实验报告

实验名称	实验一 Linux 常用命令（一）		
实验教室	丹青 918	实验日期	2023 年 3 月 8 日
学　　号	2021213113	姓　　名	张学浩
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

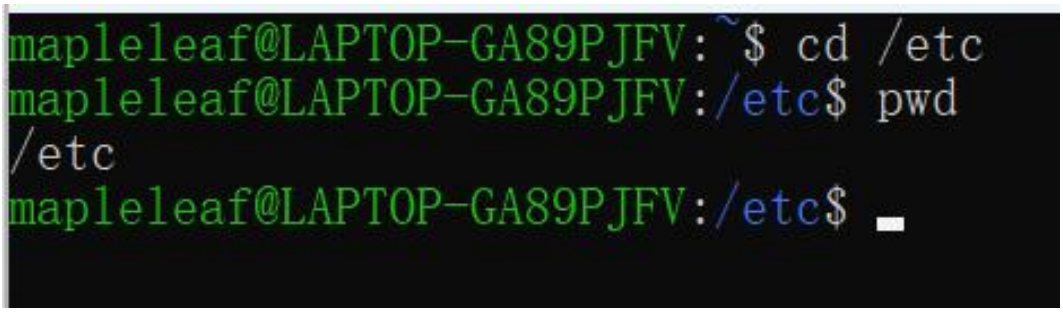
二、实验环境

（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/{用户名} 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/{用户名}/linux ， 然后删除该目录

```
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir linux
mapleleaf@LAPTOP-GA89PJFV:~$ ls
linux
mapleleaf@LAPTOP-GA89PJFV:~$ rmdir linux
mapleleaf@LAPTOP-GA89PJFV:~$ ls
mapleleaf@LAPTOP-GA89PJFV:~$
```

4.使用命令cat用输出重定向在 /home/{用户名} 目录下创建文件 foo ， 文件内容为 "Hello, Linux!" ， 并查 看该文件的内容

```
mapleleaf@LAPTOP-GA89PJFV:~$ cat >foo
Hello linux!
mapleleaf@LAPTOP-GA89PJFV:~$ cat foo
Hello linux!
mapleleaf@LAPTOP-GA89PJFV:~$
```

5. 使用命令创建目录 /home/{用户名}/foo.bak ，然后将 /home/{用户名}/foo 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

```
mapleleaf@LAPTOP-GA89PJFV:~$ ls
foo
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir bak
mapleleaf@LAPTOP-GA89PJFV:~$ ls
bak  foo
mapleleaf@LAPTOP-GA89PJFV:~$ cp -r foo bak
mapleleaf@LAPTOP-GA89PJFV:~$ ls
bak  foo
mapleleaf@LAPTOP-GA89PJFV:~$ cd bak
mapleleaf@LAPTOP-GA89PJFV:~/bak$ rm -i foo
rm: remove regular file 'foo'? y
mapleleaf@LAPTOP-GA89PJFV:~/bak$ cd ..
mapleleaf@LAPTOP-GA89PJFV:~$ rmdir bak
mapleleaf@LAPTOP-GA89PJFV:~$ ls
foo
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:/etc$ head -n 3 adduser.conf
# /etc/adduser.conf: `adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.

mapleleaf@LAPTOP-GA89PJFV:/etc$ tail -n 5 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
mapleleaf@LAPTOP-GA89PJFV:/etc$
```

7. 分屏查看文件 /etc/adduser.conf 的内容

```
# /etc/adduser.conf: `adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.

# The DSHELL variable specifies the default login shell on your
# system.
DSHELL=/bin/bash

# The DHOME variable specifies the directory containing users' home
# directories.
DHOME=/home

# If GROUPHOMES is "yes", then the home directories will be created as
# /home/groupname/user.
GROUPHOMES=no

# If LETTERHOMES is "yes", then the created home directories will have
# an extra directory - the first letter of the user name. For example:
# /home/u/user.
LETTERHOMES=no

# The SKEL variable specifies the directory containing "skeletal" user
# files; in other words, files such as a sample .profile that will be
# copied to the new user's home directory when it is created.
SKEL=/etc/skel

# FIRST_SYSTEM_[GU]ID to LAST_SYSTEM_[GU]ID inclusive is the range for UIDs
# 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.
--More--(36%)
```

8. 使用命令 cat 用输出重定向在 /home/{用户名} 目录下创建文件 bar.txt ，文件内容为

```
google  110 5000
baidu   100 5000
guge    50  3000
sohu    100 4500
```

> 其中：第1列为公司名称，第2列为公司人数，第3列为员工平均工资。

```
mapleleaf@LAPTOP-GA89PJFV:~$ cat >bar.txt
google  110  5000
baidu   100  5000
guge    50   3000
sohu    100  4500
mapleleaf@LAPTOP-GA89PJFV:~$ ls
bar.txt
mapleleaf@LAPTOP-GA89PJFV:~$ cat bar.txt
google  110  5000
baidu   100  5000
guge    50   3000
sohu    100  4500
mapleleaf@LAPTOP-GA89PJFV:~$
```

使用 sort 命令完成下列排序：

（1）按公司字母顺序排序；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sort bar.txt
baidu   100  5000
google  110  5000
guge    50   3000
sohu    100  4500
mapleleaf@LAPTOP-GA89PJFV:~$
```

（2）按公司人数排序；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sort -n -t ' ' -k 2 bar.txt
guge    50   3000
baidu   100  5000
sohu    100  4500
google  110  5000
mapleleaf@LAPTOP-GA89PJFV:~$
```

（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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sort -n -t ' ' -k 2 -k 3 bar.txt
guge    50   3000
baidu   100  5000
sohu    100  4500
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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sort -n -t ' ' -k 3r -k 2 bar.txt
google  110  5000
baidu   100  5000
guge    50   3000
sohu    100  4500
```

（5）从公司英文名称的第2个字母开始进行排序

```
mapleleaf@LAPTOP-GA89PJFV:~$ 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

在本次实验过程中不太熟悉命令，会在一些比较简单的地方犯错。比如用 cat 写完内容后不知道按 ctrl+d 方可完成编辑，用 rm -i 交互式删除文件需要按 y 才能成功删除，最后的排序也是比较难懂，但是弄清楚原理后就很简单。

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 918	实验日期	2023 年 3 月 16 日
学　　号	2021213113	姓　　名	张学浩
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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir baz
mapleleaf@LAPTOP-GA89PJFV:~$ ls
baz
mapleleaf@LAPTOP-GA89PJFV:~$ cd baz
mapleleaf@LAPTOP-GA89PJFV:~/baz$ vim qux
mapleleaf@LAPTOP-GA89PJFV:~/baz$ ls
qux
mapleleaf@LAPTOP-GA89PJFV:~/baz$ cat qux
hello world!
today
is
a
good
day
```

（2）在用户目录下查找文件 qux ，并显示该文件位置信息；

```
mapleleaf@LAPTOP-GA89PJFV:~$ find -name qux
./baz/qux
mapleleaf@LAPTOP-GA89PJFV:~$ cd baz
mapleleaf@LAPTOP-GA89PJFV:~/baz$ ls -i -F qux
2404 qux
```

（3）统计文件 qux 中所包含内容的行数、字数和字节数；

```
mapleleaf@LAPTOP-GA89PJFV:~$ cd baz
mapleleaf@LAPTOP-GA89PJFV:~/baz$ wc -l qux
7 qux
mapleleaf@LAPTOP-GA89PJFV:~/baz$ wc -m qux
34 qux
mapleleaf@LAPTOP-GA89PJFV:~/baz$ wc -c qux
34 qux
mapleleaf@LAPTOP-GA89PJFV:~/baz$ wc qux
 7  7 34 qux
```

（4）在用户目录下查找文件 qux ，并删除该文件；

```
mapleleaf@LAPTOP-GA89PJFV:~$ find -name qux -exec rm -rf {} \;
mapleleaf@LAPTOP-GA89PJFV:~$ cd baz
mapleleaf@LAPTOP-GA89PJFV:~/baz$ ls
mapleleaf@LAPTOP-GA89PJFV:~/baz$
```

（5）查看文件夹 baz 内容，看一下是否删除了文件 qux 。

```
mapleleaf@LAPTOP-GA89PJFV:~$ cd baz
mapleleaf@LAPTOP-GA89PJFV:~/baz$ ls
mapleleaf@LAPTOP-GA89PJFV:~/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 ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir path1
mapleleaf@LAPTOP-GA89PJFV:~$ cd path1
mapleleaf@LAPTOP-GA89PJFV:~/path1$ touch file1
mapleleaf@LAPTOP-GA89PJFV:~/path1$ touch file2
mapleleaf@LAPTOP-GA89PJFV:~/path1$ ls
file1  file2
mapleleaf@LAPTOP-GA89PJFV:~/path1$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir path2
mapleleaf@LAPTOP-GA89PJFV:~$ cd path2
mapleleaf@LAPTOP-GA89PJFV:~/path2$ touch file3
mapleleaf@LAPTOP-GA89PJFV:~/path2$ ls
file3
```

（3）在用户目录下新建文件 file4 ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ touch file4
mapleleaf@LAPTOP-GA89PJFV:~$ ls
file4  path1  path2
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ touch file4
mapleleaf@LAPTOP-GA89PJFV:~$ ls
file4  path1  path2
mapleleaf@LAPTOP-GA89PJFV:~$ tar -cvf package.tar file4 path1
file4
path1/
path1/file2
path1/file1
mapleleaf@LAPTOP-GA89PJFV:~$
```

（5）查看包 package.tar 的内容；

```
mapleleaf@LAPTOP-GA89PJFV:~$ cat package.tar
file4000064400017500001750000000000001443267416101244O ustar
mapleleafmapleleafpath1/000075500017500001750000000000001443267
4004012535 5ustar  mapleleafmapleleafpath1/file2000064400017500
00175000000000001443267400401344 7 0ustar  mapleleafmapleleafpa
th1/file1000064400017500001750000000000001443267377401346 3 Oust
ar  mapleleafmapleleafmapleleaf@LAPTOP-GA89PJFV:~$
```

（6）向包 package.tar 里添加文件夹 path2 的内容；

```
mapleleaf@LAPTOP-GA89PJFV:~$ tar -rf package.tar path2
mapleleaf@LAPTOP-GA89PJFV:~$ tar -tf package.tar
file4
path1/
path1/file2
path1/file1
path2/
path2/file3
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir path3
mapleleaf@LAPTOP-GA89PJFV:~$ cp package.tar path3
mapleleaf@LAPTOP-GA89PJFV:~$ cd path3
mapleleaf@LAPTOP-GA89PJFV:~/path3$ ls
package.tar
mapleleaf@LAPTOP-GA89PJFV:~/path3$
```

（8）进入 path3 文件夹，并还原包 package.tar 的内容。

```
mapleleaf@LAPTOP-GA89PJFV:~/path3$ tar -xvf package.tar
file4
path1/
path1/file2
path1/file1
path2/
path2/file3
path2/
path2/file3
mapleleaf@LAPTOP-GA89PJFV:~/path3$ ls
file4  package.tar  path1  path2
```

3. 符号链接内容

（1）新建文件 foo.txt ，内容为 123 ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ echo 123 > foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls
foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$
```

（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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ ln foo.txt bar.txt
mapleleaf@LAPTOP-GA89PJFV:~$ diff foo.txt bar.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls
bar.txt  foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$ comm foo.txt bar.txt
                123
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ ls -i -F foo.txt
1910 foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls -i -F bar.txt
1910 bar.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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi bar.txt
mapleleaf@LAPTOP-GA89PJFV:~$ diff bar.txt foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$ comm bar.txt foo.txt
                abc
mapleleaf@LAPTOP-GA89PJFV:~$ cat foo.txt
abc
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ ls
bar.txt  foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$ rm foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls -i -F bar.txt
1910 bar.txt
mapleleaf@LAPTOP-GA89PJFV:~$ cat bar.txt
abc
mapleleaf@LAPTOP-GA89PJFV:~$
```

（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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ ln -s bar.txt baz.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls
bar.txt  baz.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls -i -F bar.txt
1910 bar.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls -i -F baz.txt
1971 baz.txt@
mapleleaf@LAPTOP-GA89PJFV:~$ comm baz.txt bar.txt
                abc
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ rm bar.txt
mapleleaf@LAPTOP-GA89PJFV:~$ cat baz.txt
cat: baz.txt: No such file or directory
mapleleaf@LAPTOP-GA89PJFV:~$
```

4. 权限管理

（1）新建文件 qux.txt ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ touch qux.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls
qux.txt
mapleleaf@LAPTOP-GA89PJFV:~$
```

（2）为文件 qux.txt 增加执行权限（所有用户都可以执行）

```
mapleleaf@LAPTOP-GA89PJFV:~$ chmod a+x qux.txt
mapleleaf@LAPTOP-GA89PJFV:~$ ls -al qux.txt
-rwxr-xr-x 1 mapleleaf mapleleaf 0 May 22 22:37 qux.txt
mapleleaf@LAPTOP-GA89PJFV:~$
```

四、实验过程分析与讨论

对于很多命令的不熟悉导致做实验时频频出错，所以对 Linux 这门课需要不断地练习和记忆来达到孰能生巧。往往一个命令拥有很多的参数，能够实现各种各样的功能。其中 ln 命令生成硬链接时是使当前文件指向该文件的索引号，这样删除另一个文件，通过硬链接仍然可以访问到以前的数据。而生成软连接时实际上是生成了一个包含另一文件的位置信息的文本文件，当源文件删除，该文件便不可访问内容。

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 918	实验日期	2023 年 3 月 23 日
学　　号	2021213113	姓　　名	张学浩
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。

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 ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir workspace1
mapleleaf@LAPTOP-GA89PJFV:~$ ls
workspace1
mapleleaf@LAPTOP-GA89PJFV:~$
```

（2）进入目录 workspace1 ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ cd workspace1
mapleleaf@LAPTOP-GA89PJFV:~/workspace1$ ls
mapleleaf@LAPTOP-GA89PJFV:~/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：

```c
#include <stdio.h>

int main( )
{
    printf("hello world!\n");
    return 0;
}
```

```
mapleleaf@LAPTOP-GA89PJFV:~/workspace1$ cat test.c
#include <stdio.h>
int main()
{
        printf("hello world!\n");
        return 0;
}

mapleleaf@LAPTOP-GA89PJFV:~/workspace1$
```

（4）保存 test.c 的内容，并退出；

```
mapleleaf@LAPTOP-GA89PJFV:~/workspace1$ cat test.c
#include <stdio.h>
int main()
{
        printf("hello world!\n");
        return 0;
}

mapleleaf@LAPTOP-GA89PJFV:~/workspace1$
```

（5）编译 test.c 文件，生成可执行文件 test ，并执行，查看执行结果。

```
mapleleaf@LAPTOP-GA89PJFV:~/workspace1$ gcc test.c -o test
mapleleaf@LAPTOP-GA89PJFV:~/workspace1$ ./test
hello world!
mapleleaf@LAPTOP-GA89PJFV:~/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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ mkdir workspace2
mapleleaf@LAPTOP-GA89PJFV:~$ ls
workspace2  workspace1
mapleleaf@LAPTOP-GA89PJFV:~$
```

（2）进入 workspace2 目录；

```
mapleleaf@LAPTOP-GA89PJFV:~$ cd workspace2
mapleleaf@LAPTOP-GA89PJFV:~/workspace2$ ls
mapleleaf@LAPTOP-GA89PJFV:~/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；

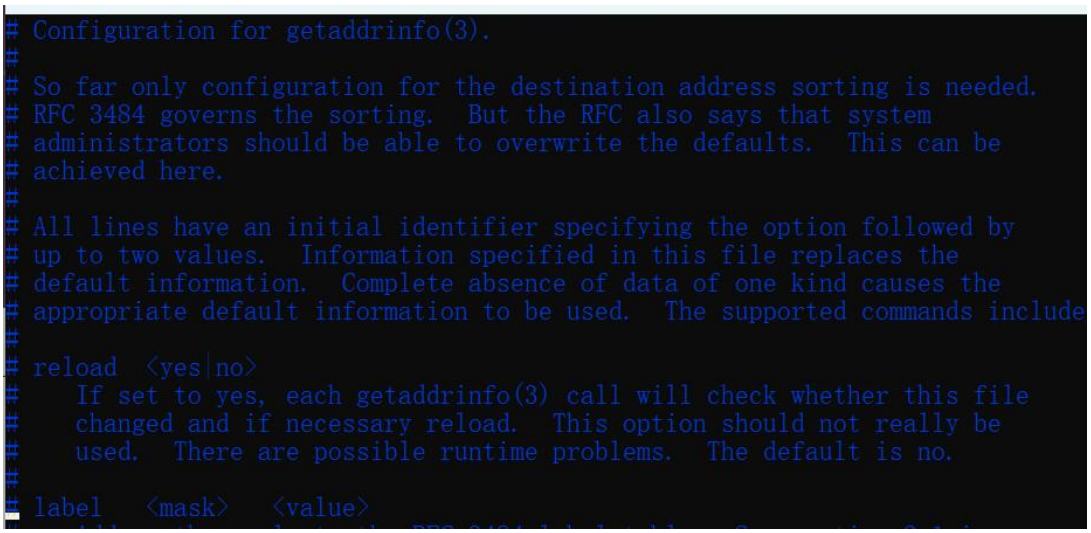
```
cat /etc/gai.conf > ./gai.conf
```

```
mapleleaf@LAPTOP-GA89PJFV:~$ cd workspace2
mapleleaf@LAPTOP-GA89PJFV:~/workspace2$ ls
mapleleaf@LAPTOP-GA89PJFV:~/workspace2$ cat /etc/gai.conf > ./gai.conf
mapleleaf@LAPTOP-GA89PJFV:~/workspace2$ ls
gai.conf
mapleleaf@LAPTOP-GA89PJFV:~/workspace2$
```

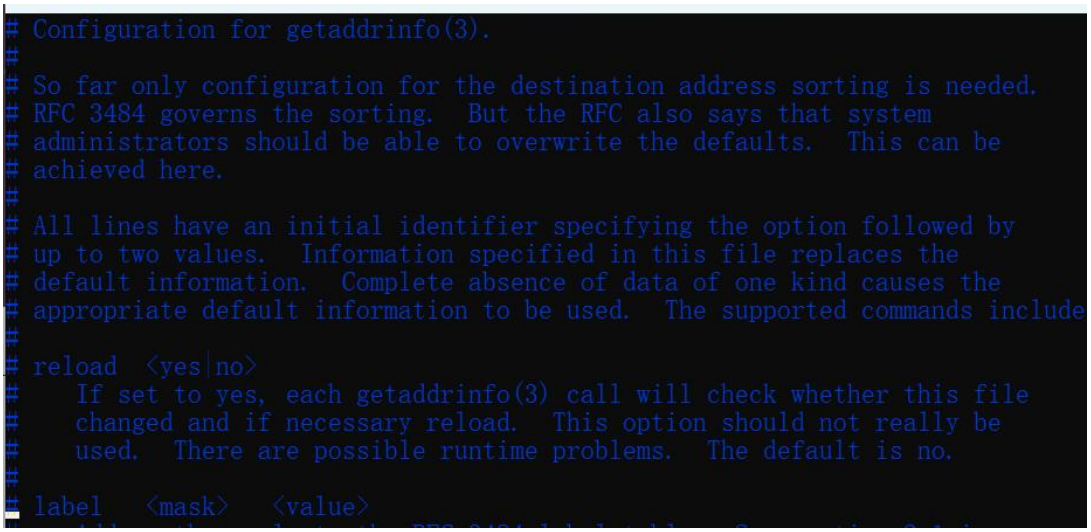
（4）使用 vim 编辑当前目录下的 gai.conf ；

```
# Configuration for getaddrinfo(3).
#
# So far only configuration for the destination address sorting is needed.
# RFC 3484 governs the sorting.  But the RFC also says that system
# administrators should be able to overwrite the defaults.  This can be
# achieved here.
#
# All lines have an initial identifier specifying the option followed by
# up to two values.  Information specified in this file replaces the
# default information.  Complete absence of data of one kind causes the
# appropriate default information to be used.  The supported commands include:
#
# reload      <yes|no>
#     If set to yes, each getaddrinfo(3) call will check whether this file
#     changed and if necessary reload.  This option should not really be
#     used.  There are possible runtime problems.  The default is no.
#
# label   <mask>  <value>
#     Add another rule to the RFC 3484 label table.  See section 2.1 in
#     RFC 3484.  The default is:
#
#label ::1/128       0
#label ::/0          1
#label 2002::/16     2
#label ::/96         3
#label ::ffff:0:0/96 4
#label fec0::/10     5
#label fc00::/7      6
#label 2001:0::/32   7
#
#    This default differs from the tables given in RFC 3484 by handling
#    (now obsolete) site-local IPv6 addresses and Unique Local Addresses.
-- INSERT --                                                    1,1           Top
```

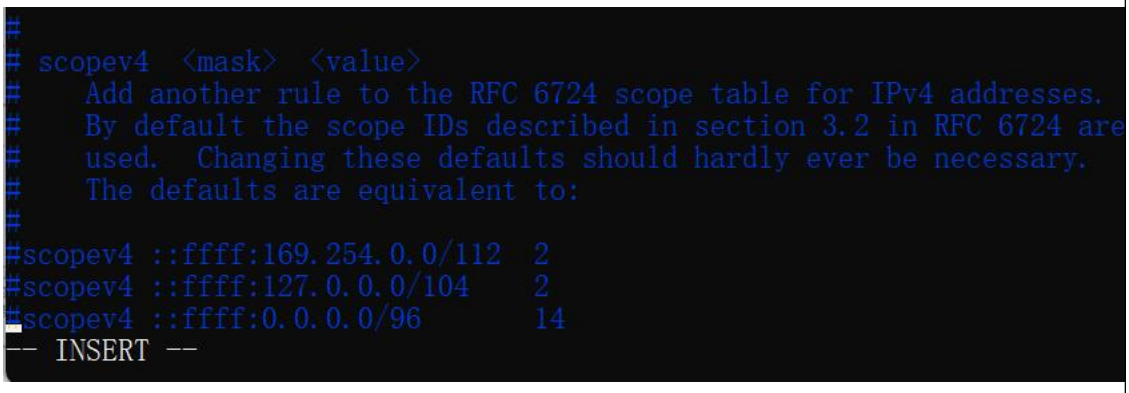
（5）将光标移到第 18 行；



（6）复制该行内容；



（7）将光标移到最后一行行首；



（8）粘贴复制行的内容;

```
#    For sites which prefer IPv4 connections change the last line to
#
#precedence ::ffff:0:0/96  100

#
# scopev4  <mask>  <value>
#    Add another rule to the RFC 6724 scope table for IPv4 addresses.
#    By default the scope IDs described in section 3.2 in RFC 6724 are
#    used.  Changing these defaults should hardly ever be necessary.
#    The defaults are equivalent to:
#
#scopev4 ::ffff:169.254.0.0/112  2
#scopev4 ::ffff:127.0.0.0/104    2
#scopev4 ::ffff:0.0.0.0/96       14
# label   <mask>   <value>
```

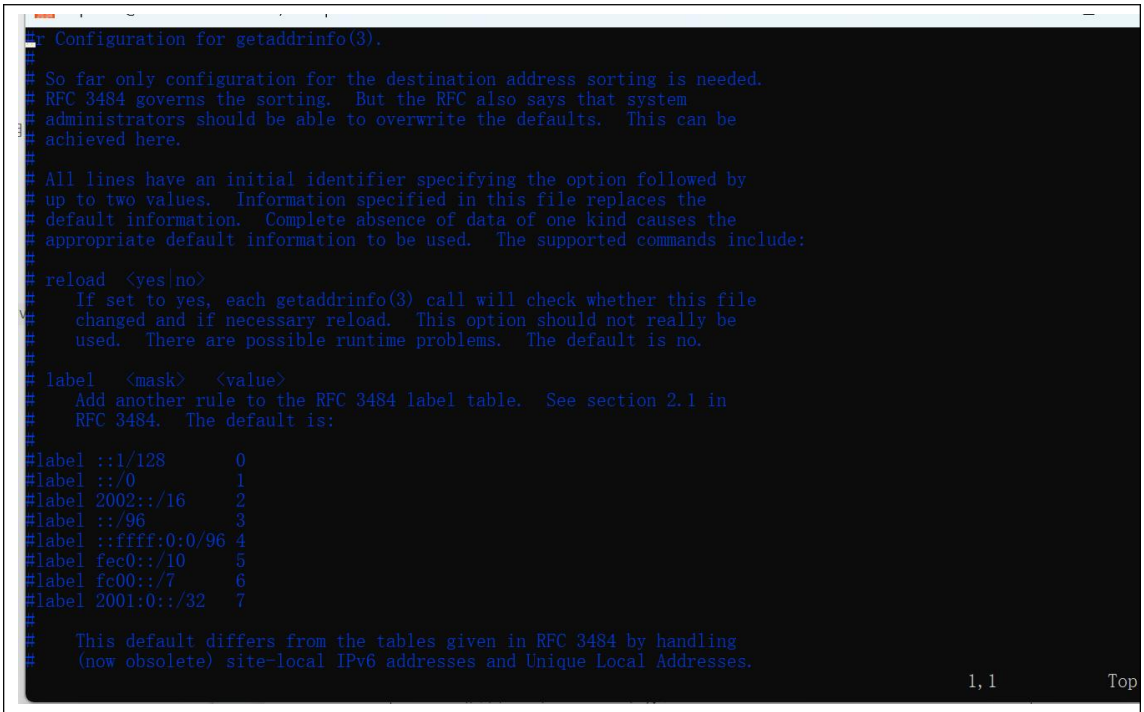
（9）撤销第 8 步的动作；

```
#
# scopev4  <mask>  <value>
#    Add another rule to the RFC 6724 scope table for IPv4 addresses.
#    By default the scope IDs described in section 3.2 in RFC 6724 ar
#    used.  Changing these defaults should hardly ever be necessary.
#    The defaults are equivalent to:
#
#scopev4 ::ffff:169.254.0.0/112  2
#scopev4 ::ffff:127.0.0.0/104    2
#scopev4 ::ffff:0.0.0.0/96       14
```

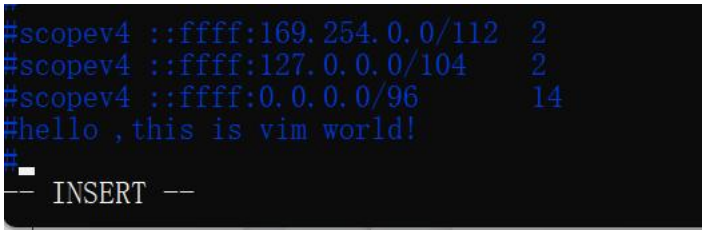
（10）存盘但不退出；

```
#    used.  Changing these defaults should har
#    The defaults are equivalent to:
#
#scopev4 ::ffff:169.254.0.0/112  2
#scopev4 ::ffff:127.0.0.0/104    2
#scopev4 ::ffff:0.0.0.0/96       14
"gai.conf" 65L, 2585C written
```

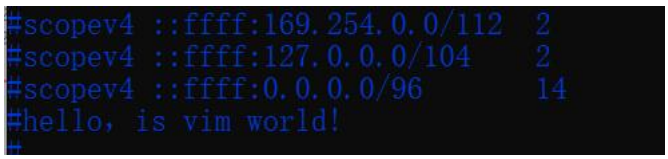
（11）将光标移到首行；



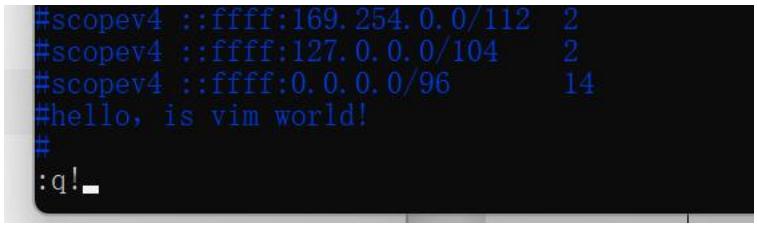
（12）插入模式下输入 "Hello, this is vim world!"



（13）删除字符串 "this" ；



（14）强制退出 vim ， 不存盘



四、实验过程分析与讨论

Vim 的使用操作很多很杂，但整体上来将难度并不大，只要跟着笔记和资料都是能轻松完成的。但是想要熟练运用，还是需要大量的记忆和联系，只要能熟悉这些操作，就能多快好省的完成编辑工作。

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 918	实验日期	2023 年 3 月 30 日
学　　号	2021213113	姓　　名	张学浩
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@LAPTOP-GA89PJFV:/home# useradd -d /home/foo -s /bin/sh -g users -m foo
root@LAPTOP-GA89PJFV:/home# usermod -c bar foo
root@LAPTOP-GA89PJFV:/home# tail -1 /etc/passwd
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
landscape:x:110:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
fwupd-refresh:x:112:116:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mapleleaf:x:1000:1000:,,,:/home/mapleleaf:/bin/bash
foo:x:1001:100:bar:/home/foo:/bin/sh
root@LAPTOP-GA89PJFV:/home# usermod -p 123456 foo
root@LAPTOP-GA89PJFV:/home# tail -1 /etc/passwd
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
landscape:x:110:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
fwupd-refresh:x:112:116:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mapleleaf:x:1000:1000:,,,:/home/mapleleaf:/bin/bash
foo:x:1001:100:bar:/home/foo:/bin/sh
root@LAPTOP-GA89PJFV:/home#
```

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@LAPTOP-GA89PJFV:/home# usermod -u 2000 -s /bin/csh foo
usermod: no changes
root@LAPTOP-GA89PJFV:/home# su foo
% tail -1 /etc/passwd
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
landscape:x:110:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
fwupd-refresh:x:112:116:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mapleleaf:x:1000:1000:,,,:/home/mapleleaf:/bin/bash
foo:x:2000:100:bar:/home/foo:/bin/csh
%
```

3. 从用户 foo 切换到 root ；

```
% exit
% exit
root@LAPTOP-GA89PJFV:/home#
```

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@LAPTOP-GA89PJFV:/home# userdel -r foo
userdel: foo mail spool (/var/mail/foo) not found
root@LAPTOP-GA89PJFV:/home# tail -1 /etc/passwd
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
landscape:x:110:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
fwupd-refresh:x:112:116:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mapleleaf:x:1000:1000:,,,:/home/mapleleaf:/bin/bash
root@LAPTOP-GA89PJFV:/home# cd foo
bash: cd: foo: No such file or directory
root@LAPTOP-GA89PJFV:/home# ll
total 12
drwxr-xr-x  3 root      root      4096 May 23 00:33 ./
drwxr-xr-x 19 root      root      4096 May 22 21:51 ../
drwxr-xr-x  3 mapleleaf mapleleaf 4096 May 22 23:57 mapleleaf/
root@LAPTOP-GA89PJFV:/home#
```

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@LAPTOP-GA89PJFV:/home# vi manage
root@LAPTOP-GA89PJFV:/home# vi userspass
root@LAPTOP-GA89PJFV:/home# newusers manage
root@LAPTOP-GA89PJFV:/home# chpasswd < userspass
root@LAPTOP-GA89PJFV:/home# cat etc/passwd | tail -n 3
cat: etc/passwd: No such file or directory
root@LAPTOP-GA89PJFV:/home# cat /etc/passwd | tail -n 3
user1:x:2001:2001:user1:/home/user1:/bin/bash
user2:x:2002:2002:user2:/home/user2:/bin/bash
user3:x:2003:2003:user3:/home/user3:/bin/bash
root@LAPTOP-GA89PJFV:/home#
```

6. 创建用户组 group1 ，并在创建时设置其 GID 为 3000 ；

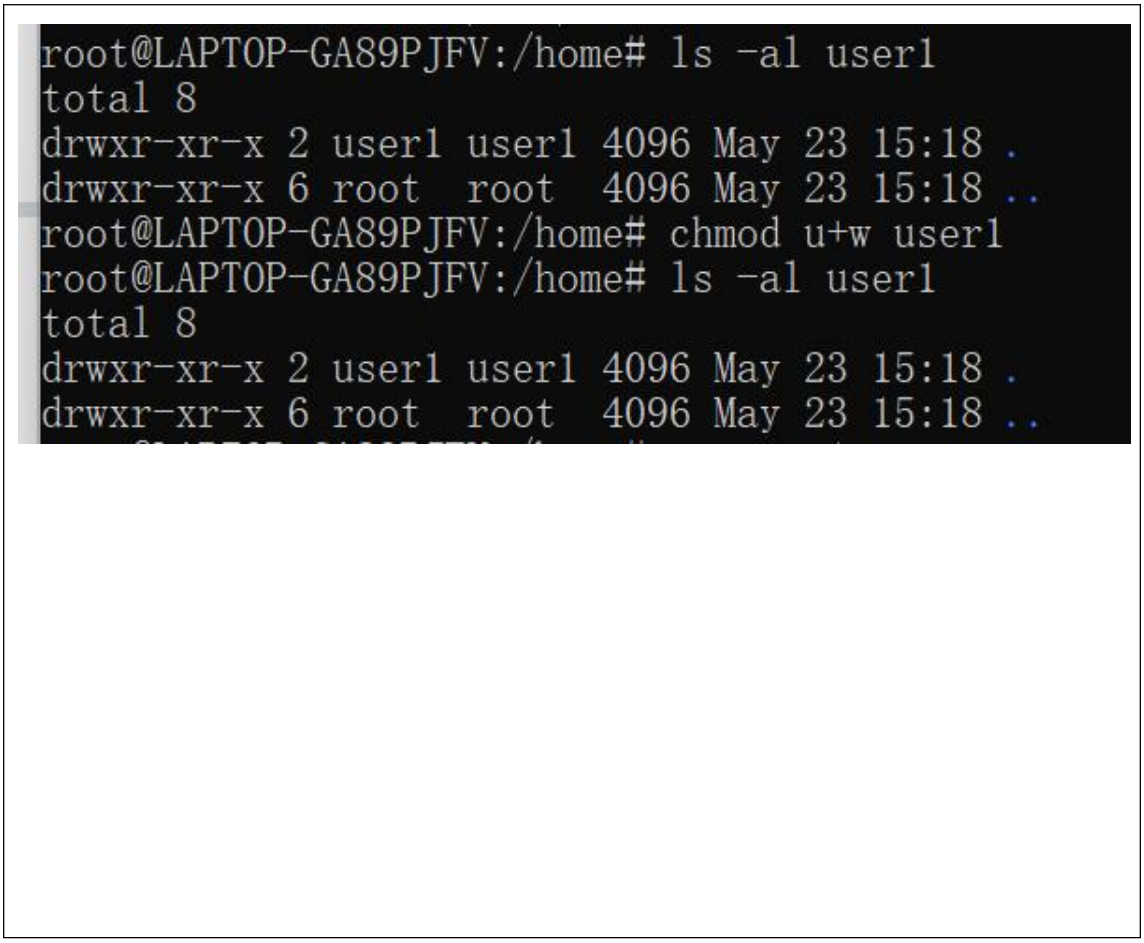
```
root@LAPTOP-GA89PJFV:/home# groupadd -g 3000 group1
root@LAPTOP-GA89PJFV:/home# tail -1 /etc/group
group1:x:3000:
root@LAPTOP-GA89PJFV:/home#
```

7. 在用户组 group1 中添加两个之前批量创建的用户；

```
root@LAPTOP-GA89PJFV:/home# gpasswd -a user1 group1
Adding user user1 to group group1
root@LAPTOP-GA89PJFV:/home# gpasswd -a user2 group1
Adding user user2 to group group1
```

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

```
root@LAPTOP-GA89PJFV:/# su user1
user1@LAPTOP-GA89PJFV:/$ sudo cat /etc/shadow | tail -n 3
[sudo] password for user1:
user1 is not in the sudoers file.  This incident will be reported.
user1@LAPTOP-GA89PJFV:/$ su
Password:
```


root@LAPTOP-GA89PJFV:/home# ls -al user1
total 8
drwxr-xr-x 2 user1 user1 4096 May 23 15:18 .
drwxr-xr-x 6 root root 4096 May 23 15:18 ..
root@LAPTOP-GA89PJFV:/home# chmod u+w user1
root@LAPTOP-GA89PJFV:/home# ls -al user1
total 8
drwxr-xr-x 2 user1 user1 4096 May 23 15:18 .
drwxr-xr-x 6 root root 4096 May 23 15:18 ..

四、实验过程分析与讨论

用户与组的操作看似比较简单，但并非如此，在记忆并运用操作指令的同时，还需要注意所操作内容的文件信息，以及每个字段代表的是什么意思。第一次切换到 root 用户时显示密码错误，但实际上我从未给 root 设置过密码，之后从我所建立的用户退出是，也出了点小问题，是上 csdn 上查找解决的。

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 918	实验日期	2023 年 4 月 6 日
学　　号	2021213113	姓　　名	张学浩
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

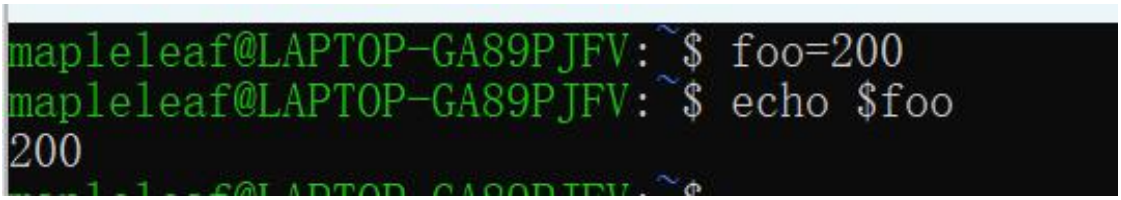
二、实验环境

（1）计算机的硬件配置 PC 系列微机。

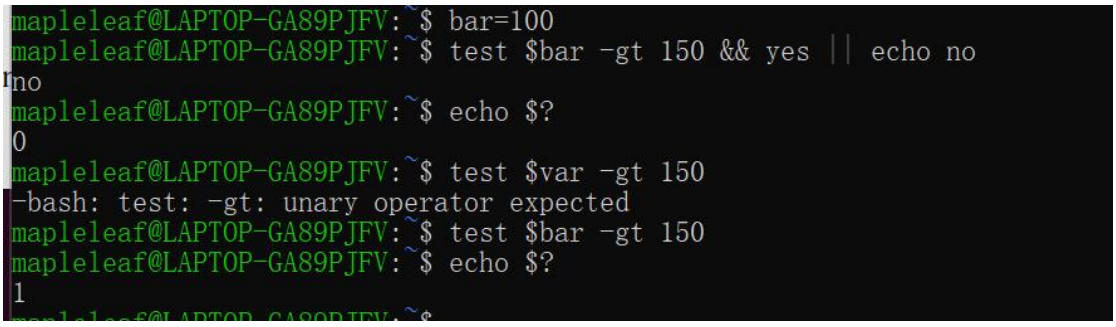
（2）计算机的软件配置 VMware 虚拟机软件及 Ubuntu 虚拟机。

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）；



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）；



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 ）；

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi hostname
mapleleaf@LAPTOP-GA89PJFV:~$ sh hostname
LAPTOP-GA89PJFV
Tue May 23 15:57:59 CST 2023
mapleleaf@LAPTOP-GA89PJFV:~$ cat hostname
#!/bin/bash
hn=$(cat /etc/hostname)
dt=$(date)
echo $hn
echo $dt
mapleleaf@LAPTOP-GA89PJFV:~$
```

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：
>
> ```
> 153 == 1^3+3^3+5^3
> ```
>
> 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。

```
mapleleaf@LAPTOP-GA89PJFV:~$ vim shuixianhua
mapleleaf@LAPTOP-GA89PJFV:~$ sh shuixianhua 153
a=1 b=5 c=3
a^3=1 b^3=125 c^3=27
Yes! 1 + 125 + 27 = 153
mapleleaf@LAPTOP-GA89PJFV:~$ cat shuixianhua
#!/bin/bash
num=$#   # get the number of parameter
if [ $num -lt 1 ]; then
        echo "The number of paramter is less than one!"
else
        x=$1 # get the first parameter
        if [ $x -gt 999 -o $x -lt 100 ]; then
                echo "The number must be between 100 and 999!"
        else
                        a=$(($x/100))
                        b=$(($x/10%10))
                        c=$(($x%10))
                        echo "a=$a b=$b c=$c"
                        a=$(($a*$a*$a))
                        b=$(($b*$b*$b))
                        c=$(($c*$c*$c))
                        echo "a^3=$a b^3=$b c^3=$c"
                        if [ $(($a+$b+$c)) = $x ]; then
                        echo "Yes! $a + $b + $c = $x"
                        else
                        echo "No!  $a + $b+ $c != $x"
                fi
         fi
mapleleaf@LAPTOP-GA89PJFV:~$
```

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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi sum
mapleleaf@LAPTOP-GA89PJFV:~$ ./sum
Please input the first number:11
Please input the second number:22
Please input the third number:33
66
mapleleaf@LAPTOP-GA89PJFV:~$ cat sum
#!/bin/bash
read -p "Please input the first number:" num1
read -p "Please input the second number:" num2
read -p "Please input the third number:" num3
echo "$((num1+num2+num3))"
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
if
elif
else
fi
```

实现。

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi grade
mapleleaf@LAPTOP-GA89PJFV:~$ sh grade 85
B
mapleleaf@LAPTOP-GA89PJFV:~$ sh grade 61
D
mapleleaf@LAPTOP-GA89PJFV:~$ sh grade 59
E
mapleleaf@LAPTOP-GA89PJFV:~$ cat grade
#!/bin/bash
num=$#   # get the number of parameter
if [ $num -lt 1 ]; then
        echo "The number of paramter is less than one!"
else
        if [ $1 -lt 0 -o $1 -gt 100 ]; then
                echo "The grade must be between 0 and 100!"
        elif [ $1 -ge 90 ]; then
                echo "A"
        elif [ $1 -ge 80 -a $1 -lt 90 ]; then # >=80 && <90
                echo "B"
        elif [ $1 -ge 70 -a $1 -lt 80 ]; then
                echo "C"
        elif [ $1 -ge 60 -a $1 -lt 70 ]; then
                echo "D"
        else
                echo "E"
        fi
fi
```

四、实验过程分析与讨论

运用 shell 脚本进行编程时，需要熟练地掌握 vim 的使用方法。另外在编程中如果想使用某个命令的返回值时，要么使用反引，要么使用美元符号加括号。在进行数字运算的时候，要用$(())的格式，或者使用命令。而且有些时候 sh 命令并不能解释带数字运算的语句，这个时候可以用 bash。

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 918	实验日期	2023 年 4 月 15 日
学　　号	2021213113	姓　　名	张学浩
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。

```
mapleleaf@LAPTOP-GA89PJFV: /test$ cat >c1.c
123
mapleleaf@LAPTOP-GA89PJFV:~/test$ cat >c2.c
456
mapleleaf@LAPTOP-GA89PJFV:~/test$ cat >c3.c
789
mapleleaf@LAPTOP-GA89PJFV:~/test$ vi copy.c
mapleleaf@LAPTOP-GA89PJFV:~/test$ sh copy.c
./c1.c ./c2.c ./c3.c ./copy.c
Please input dir path: /home/mapleleaf/workspace
mapleleaf@LAPTOP-GA89PJFV:~/test$ cd ..
mapleleaf@LAPTOP-GA89PJFV:~$ cd /workspace
-bash: cd: /workspace: No such file or directory
mapleleaf@LAPTOP-GA89PJFV:~$ cd workspace
mapleleaf@LAPTOP-GA89PJFV:~/workspace$ ls
c1.c  c2.c  c3.c  copy.c
mapleleaf@LAPTOP-GA89PJFV:~/workspace$ cat copy.c
#!/bin/bash
filelist=`ls ./*c`
echo $filelist
read -p "Please input dir path: " dir
test -e $dir || mkdir $dir
for i in $filelist
do
        cp $i $dir
        chmod a+x $dir/$i
done
```

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

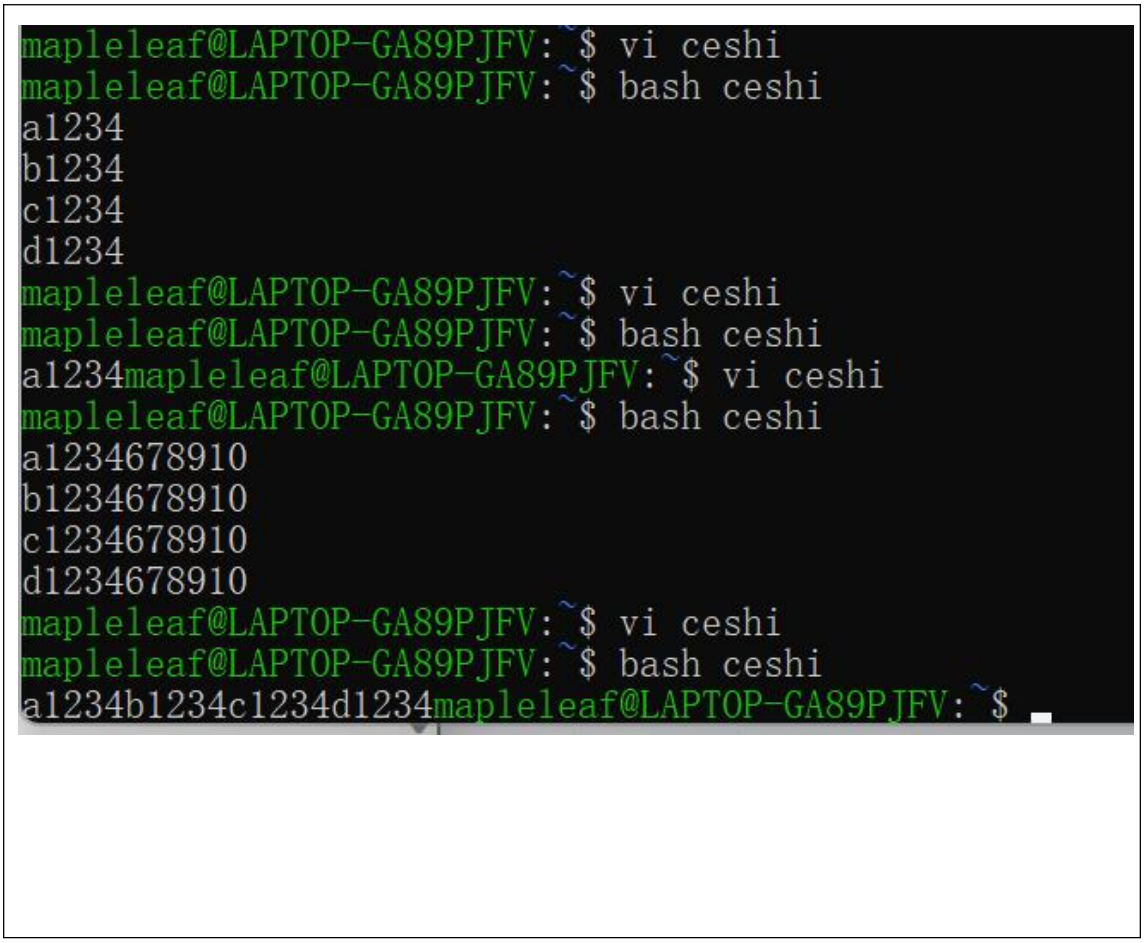
```
mapleleaf@LAPTOP-GA89PJFV:~$ vi qiuhe
mapleleaf@LAPTOP-GA89PJFV:~$ bash qiuhe
110
mapleleaf@LAPTOP-GA89PJFV:~$ cat qiuhe
#!/bin/bash
i=2
sum=0
while(($i<=20))
do
        sum=$((sum+i))
        i=$((i+2))
done
echo $sum
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi quater
mapleleaf@LAPTOP-GA89PJFV:~$ bash quater
385
mapleleaf@LAPTOP-GA89PJFV:~$ cat quater
#!/bin/bash

i=1
sq=0
until [ $i -ge 11 ]
do
   temp=$((i*i))
   sq=$((sq+temp))
   i=$((i+1))
done
echo $sq
mapleleaf@LAPTOP-GA89PJFV:~$
```

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 in 1 2 3 4 5 6 7 8 9 10; do
        if [[ $j -eq 5 ]]; then
            ---
        fi
        echo -n $j
    done
    echo ''
done
```


mapleleaf@LAPTOP-GA89PJFV:~$ vi ceshi
mapleleaf@LAPTOP-GA89PJFV:~$ bash ceshi
a1234
b1234
c1234
d1234
mapleleaf@LAPTOP-GA89PJFV:~$ vi ceshi
mapleleaf@LAPTOP-GA89PJFV:~$ bash ceshi
a1234mapleleaf@LAPTOP-GA89PJFV:~$ vi ceshi
mapleleaf@LAPTOP-GA89PJFV:~$ bash ceshi
a1234678910
b1234678910
c1234678910
d1234678910
mapleleaf@LAPTOP-GA89PJFV:~$ vi ceshi
mapleleaf@LAPTOP-GA89PJFV:~$ bash ceshi
a1234b1234c1234d1234mapleleaf@LAPTOP-GA89PJFV:~$

四、实验过程分析与讨论

Until 循环条件中若为假继续循环，若为真则跳出循环，这一点是我们平常容易混淆的地方。另外在引用变量时千万不要忘记加‘$’符号。Break 是终止一层循环，continue 是跳出当前循环；而 break 2 是终止两层循环，continue 2 是跳过两个循环。

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 918	实验日期	2023 年 4 月 22 日
学　　号	2021213113	姓　　名	张学浩
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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi qiuhe
mapleleaf@LAPTOP-GA89PJFV:~$ bash qiuhe
please input the first number:5
please input the second number:6
11
mapleleaf@LAPTOP-GA89PJFV:~$ cat qiuhe
#!/bin/bash

read -p"please input the first number:" a
read -p"please input the second number:" b
sum(){
result=$((a+b))
echo $result
}
sum
mapleleaf@LAPTOP-GA89PJFV:~$
```

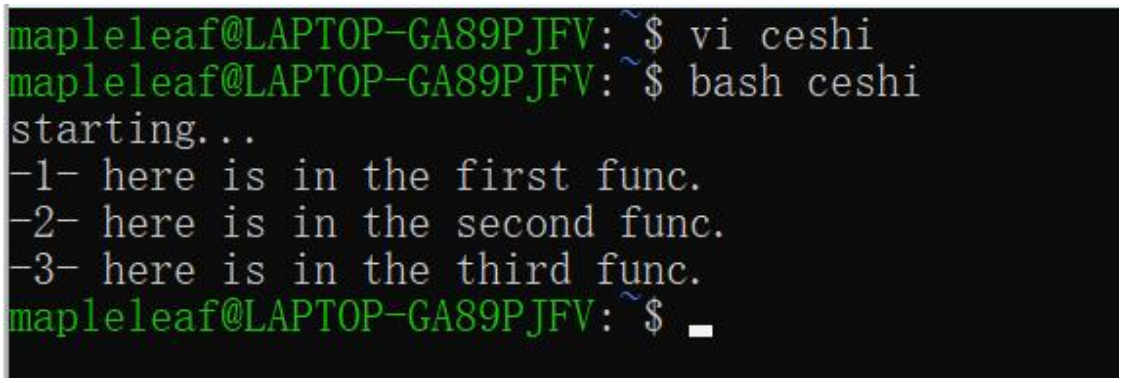
2. 编写Shell脚本，在脚本中定义一个递归函数，实现 n 的阶乘的求解；

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi jiecheng
mapleleaf@LAPTOP-GA89PJFV:~$ bash jiecheng
please input a positive number:5
120
mapleleaf@LAPTOP-GA89PJFV:~$ bash jiecheng
please input a positive number:6
720
mapleleaf@LAPTOP-GA89PJFV:~$ cat jiecheng
#!/bin/bash

factor(){
        if [ $1 -eq 1 ]
            then echo 1
        else
            local temp=$[ $1 - 1]
            local num=$(factor $temp)
            echo $[ $1 * $num ]
        fi
}
read -p "please input a positive number:" n
sum=$(factor $n)
echo $sum
mapleleaf@LAPTOP-GA89PJFV:~$
```

3. 一个Shell脚本的内容如下所示

```
#!/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
```

试运行该程序，并观察程序运行结果，理解函数嵌套的含义



四、实验过程分析与讨论

Shell 函数的括号里不可以像其它编程语言直接传参，但是可以通过函数名再在后面加上参数，就可以在函数内部用$1,$2……的方式调用参数了。函数嵌套即在一个函数中可以定义另一个函数，如此往复，而且不论在那一层都可以调用定义过的函数。

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 918	实验日期	2023 年 5 月 3 日
学　　号	2021213113	姓　　名	张学浩
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；

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi quote.txt
mapleleaf@LAPTOP-GA89PJFV:~$ 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.
mapleleaf@LAPTOP-GA89PJFV:~$
```

（2）显示包含 music 文字的行内容及行号；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sed -n '/music/{=;p}' quote.txt
2
It was an evening of splendid music and company.
mapleleaf@LAPTOP-GA89PJFV:~$
```

（3）在第 4 行后面追加内容： "hello world!" ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sed '$ahello world!' quote.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
hello world!
mapleleaf@LAPTOP-GA89PJFV:~$
```

（4）将文本 "The" 替换为 "Quod" ；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sed 's/The/Quod/' quote.txt
Quod honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
Too bad the disco floor fell through at 23:10.
Quod local nurse Miss P.Neave was in attendance.
mapleleaf@LAPTOP-GA89PJFV:~$
```

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

```
mapleleaf@LAPTOP-GA89PJFV:~$ sed '3cThis is the third line.' quote.txt
The honeysuckle band played all night long for only $90.
It was an evening of splendid music and company.
This is the third line.
The local nurse Miss P.Neave was in attendance.
mapleleaf@LAPTOP-GA89PJFV:~$
```

（6）删除第 2 行内容；

```
mapleleaf@LAPTOP-GA89PJFV:~$ sed '2d' quote.txt
The honeysuckle band played all night long for only $90.
Too bad the disco floor fell through at 23:10.
The local nurse Miss P.Neave was in attendance.
mapleleaf@LAPTOP-GA89PJFV:~$
```

（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。

```
mapleleaf@LAPTOP-GA89PJFV: $ var='evening'
mapleleaf@LAPTOP-GA89PJFV:~$ sed -n '/'$var'/p' quote.txt
It was an evening of splendid music and company.
mapleleaf@LAPTOP-GA89PJFV:~$
```

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；

```
mapleleaf@LAPTOP-GA89PJFV: $ vi numbers.txt
mapleleaf@LAPTOP-GA89PJFV:~$ awk -F " " '{print $2}' numbers.txt
:
:
mapleleaf@LAPTOP-GA89PJFV:~$ awk -F ":" '{print $2}' numbers.txt
 two
 five
mapleleaf@LAPTOP-GA89PJFV:~$
```

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：

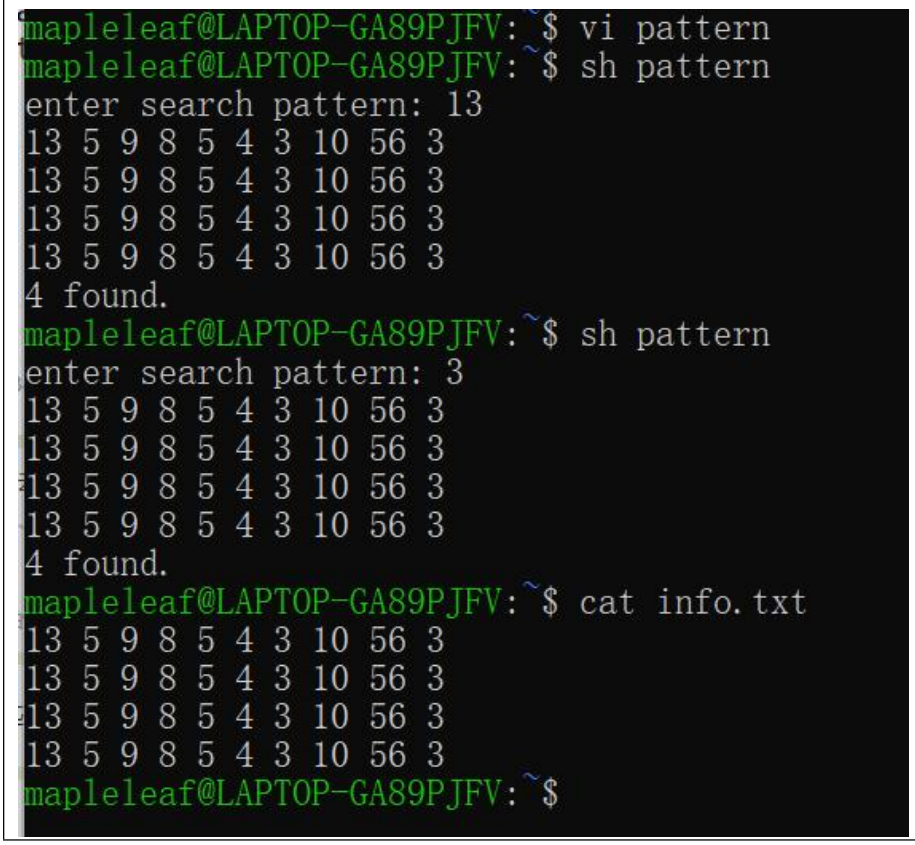
```
even:
2
4
46
numbers:
3
```

```
mapleleaf@LAPTOP-GA89PJFV:~$ vi foo.txt
mapleleaf@LAPTOP-GA89PJFV:~$ vi chengxu
mapleleaf@LAPTOP-GA89PJFV:~$ bash chengxu
even
2
4
46
number:
3
mapleleaf@LAPTOP-GA89PJFV:~$ cat chengxu
#!/bin/bash

cont=0
echo even
for j in `awk '{print $0}' foo.txt`
do
    if [ $((j%2)) -eq 0 ]
    then echo $j
         cont=$((cont+1))
    fi
done
echo number:
echo $cont
mapleleaf@LAPTOP-GA89PJFV:~$
```

4. 脚本的内容如下所示：

```
#!/bin/bash

read –p "enter search pattern: " pattern

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

试运行该脚本，并理解该脚本实现的功能。



四、实验过程分析与讨论

本实验的第 4 题脚本实现功能输入要匹配的模式串，程序返回模式串所在行的内容以及出现的次数。Sed 和 awk 实现的功能是非常强大的，利用好了可以解决很多问题

五、指导教师意见

指导教师签字：卢洋