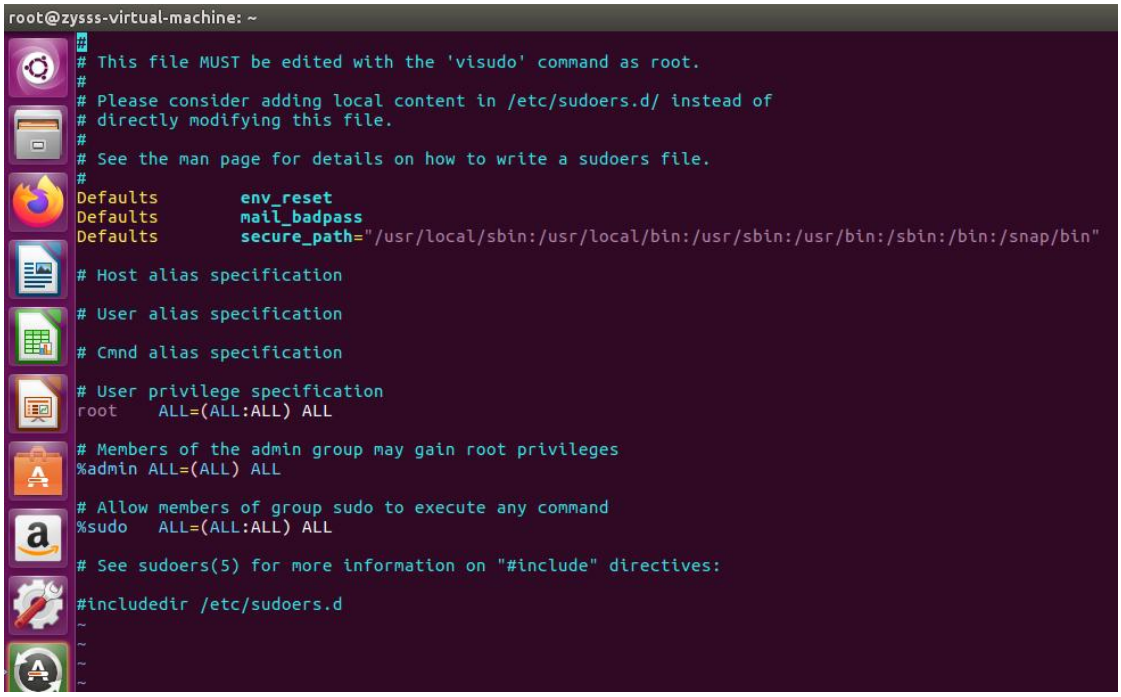
7.在用户组 group1 中添加两个之前批量创建的用户

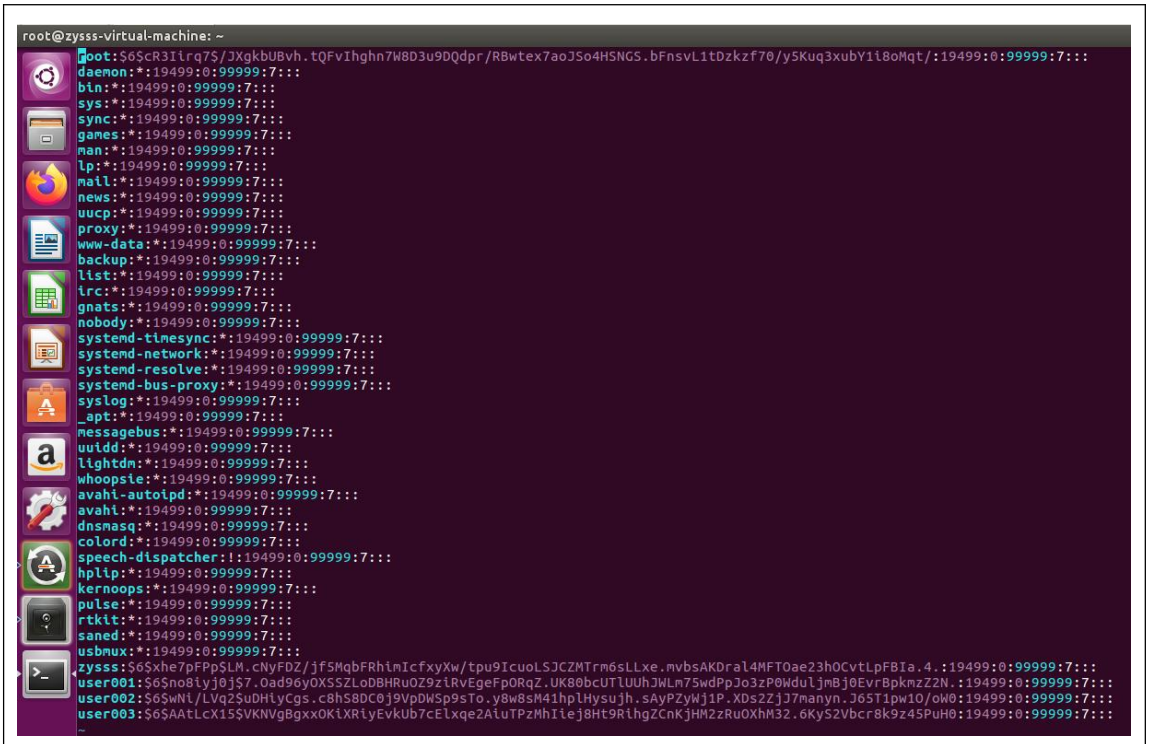
```
root@zysss-virtual-machine:~# sudo usermod -a -G group1 user001
root@zysss-virtual-machine:~# sudo usermod -a -G group1 user002
root@zysss-virtual-machine:~#
```

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@zysss-virtual-machine:~# sudo -u user001 -i
user001@zysss-virtual-machine:~$ sudo /etc/shadow
[sudo] user001 的密码:
user001 不在 sudoers 文件中。此事将被报告。
user001@zysss-virtual-machine:~$ exit
注销
root@zysss-virtual-machine:~# vi /etc/sudoers
root@zysss-virtual-machine:~# sudo vi /etc/shadow
root@zysss-virtual-machine:~#
```





四、实验过程分析与讨论

创建用户组时遇到了一些问题，通过问同学和 CSDN 成功解决问题。

五、指导教师意见

指导教师签字： 卢洋

实验报告

实验名称	实验五 Shell 程序的创建及条件判断语句		
实验教室	丹青 922	实验日期	2023 年 4 月 5 日
学　　号	2021223124	姓　　名	张颖
专业班级	计算机科学与技术 5 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）

```
zysss@zysss-virtual-machine:~$ foo=200
zysss@zysss-virtual-machine:~$ echo $foo
200
zysss@zysss-virtual-machine:~$
```

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）

```
zysss@zysss-virtual-machine:~$ bar=100
zysss@zysss-virtual-machine:~$ test $bar -gt 150
zysss@zysss-virtual-machine:~$ echo $?
1
zysss@zysss-virtual-machine:~$
```

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 ）

```
zysss@zysss-virtual-machine:~$ vi test.sh
```

```
zysss@zysss-virtual-machine:~$ sh test.sh
Hostname: zysss-virtual-machine
System Time: 2023年 05月 22日 星期一 23:42:12 CST
zysss@zysss-virtual-machine:~$
```

```
hostname=$(hostname)
current_date=$(date)
echo "Hostname: $hostname"
echo "System Time: $current_date"
~
```

4、创建一个简单的 Shell 程序，要求带一个参数，判断该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=13+33+53，153 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

```
zysss@zysss-virtual-machine:~$ vim myscript.sh
```

```
#!/bin/bash

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

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

```
zysss@zysss-virtual-machine:~$ bash myscript.sh 153
153 is shuixianhua num!
zysss@zysss-virtual-machine:~$ bash myscript.sh 124
124 is not a shuixianhua num!
zysss@zysss-virtual-machine:~$ bash myscript.sh 370
370 is shuixianhua num!
zysss@zysss-virtual-machine:~$
```

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

```
zysss@zysss-virtual-machine:~$ vim sum.sh
zysss@zysss-virtual-machine:~$ chmod +s sum.sh
zysss@zysss-virtual-machine:~$ ./sum.sh 1 2 3
The sum of the three numbers is: 6
zysss@zysss-virtual-machine:~$
```

```
#!/bin/bash

num1=$1
num2=$2
num3=$3

sum=$((num1+num2+num3))
echo "The sum of the three numbers is: $sum"
~
```

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，90 分以上为 A，80-90 为 B，70-80 为 C，60-70 为 D，小于 60 分为 E。要求使用 if…elif….else　fi 实现

```
zysss@zysss-virtual-machine:~$ vim grade.sh
zysss@zysss-virtual-machine:~$ sh grade.sh 91 12 60
A
E
D
zysss@zysss-virtual-machine:~$
```

```
#!/bin/bash

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

四、实验过程分析与讨论

shell 中的逻辑判断一般用 if 语句，if 语句中通常用[]来表示条件测试，可以比较字符串、判断文件是否存等。备注：[ ] 中表达式两边与括号之间要有空格

if … else 语句常用基本的语法如下：

1.if [];then elif []; then else fi 语句，哪个 expression 表达式成立则执行哪个 then 后面的语句，否则执行 else 后面的语句。

2.if … else 语句也经常与 test 命令结合使用,test 命令用于检查某个条件是否成立，与方括号[ ]功能类似

3.if 语句常用命令选项有：

== or =： 等于

-eq ： 等于

-ne ：不等于

-gt ：大于

-ge ：大于等于

-lt ：小于

-le ：小于等于

五、指导教师意见

指导教师签字： 卢洋

实验报告

实验名称	实验六 Shell 循环控制语句		
实验教室	丹青 922	实验日期	2023 年 4 月 12 日
学　　号	2021223124	姓　　名	张颖

专业班级	计算机科学与技术 5 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)

```
zysss@zysss-virtual-machine:~$ vim copy.sh
zysss@zysss-virtual-machine:~$ sh copy.sh
successful copied
```

```
#!/bin/bash
target_dir=~/workspace/
for file in *.c;do
     cp "$file" "$target_dir"
done
echo "successful copied"
~
```

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

```
zysss@zysss-virtual-machine:~$ vim oushu.sh
zysss@zysss-virtual-machine:~$ sh oushu.sh
sum is:90
zysss@zysss-virtual-machine:~$
```

```
i=0
sum=0
while [ "$i" -lt 20 ]
do
if [ $(($i%2)) -eq 0 ];then
     sum=$(($sum+$i))
fi
i=$(($i+1))
done
echo "sum is:$sum"
```

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

```
zysss@zysss-virtual-machine:~$ vim pingfang.sh
zysss@zysss-virtual-machine:~$ sh pingfang.sh
385
zysss@zysss-virtual-machine:~$
```

```
i=1
sum=0
until [ "$i" -gt 10 ]
do
sum=$(($sum+$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、break2、continue、continue2，并观察四种情况下的实验结果

```
#!/bin/sh
for i in a b c d
 do
echo -n $i
   for j in 1 2 3 4 5 6 7 8 9 10
   do
    if [ $j -eq 5 ];then
      ---
    fi
    echo –n “ $j”
   done
 echo $j
done
```

（1）break：

```
root@zysss-virtual-machine:/home/zysss# vim myscript6.sh
root@zysss-virtual-machine:/home/zysss# chmod u+x myscript6.sh
root@zysss-virtual-machine:/home/zysss# bash myscript6.sh
a1234
b1234
c1234
d1234
root@zysss-virtual-machine:/home/zysss#
```

```
#!/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
~
```

（2）Break 2：

```
root@zysss-virtual-machine:/home/zysss# vim myscript6.sh
root@zysss-virtual-machine:/home/zysss# chmod u+x myscript6.sh
root@zysss-virtual-machine:/home/zysss# bash myscript6.sh
a1234root@zysss-virtual-machine:/home/zysss#
```

```
#!/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
~
```

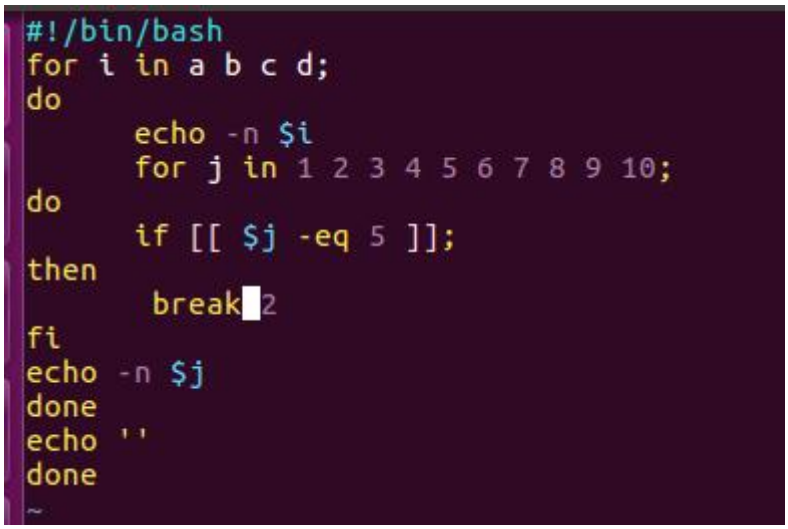
（3）Continue：

```
root@zysss-virtual-machine:/home/zysss# vim myscript6.sh
root@zysss-virtual-machine:/home/zysss# chmod u+x myscript6.sh
root@zysss-virtual-machine:/home/zysss# bash myscript6.sh
a1234678910
b1234678910
c1234678910
d1234678910
root@zysss-virtual-machine:/home/zysss#
```

```
#!/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
~
```

（4）Continue 2：

```
root@zysss-virtual-machine:/home/zysss# vim myscript6.sh
root@zysss-virtual-machine:/home/zysss# chmod u+x myscript6.sh
root@zysss-virtual-machine:/home/zysss# bash myscript6.sh
a1234b1234c1234d1234root@zysss-virtual-machine:/home/zysss#
```



```
#!/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
~
```

四、实验过程分析与讨论

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

（2）列表 for 循环

do 和 done 之间的命令称为循环体，执行次数和 list 列表中常数或字符串的个数相同。for 循环，首先将 in 后 list 列表的第一个常数或字符串赋值给循环变量，然后执行循环体，以此执行 list，最后执行 done 命令后的命令序列。

Sheel 支持列表 for 循环使用略写的计数方式，1～5 的范围用{1…5}表示（大括号不能去掉，否则会当作一个字符串处理）。

Sheel 中还支持按规定的步数进行跳跃的方式实现列表 for 循环，例如计算 1～100 内所有的奇数之和。

$ #表示参数的个数， @ 表 示 参 数 列 表 而 @表示参数列表而

@表示参数列表而*则把所有的参数当作一个字符串显示。 （3）不带列表 for 循环
五、指导教师意见 指导教师签字： 卢洋

实验报告

实验名称	实验七 Shell 函数

实验教室	丹青 922	实验日期	2023 年 4 月 19 日
学　　号	2021223124	姓　　名	张颖
专业班级	计算机科学与技术 5 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

```
root@zysss-virtual-machine:/home/zysss# vim myscript7.sh
root@zysss-virtual-machine:/home/zysss# bash myscript7.sh
the sum of 3 and 4 is 7
root@zysss-virtual-machine:/home/zysss#
```

```
function addtion(){
    sum=$(($1 + $2))
    echo "the sum of $1 and $2 is $sum"
}

addtion 3 4
~
```

(2) 编写 shell 脚本，该脚本中定义一个递归函数，求 n 的阶乘

```
root@zysss-virtual-machine:/home/zysss# vim myscript8.sh
root@zysss-virtual-machine:/home/zysss# bash myscript8.sh 5
120
root@zysss-virtual-machine:/home/zysss#
```

```
#!/bin/bash

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

(3)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 "start..."
first
```

```
root@zysss-virtual-machine:/home/zysss# vim myscript5.sh
root@zysss-virtual-machine:/home/zysss# bash myscript5.sh
start...
this is the first
this is the second
-----this is third
root@zysss-virtual-machine:/home/zysss#
```

```
#!/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 "start..."
first
~
```

四、实验过程分析与讨论

函数调用的相关知识。

Shell 函数定义的语法格式如下：

function name() {

    statements

    [return value]

}

对各个部分的说明：

function 是 Shell 中的关键字，专门用来定义函数；

name 是函数名；

statements 是函数要执行的代码，也就是一组语句；

return value 表示函数的返回值，其中 return 是 Shell 关键字，专门用在函数中返回一个值；这一部分可以写也可以不写。

由{ }包围的部分称为函数体，调用一个函数，实际上就是执行函数体中的代码。

函数定义的简化写法

如果你嫌麻烦，函数定义时也可以不写 function 关键字：

```
name() {
    statements
    [return value]
}
```

如果写了 function 关键字，也可以省略函数名后面的小括号：

```
function name {
    statements
    [return value]
}
```

函数调用

调用 Shell 函数时可以给它传递参数，也可以不传递。如果不传递参数，直接给出函数名字即可：

```
name
```

如果传递参数，那么多个参数之间以空格分隔：

```
name param1 param2 param3
```

不管是哪种形式，函数名字后面都不需要带括号。

和其它编程语言不同的是，Shell 函数在定义时不能指明参数，但是在调用时却可以传递参数，并且给它传递什么参数它就接收什么参数。

Shell 也不限制定义和调用的顺序，你可以将定义放在调用的前面，也可以反过来，将定义放在调用的后面。

五、指导教师意见

指导教师签字： 卢洋

实验报告

实验名称	实验八 sed 和 awk		
实验教室	丹青 922	实验日期	2023 年 5 月 21 日
学　　号	2021223124	姓　　名	张颖
专业班级	计算机科学与技术 5 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）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、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

```
root@zysss-virtual-machine:/home/zysss# vim quote.txt
root@zysss-virtual-machine:/home/zysss# 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.
root@zysss-virtual-machine:/home/zysss#
```

（2）显示包含 music 文字的行内容及行号

```
root@zysss-virtual-machine:/home/zysss# cat quote.txt |sed -n '/music/p'
It was an evening of splendid music and company.
root@zysss-virtual-machine:/home/zysss#
```

（3）在第 4 行后面追加文件“hello world！”

```
root@zysss-virtual-machine:/home/zysss# 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!
root@zysss-virtual-machine:/home/zysss#
```

（4）将文本“The”修改为“Quod”

```
root@zysss-virtual-machine:/home/zysss# 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.
root@zysss-virtual-machine:/home/zysss#
```

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

```
root@zysss-virtual-machine:/home/zysss# 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.
root@zysss-virtual-machine:/home/zysss#
```

（6）删除第 2 行内容

```
root@zysss-virtual-machine:/home/zysss# 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.
root@zysss-virtual-machine:/home/zysss#
```

（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@zysss-virtual-machine:/home/zysss# var=evening
root@zysss-virtual-machine:/home/zysss# cat quote.txt |sed -n "/$var/p"
It was an evening of splendid music and company.
root@zysss-virtual-machine:/home/zysss#
```

3. 文件 number.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 行的内容，观察两者的区别

```
root@zysss-virtual-machine:/home/zysss# cat <<EOF >number.txt
> one : two : three
> four : five : six
> EOF
root@zysss-virtual-machine:/home/zysss# cat number.txt | awk '{FS=" "}{print $2}'
:
:
root@zysss-virtual-machine:/home/zysss#
```

如果以一个空格作为分隔符，则冒号会被视为单独的一列

如果以一个冒号作为分隔符，则则会将字段分为 5 组，且第一组的冒号: 会被保留，且对角线上的元素会被分为一列

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。

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：

2

4

46

```
root@zysss-virtual-machine:/home/zysss# cat <<EOF >foo.txt
> 2 4 3
> 15 46 79
> EOF
root@zysss-virtual-machine:/home/zysss# awk '{for(i=1;i<=NF;i++) if($i%2==0) {print $i; count++}} END{print "The total number of even numbers is: "count}' foo.txt
2
4
46
The total number of even numbers is: 3
root@zysss-virtual-machine:/home/zysss#
```

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。

#!/bin/bash

read –p “enter search pattern: “ pattern

awk “/$pattern/”‘{ nmatches++; print } END { print nmatches “found.” }’ info.txt

```
root@zysss-virtual-machine:/home/zysss# vim scripts.sh
root@zysss-virtual-machine:/home/zysss# cat scripts.sh
#!/bin/bash
read -p "enter search pattern: " pattern
awk "/$pattern/" '{nmatches++; print } END { print nmatches, "found." }' info.txt
root@zysss-virtual-machine:/home/zysss# cat info.txt
1 2 3
123
abc
root@zysss-virtual-machine:/home/zysss# sh scripts.sh
enter search pattern: abc
```

awk 中"/$pattern/"这一部分用双引号括起来，是为了允许引号内的 Shell 变量进行替换

此脚本的作用用于匹配字符串

首先输入你要匹配的字符串，脚本中指定的文件为 info.txt

并在 info.txt 文件中查找相应的字符串，如果能匹配到，则 nmatches 变量就加一，并在最后输出要匹配字符串出现的位置，以及出现的次数

四、实验过程分析与讨论

sed 和 awk 的用法：

sed 命令的作用是利用脚本来处理文本文件。使用方法：

sed [参数] [n1][n2]function

n1,n2 不一定存在，一般表示进行动作的行。如果动作在 10-20 行进行，则为 10,20[function]

参数说明：

-e 或--expression= 以选项中指定的 script 来处理输入的文本文件，这个-e 可以省略，直接写表达式。

-f 或--file=以选项中指定的 script 文件来处理输入的文本文件。

-h 或--help 显示帮助。

-n 或 --quiet 或 --silent 仅显示 script 处理后的结果。

-V 或 --version 显示版本信息。

-i 直接在源文件里修改内容

动作说明[function]：

a：追加， a 的后面可以接字串，而这些字串会在目标行末尾追加～

c：取代， c 的后面可以接字串，这些字串可以取代 n1,n2 之间的行！

d：删除，因为是删除啊，所以 d 后面通常不接任何咚咚；

i：插入， i 的后面可以接字串，而这些字串会在新的一行出现(目前的上一行)；

p：打印，亦即将某个选择的数据印出。通常 p 会与参数 sed -n 一起运行～

s：取代，通常这个 s 的动作可以搭配正规表示法，例如 1,20s/old/new/g

五、指导教师意见

指导教师签字： 卢洋