# 第3章 文件系统操作命令

## 3.1 文件系统的基本概念

Linux 系统中，每一个分区都是一个文件系统，都有自己的目录层次结构，而每一个文件系统具有不同的格式，这些格式决定了信息被存储为文件或目录的格式，不同的存储格式就是不同的文件系统类型。目前，Linux 系统支持大部分文件系统，常见类型如下。

(1) ext2 和 ext3：Linux 系统默认的文件系统类型。

(2) RAMFS：内存文件系统，速度很快。

(3) NFS：网络文件系统，用于远程文件共享。

(4) FAT 和 NTFS：Windows 操作系统采用的文件系统。

(5) HPFS：OS/2 操作系统采用的文件系统。

(6) PROC：虚拟进程文件系统。

(7) ISO9660：光盘文件系统。

Linux 系统采用树状结构组织文件，为每个文件分配文件块，然后把数据存储在存储设备中。不同的文件系统用不同的方式分配和读取文件，Linux 系统常用的文件分配策略为块分配和扩展分配。块分配是将磁盘上的文件根据需要分配给文件，这种方式可以避免存储空间的浪费，但当一个文件不断扩充时，就可能造成文件中的文件块不连续，从而导致过多的磁盘寻道时间，当读取一个文件时，可能会随机读取，而不是连续读取，读取效率会降低。优化文件块分配策略可以避免文件块的随机分配，尽可能实现块的连续分配，减少磁盘的寻道时间。块分配策略是当一个文件增大时为文件分配磁盘空间，而扩展分配是当某个文件的磁盘空间不足时一次性分配一连串连续的块。扩展分配可以优化磁盘寻道时间，可以成组的分配块有利于一次写入一大批数据到存储设备中，从而减少 SCSI 设备写数据的时间。

## 3.2 文件的操作

### 3.2.1 复制

cp(copy)命令的功能是将一个文件或目录从一个位置复制到另一个位置，命令格式如下。

```
cp  [option] 源文件名  目标文件名
```

常用的 option 选项有如下 4 种。

(1) -i：interactive，交互的，在复制之前给出提示信息。

(2) -r：recursive，递归的复制目录，当复制一个目录时，复制该目录中的所有内容，包括子目录中的所有文件。

(3) -p：preserve，维持，保留一些特定的属性。

(4) -f：force，强制的，如目标文件已经存在，不询问是否覆盖，而是直接覆盖复制。

执行复制操作时，文件名可以使用相对路径(只写文件名，不写路径，默认在工作目录下操作)，也可以使用绝对路径(对其他地方的文件执行复制操作)。普通用户无权将文件复制到/root 目录下，但 root 用户可以把文件复制到其他用户的目录下。

**例 3.1** root 用户将文件/root/mydream.txt 复制到/home/jerrymos 目录下。

```
root@liuhui-VirtualBox:~# cp mydream.txt /home/jerrymos/ mydream.txt
```

例 3.1 运行结果没有显示，可以通过 ls 命令查看/home/jerrymos 目录下的内容来确认是否正确地把 mydream.txt 文件复制到了目的地。

**例 3.2** 确认复制命令的运行结果。

```
root@liuhui-VirtualBox:~# ls -l /home/jerrymos/
total 4
-rw-r--r-- 1 root root    0  9 月 17 13:10 1.txt
-rw-r--r-- 1 root root 2059  9 月 17 13:09 mydream.txt
```

例 3.2 运行结果显示，root 用户的 mydream.txt 文件复制到了/home/jerrymos 目录下。

## 3.2.2 剪切和重命名

mv(move)命令既可以在不同目录之间复制文件和目录，也可以在同一个目录中重命名文件和目录，被移动或重命名的文件不会发生改变，类似于 Windows 系统的 cut 命令。

假定当前登录用户是 liuhui，登录的工作目录是普通用户的家目录/home/liuhui，整理工作目录，把有关机器学习的资料放入一个新的目录 deeplearning 中，机器学习的资料都是以 dl 开头的文件。

### 1. 把文件从一个目录移动到另一个目录

先在工作目录/home/liuhui 下创建一个新的目录 deeplearning。

**例 3.3** 创建一个新的目录 deeplearning。

```
liuhui@liuhui-VirtualBox:~ $ mkdir deeplearning
```

**例 3.4** 查看工作目录/home/liuhui 下所有以 dl 开头的文件。

```
liuhui@liuhui-VirtualBox:~$ ls -l dl*
-rw-rw-r-- 1 liuhui liuhui  137  9月 17 15:17 dladress
-rw-rw-r-- 1 liuhui liuhui  537  9月 17 15:17 dlreadme
-rw-rw-r-- 1 liuhui liuhui  537  9月 17 13:33 dlreadtext
-rw-rw-r-- 1 liuhui liuhui 1085  9月 17 15:17 dlsvm
liuhui@liuhui-VirtualBox:~$
```

**例 3.5** 把/home/liuhui 目录中以 dl 开头的文件移动到子目录/deeplearning 中。

```
liuhui@liuhui-VirtualBox:~ $ mv dladress  deeplearning
```

```
liuhui@liuhui-VirtualBox:~$ mv dlreadme  deeplearning
liuhui@liuhui-VirtualBox:~$ mv dlsvm  deeplearning
liuhui@liuhui-VirtualBox:~$ mv dlreadtext deeplearning
```

也可以使用一个命令一次性移动多个文件。

**例 3.6** 用相对路径移动文件。

```
liuhui@liuhui-VirtualBox:~$ mv dladress dlreadme dlsvm dlreadtext deeplearning
```

**例 3.7** 用绝对路径移动文件。

```
liuhui@liuhui-VirtualBox:~$ mv dl*  /home/liuhui/deeplearning
```

例 3.6 和例 3.7 中的命令运行结果没有显示,可以用 ls 命令来查看。

**例 3.8** 查看 deeplearning 目录中是否有刚刚移来的 4 个文件。

```
liuhui@liuhui-VirtualBox:~$ mv dl*  deeplearning
liuhui@liuhui-VirtualBox:~$ ls -ls deeplearning
total 16
4 -rw-rw-r-- 1 liuhui liuhui  137  9月 17 15:17 dladress
4 -rw-rw-r-- 1 liuhui liuhui  537  9月 17 15:17 dlreadme
4 -rw-rw-r-- 1 liuhui liuhui  537  9月 17 13:33 dlreadtext
4 -rw-rw-r-- 1 liuhui liuhui 1085  9月 17 15:17 dlsvm
liuhui@liuhui-VirtualBox:~$
```

**例 3.9** 查看/home/liuhui 目录下是否还有例 3.5 中移走的 4 个文件 dladress、dlreadme、dlsvm、dlreadtext。

```
liuhui@liuhui-VirtualBox:~$ ls -l dl*
ls: cannot access dl*: No such file or directory
liuhui@liuhui-VirtualBox:~$
```

例 3.9 运行结果显示工作目录/home/liuhui 下面没有了移走的两个文件,表明 mv 命令把文件移走了。

### 2. 在同一个目录中重命名文件

将子目录/deeplearning 中的文件 dlsvm 重命名为 dlsvmcn。

**例 3.10** 用 mv 命令重命名文件。

```
liuhui@liuhui-VirtualBox:~$ mv  deeplearning/dlsvm  deeplearning/dlsvmcn
```

例 3.10 运行结果没有显示,可以用 ls 命令查验更名是否成功。

**例 3.11** 查验 mv 命令的更名功能。

```
liuhui@liuhui-VirtualBox:~$ mv deeplearning/dlsvm  deeplearning/dlsvmcn
liuhui@liuhui-VirtualBox:~$ ls -l deeplearning
total 16
-rw-rw-r-- 1 liuhui liuhui  137  9月 17 15:17 dladress
-rw-rw-r-- 1 liuhui liuhui  537  9月 17 15:17 dlreadme
-rw-rw-r-- 1 liuhui liuhui  537  9月 17 13:33 dlreadtext
-rw-rw-r-- 1 liuhui liuhui 1085  9月 17 15:17 dlsvmcn
liuhui@liuhui-VirtualBox:~$
```

例 3.11 运行结果显示,子目录 deeplearning 中的 dlsvm 文件不见了,而出现了 dlsvmcn。

### 3. 移动文件并同时完成重命名

把 deeplearning 子目录下的文件 dlreadme 移到工作目录/home/liuhui 下,同时更名为

dlrdmetext。

**例 3.12** 移动并更名。

```
liuhui@liuhui-VirtualBox:~$ mv  deeplearning/dlreadme  dlrdmetext
```

例 3.12 运行结果没有显示，用户可以用 ls 命令验证，也可以直接在图形界面中选择 files→computer→home→liuhui 命令进行查看，可以看到 liuhui 目录下增加了 dlrdmetext 文件，而 deeplearning 目录下的文件 dlreadme 不见了，如图 3.1 和图 3.2 所示。

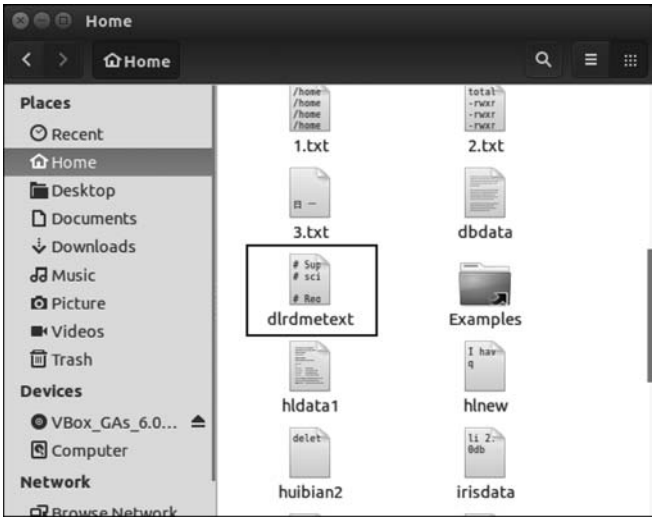



图 3.1 /home/liuhui 目录下多出来 dlrdmetext 文件

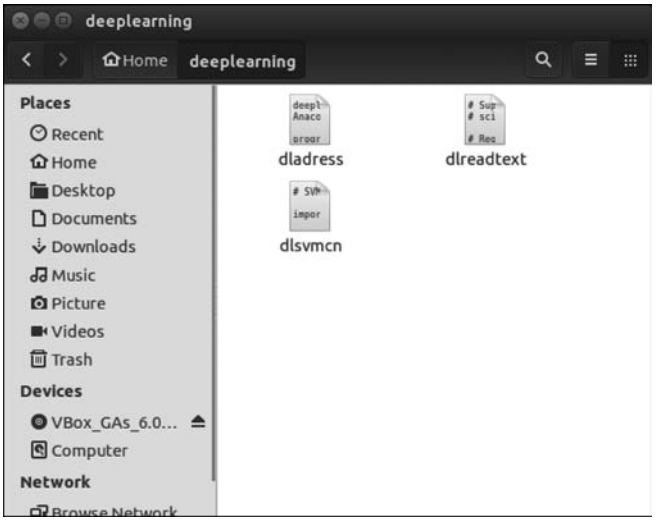



图 3.2 deeplearning 目录下的 dlreadme 文件消失了

### 4. 重命名目录

**例 3.13** 把 deeplearning 目录名变更为 machinelearn。

```
liuhui@liuhui-VirtualBox:~$ mv deeplearning  machinelearn
```

验证重命名的效果。

**例 3.14** 查看/home/liuhui 目录下的所有子目录及文件。

```
liuhui@liuhui-VirtualBox:~$ mv deeplearning/dlreadme  dlrdmetext
liuhui@liuhui-VirtualBox:~$ mv deeplearning  machinelearn
liuhui@liuhui-VirtualBox:~$ ls -R
.:
1.txt       Downloads          machinelearn  Picture        windata
2.txt       examples.desktop   mathdata      Pictures       windata~
3.txt       hldata1            mathdata~     pictures.tar   windata.fmt
backup      hlnew              Music         play.txt.gz    windata.spaces
dbdata      hlprivacy1         mydream~      Public
Desktop     huibian2           mydream.txt   t1.txt
dlrdmetext  irisdata           newdream      Templates
Documents   irisdata~          newdream~     Videos

./machinelearn:
dladress  dlreadtext  dlsvmcn
```

例 3.14 运行结果显示，工作目录/home/liuhui 下有一个 machinelearn 子目录。用 ls -R 命令把工作目录和它的子目录的内容全部显示出来。

### 5. 移动目录及目录下包含的文件

工作目录/home/liuhui 下有两个子目录 machinelearn 和 hlprivacy1，把目录 machinelearn 下的内容移动到另一个用户 jerrymos 的家目录下，这相当于在/home/jerrymos 目录下创建一个新目录 machinelearn。这涉及两个目录，所以需要 root 用户的权限，先用 su 命令切换到 root 用户，然后查看这两个目录的内容移动目录。

**例 3.15** 切换到 root 用户。

```
liuhui@liuhui-VirtualBox:~$ su  -  root
```

**例 3.16** 查看移动之前两个目录的内容。

```
[root@localhost ~]# ls /home/liuhui /machinelearn
[root@localhost ~]# ls /home/jerrymos
```

例 3.16 运行结果太多，此处省略。

**例 3.17** 移动目录。

```
[root@localhost ~]# mv /home/liuhui/machinelearn  /home/ jerrymos
```

例 3.17 运行结果不显示，可用 ls 命令分别查看/home/liuhui 目录和/home/ jerrymos 目录下的文件，验证移动结果，再用 ls 命令查看验证是否把 machinelearn 目录下的所有内容都移动到了新位置。

**例 3.18** 验证目录的移动结果。

```
root@liuhui-VirtualBox:~# mv /home/liuhui/machinelearn  /home/jerrymos
root@liuhui-VirtualBox:~# ls /home/liuhui
1.txt       Documents          irisdata      newdream       t1.txt
2.txt       Downloads          irisdata~     newdream~      Templates
3.txt       examples.desktop   mathdata      Picture        Videos
backup      hldata1            mathdata~     Pictures       windata
dbdata      hlnew              Music         pictures.tar   windata~
Desktop     hlprivacy1         mydream~      play.txt.gz    windata.fmt
dlrdmetext  huibian2           mydream.txt   Public         windata.spaces
root@liuhui-VirtualBox:~# ls /home/jerrymos
1.txt  machinelearn  machinelearning  mydream.txt
root@liuhui-VirtualBox:~# _
```

例 3.18 运行结果显示，目录/home/liuhui 下没有了子目录/machinelearn，而目录/home/jerrymos 下多了一个子目录/machinelearn。

**例 3.19** 验证目录中的文件是否移动到新位置。

```
root@liuhui-VirtualBox:~# ls -l /home/jerrymos/machinelearn
total 12
-rw-rw-r-- 1 liuhui liuhui  137  9月 17 15:17 dladress
-rw-rw-r-- 1 liuhui liuhui  537  9月 17 13:33 dlreadtext
-rw-rw-r-- 1 liuhui liuhui 1085  9月 17 15:17 dlsvmcn
root@liuhui-VirtualBox:~# _
```

例 3.19 运行结果显示，不但目录移动到了新位置，目录下所有的文件等内容也全部移动到了新位置。

### 3.2.3 文件的创建

#### 1. touch 命令

touch 命令可以创建一个空文件，也可以同时创建多个文件，其语法形式如下。

```
touch 文件名
```

文件名可以是绝对路径名，也可以是相对路径名；文件名可以是多个，每个文件名之间用空格分隔。

1）用相对路径创建文件

用相对路径创建文件时省略路径，就是在工作目录下创建文件。

**例 3.20** 当前用户 liuhui 登录后，进入默认的家目录/home/liuhui，在该目录下创建一个新文件 irisdata。

```
liuhui@liuhui-VirtualBox:~ $ touch irisdata
```

**例 3.21** 查看新建的文件。

```
liuhui@liuhui-VirtualBox:~ $ ls -l
```

为节省篇幅，此处省略运行结果，从例 3.21 运行结果看，文件 irisdata 已经存在，文件的大小为 0，创建时间为 9 月 17 14:30。为了与后面建立的重名的文件 irisdata 作对比，这里在 gedit 编辑器中打开 irisdata 文件写入一些内容，然后保存。此时，irisdata 文件的大小为 43bytes。

2）用绝对路径创建文件

用绝对路径创建文件将会指明在其他目录路径下创建一个新文件。

**例 3.22** 在/home/liuhui/backup 目录下新建文件 hldata。

```
liuhui@liuhui-VirtualBox:~ $ touch  /home/liuhui/backup/ hldata
```

**例 3.23** 验证例 3.22 是否创建成功。

```
liuhui@liuhui-VirtualBox:~ $ ls -l /home/liuhui/backup
```

为节省篇幅，此处省略运行结果。例 3.23 运行结果显示正确创建了新文件，文件的大小为 0 字节，表明创建的是空文件。

3）处理重名文件

如果新建的文件与已有的文件重命，会出现什么情况呢？在例 3.23 中已经在/home/liuhui/目录下有了一个文件 irisdata，且文件大小为 43bytes，最后修改日期为 9 月 17 14:30。现在先在该目录下新建 3 个文件，便于演示如何处理重名文件。

**例 3.24**　创建 3 个文件，文件名分别为 irisdata、mathdata、dbdata。

```
liuhui@liuhui-VirtualBox:~$ touch  /home/liuhui/irisdata mathdata dbdata
```

例 3.24 中多个文件名之间用空格分隔，命令的运行结果不显示，可以查看创建的结果。

**例 3.25**　重名文件的处理。

```
liuhui@liuhui-VirtualBox:~$ touch irisdata mathdata dbdata
liuhui@liuhui-VirtualBox:~$ ls -l *data
-rw-rw-r-- 1 liuhui liuhui  0  9月 17 15:31 dbdata
-rw-rw-r-- 1 liuhui liuhui 43  9月 17 15:31 irisdata
-rw-rw-r-- 1 liuhui liuhui  0  9月 17 15:31 mathdata
liuhui@liuhui-VirtualBox:~$
```

例 3.25 运行结果显示，新创建 mathdata 和 dbdata 两个文件的时间是 9 月 17 15:31，文件大小是 0；而 irisdata 文件的创建时间也是 9 月 17 15:31，文件的大小是 43bytes。但是第一次在例 3.20 中创建 irisdata 文件的时间为 9 月 17 14:30，，说明第二次创建的重名的空文件时会把已经存在的文件的创建时间修改为最后一次创建时间，或者说是把第一次创建的文件的访问时间修改为最后一次的时间，但文件内容还是第一次创建时形成的内容。

### 2. file 命令

在 Linux 系统中，文件的扩展名并不代表文件的类型，而打开不同类型的文件使用的命令也不一样，所以在打开文件之前，需要先确定文件的类型。确定文件类型的命令是 file，命令的语法格式如下。

```
file 文件名
```

要查看文件类型，需要先查看用户的家目录中的所有文件。

**例 3.26**　用 ls -F 命令查看普通用户 liuhui 的家目录下的文件。

```
liuhui@liuhui-VirtualBox:~$ ls -F
1.txt       Documents/        irisdata     newdream     t1.txt
2.txt       Downloads/        irisdata~    newdream~    Templates/
3.txt       examples.desktop  mathdata     Picture/     Videos/
backup/     hldata1           mathdata~    Pictures/    windata
dbdata      hlnew             Music/       pictures.tar windata~
Desktop/    hlprivacy1/       mydream~     play.txt.gz  windata.fmt
dlrdmetext  huibian2          mydream.txt  Public/      windata.spaces
liuhui@liuhui-VirtualBox:~$
```

由例 3.26 运行结果可见，用户 liuhui 家目录下有很多目录和文件，用 file 命令来查看文件的类型。

**例 3.27**　查看 Windows 系统文件的类型。

```
liuhui@liuhui-VirtualBox:~$ file  t1.txt
t1.txt: ISO-8859 English Text, with long lines, with CRLF line terminators
```

例 3.27 运行结果显示，该文件是一个内容为英文的正文文件，详细地说明了文件的长度、

回车结束符号等信息。

**例 3.28** 查看 Linux 系统文件的类型。

```
liuhui@liuhui-VirtualBox:~ $ file mathdata
mathdata:UTF-8 Unicode Text
```

例 3.28 运行结果只显示 Unicode 码的正文,显示的信息比较少。

**例 3.29** 查看目录的类型。

```
liuhui@liuhui-VirtualBox:~ $ file deeplearning
deeplearning:directory
```

从上述几个例子可以看出,file 命令的运行结果与 ls -F 命令的显示结果非常类似,差别只是 file 的信息更详细而已。

### 3.2.4 编辑

#### 1. gedit

gedit 是图形化的文本编辑工具,是一个 GNOME 桌面环境下兼容 UTF-8 的文本编辑器。gedit 使用 GTK+编写而成,十分简单易用,有良好的语法高亮,对中文支持度很好,支持包括 GB2312、GBK 在内的多种字符编码。gedit 是一个自由软件,是 Linux 系统下的一个纯文本编辑器,也可以认为是一个集成开发环境(Integrated Development Environment,IDE),它会根据不同的语言高亮显示关键字和标识符,类似于 Windows 系统下的一个文本编辑环境。

在终端的命令行方式下,输入命令 gedit,按 Enter 键,就可以启动 gedit 编辑器。

**例 3.30** 使用 gedit 编辑文件 hldata1,运行界面如图 3.3 所示。

```
liuhui@liuhui-VirtualBox:~ $ gedit hldata1
```


图 3.3 gedit 的运行界面

gedit 的使用非常简单,其支持键盘和鼠标的复制粘贴操作。

#### 2. vi

vi 是 UNIX 和 Linux 系统内嵌的标准的全屏正文编辑器,是以命令行方式输入的正文编

辑器，它可以在图形界面没有启动的情况下工作。如果系统出现问题，图形界面将无法启动，使用 vi 进行系统维护就是唯一的方法。

简单的程序编写，使用 gedit 工具就可以完成任务；在编写复杂的程序和系统无法正常启动的情况下，使用 vi 更合适。但是 vi 的使用方法复杂，更适合于专业人员使用，只有经过反复的练习才可以真正掌握。

vi 是 visual interface to the ex editor 的前两个单词的首字母，ex 是 UNIX 系统上的一种行编辑器。当用 vi 编辑一个正文文件时，vi 将文件中的所有正文放入一个内存缓冲区，后续的操作都是在内存缓冲区进行的。vim 是 vi improved 的缩写，可以认为 vim 是个程序开发工具，而不仅仅是一个正文编辑器。

在终端的命令行方式下，可以使用 vi 命令来启动 vi 编辑器以创建、修改或浏览一个或多个正文文件。vi 命令的语法格式如下。

```
vi  [选项] [文件名]
```

常用的选项是-r 和-R，使用-r 选项可以恢复退出编辑时没有保存的文件，语法格式如下。

```
vi  -r  文件名
```

-R 选项以只读方式打开文件，语法格式如下。

```
vi  -R  文件名
```

使用 vi 打开文件时，如果该文件已经存在，vi 将开启这个文件并显示该文件的内容；如果文件不存在，则在所编辑的内容第一次存盘时创建文件，并把已经编辑的内容保存进文件。

**例 3.31** 使用 vi 编辑器打开文件 mydream，运行界面如图 3.4 所示。

```
liuhui@liuhui-VirtualBox:~ $ vi mydream
```

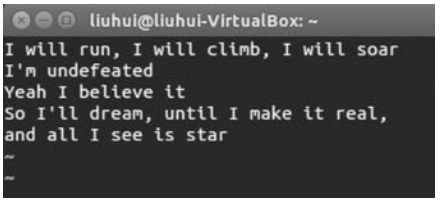



图 3.4 vi 编辑器的使用

打开 vi 编辑器以后，需要先按 Insert 键进入输入状态，然后才可以正式输入文字；需要删除一个字符时，需要先按 Delete 键，然后用 Delete 键或 Backspace 键删除一个字符；如果要删除一行，先按 Delete 键，然后连按两次 D 键即可把光标所在行删除。

要退出 vi 编辑器，先按 Esc 键，再输入冒号：，窗口最下面便会出现 ：提示符，最后输入字母 wq 或 q 并按 Enter 键，就可以退出 vi 编辑器了。

## 3.3 文件内容处理

### 3.3.1 内容浏览

#### 1. cat 命令

命令 cat(concatenate)可以实现如下所述 3 个功能。

(1) 第 1 个最常用的功能是在显示器上列出文件的内容，如果 cat 命令后面的文件是存在的，则该命令会以不停顿的只读方式显示整个文件的内容。

**例 3.32** 用 cat 命令显示正文内容。

```
liuhui@liuhui-VirtualBox:~$ cat mydream.txt
Dream it possible
互译歌词：
I will run, I will climb, I will soar.
我奔跑，我攀爬，我会飞翔。
I'm undefeated
永不言败
Jumping out of my skin, pull the chord
跳出我的皮肤，拨弄琴弦
Yeah I believe it
哦，我相信。
The past, is everything we were don't make us who we are
往昔，逝去的光阴不会决定现在
```

命令运行后，文件的内容就会直接显示在终端的窗口中，但只可以查看，不可以编辑。

(2) 第 2 个功能是从键盘创建一个新文件，命令格式如下。

```
cat  >  filename
```

按 Enter 键后，光标会停留在下一行开始处，等待用户在终端上从键盘输入字符。>为重定向符，表示把输入的内容输入文件中。输入完毕，按 Ctrl+D 快捷键退出编辑，输入的内容同时保存到新创建的文件中。

**例 3.33** 创建新文件 hlnew。

```
liuhui@liuhui-VirtualBox:~$ cat hlnew
cat: hlnew: No such file or directory
liuhui@liuhui-VirtualBox:~$ cat  >  hlnew
I will run I will climb I will soar
I'm underfeated
Jumping out of my skin pull the chord
Yeah I believe it
liuhui@liuhui-VirtualBox:~$
```

(3) 第 3 个功能是将几个文件合并为一个文件，命令格式如下。

```
cat   file1   file2  > file
```

**例 3.34** 用 cat 命令合并文件。

```
liuhui@liuhui-VirtualBox:~$ cat hlnew mydream >  newdream
liuhui@liuhui-VirtualBox:~$ cat newdream
I will run I will climb I will soar
I'm underfeated
Jumping out of my skin pull the chord
Yeah I believe it
This the second file mydream
I have a secret.
I wish I am a programmer, a Hacker!
I can steal others bank account and then
 get the bank card password.
And then, you know...
liuhui@liuhui-VirtualBox:~$
```

用 cat 命令实现合并文件，运行结果没有显示，再使用 cat 命令在终端上显示合并后的新文件的内容，可以发现确实把两个文件纵向合并成了一个文件。

cat 命令用于显示正文文件时可以正常显示，但是如果想显示二进制文件，系统会出现不稳定、终端突然死机的情况。

### 2. head 命令

head 命令默认显示文件的前 10 行，可以使用参数 -n 来指定显示的行数。

**例 3.35** 显示系统文件 passwd 的前 10 行。

```
liuhui@liuhui-VirtualBox:~$ head /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
liuhui@liuhui-VirtualBox:~$
```

**例 3.36** 指定只显示文件中前 5 行的内容。

```
liuhui@liuhui-VirtualBox:~$ head -n 5 /etc/passwd
```

或者

```
liuhui@liuhui-VirtualBox:~$ head -5 /etc/passwd
```

例 3.36 运行结果省略，可以参照例 3.37。

**例 3.37** 用--line 选项指定要显示的行数。

```
liuhui@liuhui-VirtualBox:~$ head --line 3 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
liuhui@liuhui-VirtualBox:~$
```

### 3. tail 命令

tail 命令可以显示文件最后几行的内容，默认显示最后 10 行的内容；也可以用 -n 和 +n 指定显示行数。-n 表示从文件末尾算起的 n 行；+n 表示从文件的第 n 行算起到文件结尾的内容。

**例 3.38** 显示最后 10 行。

```
liuhui@liuhui-VirtualBox:~$ tail deeplearning/mydream
From the bottom to the top
We're sparking wild fire's
Never quit and never stop
It's not until you fall that you fly
When your dreams come alive you're unstoppable
Take a shot chase the sun find the beautiful
We will glow in the dark turning dust to gold
And we'll dream it possible
Possible
And we'll dream it possible
liuhui@liuhui-VirtualBox:~$
```

**例 3.39** 用-n 选项设定显示最后两行。

```
liuhui@liuhui-VirtualBox:~$ tail -n 2 deeplearning/mydream
```

运行结果省略，参见例 3.41。

**例 3.40** 用--line 选项设定显示最后 3 行。

```
liuhui@liuhui-VirtualBox:~$ tail --line 3 deeplearning/mydream
```

**例 3.41** 用数字参数设定只显示最后两行。

```
liuhui@liuhui-VirtualBox:~$ tail -2 deeplearning/mydream
Possible
And we'll dream it possible
liuhui@liuhui-VirtualBox:~$
```

#### 4. more 命令

如果正文的内容很长，用前文介绍的命令浏览文件时不太方便，可以使用 more 命令将文件内容在屏幕上一次只显示一页，需要时可以上下翻页。在执行 more 命令时，每次在显示器上显示一页的内容，并在页面的底部出现“-more-(n%)”标记信息(n%表示已经显示的内容占总内容的百分比)，可以使用键盘上的如下按键进行操作。

(1) 空格键：向前(向下)移动一个屏幕。

(2) Enter 键：一次移动一行。

(3) B：往回(向上)移动一个屏幕。

(4) H：显示帮助菜单。

(5) /字符串：向前搜索这个字符串。

(6) N：发现这个字符串的下一个出现。

(7) Q：退出 more 命令并返回操作系统提示符下。

(8) V：在当前行启动/usr/bin/vi。

**例 3.42** 使用 more 命令查看长文件。

```
liuhui@liuhui-VirtualBox:~$ more hldata
```

more 命令运行结果与 cat 命令很相像，但 more 命令是一页一页显示，按空格键可以翻到下一页，按 B 键向前回退一页，按 Q 键退出显示。为了节省篇幅，这里省略了运行结果。

### 3.3.2 内容搜索

#### 1. grep 命令

grep 是 global/regular expressions/print 的缩写，grep 命令能够在一个或多个文件的内容中搜索某一个特定的字符模式(character pattern)，也称为正则表达式(regular expression)。一个模式可以是一个单一的字符、一个字符串、一个单词或一个句子。

一个正则表达式描述一组字符串的一个模式，正则表达式的构成模仿了数学表达式，通过使用操作符将较小的表达式组合成一个新的表达式。一个正则表达式既可以是一些纯文本文字，也可以是用来产生模式的一些特殊字符。grep 命令支持如下几种正则表达式的元字符(regular expression metacharacter)，即通配符。

(1) *：匹配 0 个(即空白)或多个字符。

(2) .：匹配任何一个字符，而且只能是一个字符。

(3) [xyz]：匹配方括号中的任意一个字符。

(4) [^xyz]：匹配不包括方括号中的字符的所有字符。

(5) ^：锁定行的开头。

(6) $：锁定行的结尾。

在基本正则表达式中，元字符 * 、+、{、|、(和 )失去了原义，如果要恢复其原义，要在之前冠以反斜线\，即\ * 、\+、\{、\|、\(、和\)，这类似于其他语言中的转义字符。

使用 grep 命令时，包含指定字符模式的每一行都会显示在显示器上，但是并不改变原有文件的内容。grep 命令的语法格式如下。

```
grep  选项  模式  文件名
```

grep 命令中常用选项如下。

(1) -c：仅列出包含模式的行数。

(2) -i：忽略模式中的字母大小写。

(3) -l：列出带有匹配行的文件名。

(4) -n：在每行的最前面列出行号。

(5) -v：列出没有匹配模式的行。

(6) -w：把表达式作为一个完整的单词来搜寻。

在用户 liuhui 的家目录下有一个 backup 子目录，里面有很多备份文件，包括 Linux 系统下书写的代码、Shell 程序、txt 文件等。

**例 3.43**　在 pythontrain 文件中搜索含有 Python 字样的内容，搜索模式是 Python。

```
liuhui@liuhui-VirtualBox:~/backup$ grep  Python pythontrain
Introduction to Python Programming
-  Introduction to Python Programming
-  Installing Python and pip
-  Scope of Python in Production
-  Features of Python
-  Python Versions
-  Python keywords and identifiers
-  Python Variables
```

例 3.43 运行结果显示了所有含有 Python 字样的文字所在的行。如果想统计共有多少行，可以使用选项 c。

**例 3.44**　统计满足条件的行数。

```
liuhui@liuhui-VirtualBox:~/backup$ grep -c  Python pythontrain
23
liuhui@liuhui-VirtualBox:~/backup$
```

Linux 系统是区分大小写的，如果希望在搜索时不区分大小写，可以使用 -i 选项忽略大小写。

**例 3.45**　在 pythontrain 文件中搜索含有 Python 字符的内容，忽略大小写。

```
liuhui@liuhui-VirtualBox:~/backup$ grep -i  Python pythontrain
```

例 3.45 运行结果省略。

**例 3.46**　在 pythontrain 文件中搜索以字符 In 开头的数据行。

```
liuhui@liuhui-VirtualBox:~/backup$ grep ^In   pythontrain
Introduction to Python Programming
liuhui@liuhui-VirtualBox:~/backup$
```

backup 子目录中有一个员工工资的数据文件 selary，下面搜索所有工资在 1000～1990 之间的数据行，在 grep 命令中，1..0 表示以 1 开头，以 0 结尾，中间有两个数字的字符串。

**例 3.47**　匹配模式为 1..0 的搜索。

```
liuhui@liuhui-VirtualBox:~/backup$  grep  '1..0'  selary
```

**例 3.48** 搜索首字母不是 1 或 2 的数据行。

```
liuhui@liuhui-VirtualBox:~/backup$ grep -v [12]… selary
```

```
liuhui@liuhui-VirtualBox:~/backup$ cat selary
Tom    1909  teacher
Mary   2387  professor
Rose   3400  professor
Charly 3214  professor
Sunny  1587  teacher
Stone  1690  teacher
Tony   3690  professor
Green  980   student
John   197   student

liuhui@liuhui-VirtualBox:~/backup$ grep '1..0' selary
Stone  1690  teacher
liuhui@liuhui-VirtualBox:~/backup$ grep -v [12]  selary
Rose   3400  professor
Tony   3690  professor
Green  980   student

liuhui@liuhui-VirtualBox:~/backup$
```

在 Linux 系统中,因为系统管理和维护的需要,经常需要用 grep 命令搜索系统配置信息。

**例 3.49** 进入/etc 目录查看/etc 目录下的相关信息。

```
liuhui@liuhui-VirtualBox:~/backup$ cd /etc
```

**例 3.50** 在当前目录下搜索 group、passwd、hosts 文件中含有 root 的文件名。

```
liuhui@liuhui-VirtualBox:etc$ grep -l root  group  passwd  hosts
```

**例 3.51** 用 ls 命令验证例 3.50 的运行结果。

```
liuhui@liuhui-VirtualBox:~/backup$ cd /etc
liuhui@liuhui-VirtualBox:/etc$ grep -l root group passwd hosts
liuhui@liuhui-VirtualBox:/etc$ ls -l  group passwd hosts
-rw-r--r-- 1 root root  941  9月 14 11:04 group
-rw-r--r-- 1 root root  232  8月 29 15:12 hosts
-rw-r--r-- 1 root root 2046  9月 14 11:04 passwd
liuhui@liuhui-VirtualBox:/etc$ _
```

**例 3.52** 用 head 命令列出 group 文件中前两行信息,以确认文件中确实含有 root 字符。

```
liuhui@liuhui-VirtualBox:etc$ head  -2 group
```

**例 3.53** 用 head 命令列出 passwd 文件中前两行信息,以确认文件中确实含有 root 字符。

```
liuhui@liuhui-VirtualBox:/etc$ head -2 group
root:x:0:
daemon:x:1:
liuhui@liuhui-VirtualBox:/etc$ head -2 passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
liuhui@liuhui-VirtualBox:/etc$
```

**例 3.54** 用 cat 命令列出 hosts 文件中的信息,以确认文件中确实不含有 root 字符。

```
liuhui@liuhui-VirtualBox:/etc$ cat hosts
127.0.0.1       localhost
127.0.1.1       liuhui-VirtualBox

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
liuhui@liuhui-VirtualBox:/etc$
```

例 3.54 运行结果显示，hosts 文件中确实不含有 root 字符，所以在例 3.50 的结果中没有 hosts 文件。

### 2. find 命令

使用 find 命令可以在文件路径中查找文件和目录，可以使用文件名、文件的大小、文件的属主、修改时间和类型等条件来进行搜索，当 find 命令找到了与搜索条件匹配的文件时，系统会将满足条件的每一个文件都显示在显示器上。find 命令的语法格式如下。

```
find pathnames  expressions  actions
```

(1) pathnames：find 命令所查找的目录路径。例如，用.来表示当前目录，用/来表示系统根目录。

(2) expressions：由一个或多个选项定义的搜寻条件。如果定义多个选项，find 命令将使用它们逻辑与(and)操作的结果，将列出满足所有条件的结果。

find 命令中常使用的条件表达式 expressions 如下。

- -name：按照文件名查找文件。文件名可以使用元字符，但是要放在双引号中。
- -size [+|-]n：查找文件大小大于或等于 +n(小于 -n)的文件。默认情况下，n 代表 512 字节大小的数据块的个数。
- -user：按照文件属主来查找文件。
- -group：按照文件所属的组来查找文件。
- -mtime -n +n：按照文件的更改时间来查找文件，-n 表示文件更改时间距现在 n 天以内，+n 表示文件更改时间距现在 n 天以前。

(3) actions：当文件被搜寻到以后需要进行的操作，默认将搜寻结果显示在显示器上。

在 find 命令中，可以使用如下的 actions(动作表达式)。

- -exec 命令{} \ ; ：find 命令对匹配的文件执行该参数所给出的 Shell 命令。相应命令的形式为'command {} \ ;'，注意{}和\ ; 之间必须有空格，{}之间无空格，\和; 之间无空格。花括号{} 表明文件名将传给前面表达式所表示的命令。{} 后面加空格，\ 和; 表示命令的结束。
- -ok 命令{} \ ; ：和-exec 的作用相同，只不过以一种更为安全的模式来执行该参数所给出的 Shell 命令，在执行每一个命令之前都会给出提示，让用户来确定是否继续执行。
- -print：find 命令将匹配的文件输出到标准输出。
- ls ：显示当前路径名和相关的统计信息。

**例 3.55** 在当前用户 liuhui 的家目录中，通过文件名查找文件内容含有 cat 字样的文件。

```
liuhui@liuhui-VirtualBox:~$ find ~  -name "*cat*"
/home/liuhui/backup2/tomcat1.txt.bz2
/home/liuhui/.local/share/applications
/home/liuhui/.cache/indicator-appmenu
```

例 3.55 运行结果较多，只截取了一部分。运行结果显示工作目录下含有 cat 字样的所有文件。

**例 3.56** 文件名的通配符的使用。

```
liuhui@liuhui-VirtualBox:~$ find  ~  -name  "*""cat""*"*
/home/liuhui/backup2/tomcat1.txt.bz2
/home/liuhui/.local/share/applications
/home/liuhui/.cache/indicator-appmenu
```

**例 3.57** 文件名的通配符的使用。

```
liuhui@liuhui-VirtualBox:~$ find  ~  -name  *"cat"*
/home/liuhui/backup2/tomcat1.txt.bz2
/home/liuhui/.local/share/applications
/home/liuhui/.cache/indicator-appmenu
```

为了节省篇幅,这里只截取了一部分运行结果。运行结果表明,3 种含有通配符的不同书写方法,对运行结果没有影响。但建议还是使用把所有文件名都括在双引号中的写法,因为有些系统可能运行结果不同。

假如用户 liuhui 的工作目录中有些文件已经过时,以后不用了,但又不能完整记住所有文件名,就可以用 find 命令先搜寻出这些文件,然后删除。

**例 3.58** 搜寻并删除工作目录下含有 huibian 字样的文件。

```
liuhui@liuhui-VirtualBox:~$ find . -name "*huibian*"
./huibian
./huibian2
./huibian1
liuhui@liuhui-VirtualBox:~$ find  .  -name "*huibian*" -exec rm {} \;
liuhui@liuhui-VirtualBox:~$
```

此命令须注意 rm {} \;{}和\;之间没有空格,其他地方有空格。

命令执行后系统无提示,可以用 ls 命令查看目录的内容来验证是否真正删除了。

**例 3.59** 验证例 3.58 是否真正删除了过时的文件 huibian。

```
liuhui@liuhui-VirtualBox:~$ find . -name "*huibian*"
liuhui@liuhui-VirtualBox:~$
```

运行结果表明确实删除了过时的文件。例 3.58 的删除操作在删除之前并没有给用户提示,让人觉得突兀,也不放心。在命令中加入-ok 动作表达式,可以强制系统在执行-ok 后面的动作时给出提示。

**例 3.60** 删除动作执行前给出提示。

```
liuhui@liuhui-VirtualBox:~$ find .  -name "*huibian*" -ok  rm {} \;
< rm ... ./huibian2 > ? n
< rm ... ./huibian1 > ? y
liuhui@liuhui-VirtualBox:~$ ls -l *huibian*
-rw-rw-r-- 1 liuhui liuhui 25  9月 19 21:16 huibian2
liuhui@liuhui-VirtualBox:~$
```

运行时根据提示输入 y,表示同意删除,输入 n 表示不同意删除。执行以后没有显示结果,用户可以再用 ls 命令查看验证。

**例 3.61** 在工作目录中查看过去 3 天内没有修改过的文件,即搜寻 3 天之前修改的文件。

```
liuhui@liuhui-VirtualBox:~$ find  .  -mtime +3
```

因为工作目录下修改时间大于 3 天的文件太多,为了节省篇幅,这里省略例 3.61 运行结果。

**例 3.62** 搜寻 backup 目录中 3 天之内修改过的文件。

```
liuhui@liuhui-VirtualBox:~$ find /home/liuhui/backup  -mtime -3
/home/liuhui/backup
/home/liuhui/backup/selary
/home/liuhui/backup/dladress
/home/liuhui/backup/dladress1
/home/liuhui/backup/linuxlearn
/home/liuhui/backup/jerrymos1
/home/liuhui/backup/dlsvm1
/home/liuhui/backup/tomcat1.txt
/home/liuhui/backup/selary~
/home/liuhui/backup/dlsvmcn
/home/liuhui/backup/Untitled Document 1~
/home/liuhui/backup/d1.txt
/home/liuhui/backup/dlreadme1
/home/liuhui/backup/tomcat1.txt~
/home/liuhui/backup/play.txt
/home/liuhui/backup/linuxdata
/home/liuhui/backup/pythontrain
/home/liuhui/backup/tcat2
liuhui@liuhui-VirtualBox:~$
```

**例 3.63** 搜寻 3 天之内被访问过的文件。

```
liuhui@liuhui-VirtualBox:~$ find /home/liuhui/backup  -atime -3
/home/liuhui/backup
/home/liuhui/backup/selary
/home/liuhui/backup/dladress
/home/liuhui/backup/dladress1
/home/liuhui/backup/linuxlearn
/home/liuhui/backup/jerrymos1
/home/liuhui/backup/dlsvm1
/home/liuhui/backup/tomcat1.txt
/home/liuhui/backup/selary~
/home/liuhui/backup/dlsvmcn
/home/liuhui/backup/Untitled Document 1~
/home/liuhui/backup/d1.txt
/home/liuhui/backup/dlreadme1
/home/liuhui/backup/tomcat1.txt~
/home/liuhui/backup/play.txt
/home/liuhui/backup/linuxdata
/home/liuhui/backup/pythontrain
/home/liuhui/backup/tcat2
liuhui@liuhui-VirtualBox:~$
```

例 3.63 运行结果没有显示文件的时间信息，可以利用 ls -l 命令查看文件的详细信息进行验证。

**例 3.64** 查看文件的时间信息。

```
liuhui@liuhui-VirtualBox:~$ ls -l /home/liuhui/backup
```

backup 目录下访问时间小于 3 天的文件太多，为了节省篇幅，这里省略了例 3.64 运行结果，运行结果可以与例 3.63 做对照，可以看出例 3.63 中显示了文件的访问时间确实是 3 天之内的。

### 3.3.3 内容统计

使用命令 wc(word count)可以统计一个文件中内容的行数、单词数和字符数，语法格式如下。

```
wc [options] file-list
```

其中，options 是选项，常用选项如下。

(1) -c：统计文件字节数。

(2) -m：统计文件字符数。

(3) -l：统计文件行数。

(4) -L：统计文件最长行的长度。

(5) -w：统计文件单词数。

如果命令中没有使用任何选项，则将文件中的行数、单词数和字符数全部显示出来。

**例 3.65** 统计工作目录下 hldata1 文件内容的行数、单词数和字符数。

```
liuhui@liuhui-VirtualBox:~$ wc hldata1
 12  36 237 hldata1
liuhui@liuhui-VirtualBox:~$
```

例 3.65 运行结果按行数、单词数和字符数的顺序显示了各项的大小。

前面学习过诸如 who、whoami、users 等获取用户信息的命令，如果想统计本机上有几个用户，就可以使用 wc 命令的统计功能。因为在/etc/passwd 文件中，每个用户都有且只有一行记录，所以/etc/passwd 文件内容的行数就是用户的个数，使用-l 选项可以只获取数据的行数，即用户数。

**例 3.66** 统计用户数。

```
liuhui@liuhui-VirtualBox:~$ wc -l /etc/passwd
39 /etc/passwd
liuhui@liuhui-VirtualBox:~$
```

例 3.66 运行结果显示了当前有 39 行数据，即有 39 个用户。用户可以用 cat 命令直接查看 passwd 文件，也可以在图形界面中直接打开/etc/passwd 文件进行查看，可见里面确实有 39 个用户，系统创建的用户在前面记录，安装系统时创建的普通用户和用 adduser 命令增加的用户在文件的最后几行记录。

**例 3.67** 统计联机字典中单词的个数。

```
liuhui@liuhui-VirtualBox:~$ wc -l /usr/share/dict/words
99171 /usr/share/dict/words
liuhui@liuhui-VirtualBox:~$
```

Linux 的联机字典存储在/usr/share/dict/words 文件中，每个单词占据一行，文件的行数就是单词的个数。

## 3.3.4 内容比较

### 1. diff 命令

diff 是单词 different 的缩写，用来比较两个文件内容的差别，运行结果的第 1 行以<开始，显示第 1 个文件的内容。第 2 行以>开始，显示第 2 个文件的内容。diff 命令经常用在软件升级时，对比新的配置文件和旧的配置文件之间的不同。

diff 命令是以逐行的方式比较文本文件内容的异同之处，如果指定比较的是目录，diff 命令会比较两个目录下名字相同的文本文件的内容，但不会比较其中的子目录。

在用户 liuhui 的家目录下有两个文件 mydream 和 newdream，先查看这两个文件的内容。

**例 3.68** 查看 mydream 文件的内容。

```
liuhui@liuhui-VirtualBox:~$ cat mydream
dream it possible
I will run I will climb I will soar
I'm undefeated
Jumping out of my skin pull the chord
Yeah I believe it
The past is everything we were don't make us who we are
So I'll dream until I make it real and all I see is stars
It's not until you fall that you fly
When your dreams come alive you're unstoppable
Take a shot chase the sun find the beautiful
We will glow in the dark turning dust to gold
And we'll dream it possible
Possible
And we'll dream it possible
I will chase I will reach I will fly
Until I'm breaking until I'm breaking
```

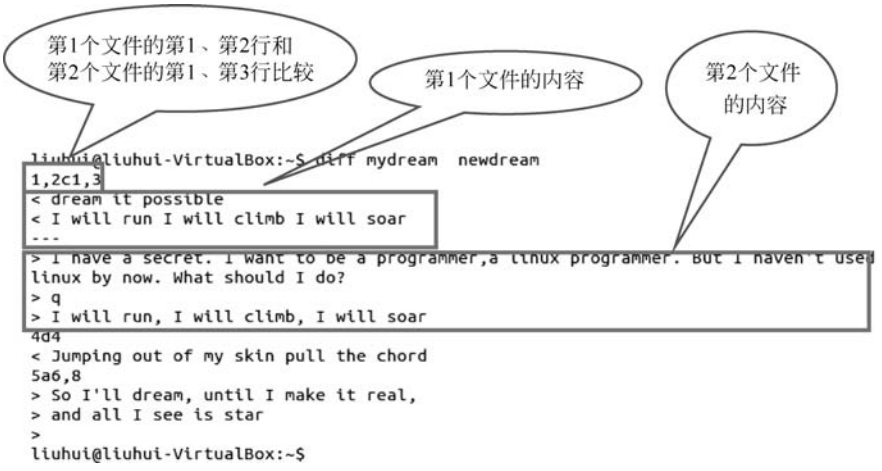
**例 3.69** 查看 newdream 文件的内容。

```
liuhui@liuhui-VirtualBox:~$ cat newdream
I have a secret. I want to be a programmer,a linux programmer.
But I haven't used linux by now. What should I do?
q
I will run, I will climb, I will soar
I'm undefeated
Yeah I believe it
So I'll dream, until I make it real,
and all I see is star

The past is everything we were don't make us who we are
So I'll dream until I make it real and all I see is stars
It's not until you fall that you fly
```

两个文件比较长，后面部分都是相同的，故例 3.68 和例 3.69 运行结果省略了后面部分的显示。

**例 3.70** 用 diff 命令比较这两个文件的不同。




### 2. sdiff 命令

sdiff(Side-by-Side Difference)命令会把比较的结果显示出来，符号的左侧显示第 1 个文件的内容，|符号的右侧显示第 2 个文件的内容；如果第 1 个文件还有内容而第 2 个文件没有内容了，则用符号<表示；如果第 2 个文件还有内容而第 1 个文件没有内容了，用符号>表示。因为比较的文件每行内容较长，所以第 2 个文件的内容另起一页显示了，如例 3.71 所示。

**例 3.71** 用 sdiff 命令比较文件。

```
liuhui@liuhui-VirtualBox:~$ sdiff mydream  newdream
dream it possible                                               |
I have a secret. I want to be a programmer,a linux programmer
I will run I will climb I will soar                             |
q
                                                                >

I will run, I will climb, I will soar
I'm undefeated                                                  I
'm undefeated
Jumping out of my skin pull the chord                           <
Yeah I believe it                                               Y
eah I believe it
                                                                >

So I'll dream, until I make it real,
                                                                >

and all I see is star
                                                                >
The past is everything we were don't make us who we are         T
he past is everything we were don't make us who we are
So I'll dream until I make it real and all I see is stars       S
```

从运行结果来看，sdiff 命令的显示结果比 diff 命令的结果更清晰简单，但是如果两个文件内容很多，而且差别又很少，sdiff 命令的运行结果就不容易阅读，还是使用 diff 命令更好一些。

## 3.3.5 内容转换

### 1. Tab 转换为空格

expand 命令用于将文件的制表符(Tab)转换为空格符(Space)，默认一个 Tab 对应 8 个空格符，并将结果输出到标准输出。若不指定任何文件名或所给文件名为 -，则 expand 命令会从标准输入读取数据。用户也可以使用输入输出重定向符指定读入和输出的文件名。

expand 命令语法格式如下。

```
expand [options][file]
```

常用选项如下。

(1) -i,--initial：不转换非空白符后的制表符。

(2) -t,--tabs=NUMBER：指定一个 Tab 替换为多少个空格，而不是默认的 8。

例如，在 liuhui 用户的家目录中有一个文件 windata，文中的分隔符是 Tab 键，将 Tab 转换为空格，首先要在命令中使用-A 选项查看该文件。

**例 3.72** 使用-A 选项查看文件中的分隔符类型。

```
liuhui@liuhui-VirtualBox:~$ cat -A windate
cat: windate: No such file or directory
liuhui@liuhui-VirtualBox:~$ cat -A windata
This^Iis^Ia^Ifile^Iedited^Iin^Iwindows7.$
The^Iseparator^Iis^I"Tab".$
We^Iwash^Iconvert^ITab^Ito^Ispace.$
Let's^Itry^Iit!$
liuhui@liuhui-VirtualBox:~$
```

例 3.72 运行结果中的^I 就表示分隔符是 Tab 键，行尾的 $ 表示是以 Enter 键换行的。下面用 expand 命令把 Tab 分隔符转换为空格符，同时把转换以后的内容重定向输出到文件 windata.spaces 中。

**例 3.73** 把 Tab 分隔符转换为空格符。

```
liuhui@liuhui-VirtualBox:~$ expand windata  >  windata.spaces
```

例 3.73 运行结果没有显示，可以使用 cat 命令查看转换以后的文件内容。

**例 3.74** 显示分隔符类型。

```
liuhui@liuhui-VirtualBox:~$ expand windata > windata.spaces
liuhui@liuhui-VirtualBox:~$ cat -A windata.spaces
This    is      a       file    edited  in      windows7.$
The     separator       is      "Tab".$
We      wash    convert Tab     to      space.$
Let's   try     it!$
liuhui@liuhui-VirtualBox:~$
```

例 3.74 运行结果的分隔符看不到特殊符号，表明是空格分隔符。

### 2. 文件内容的格式化转换

fmt 命令可以将文件格式化成段落，就是定义一行文字的格式，其语法形式如下。

```
fmt [-WIDTH] [OPTION]... [FILE]...
```

每行的宽度使用 wn 选项来定义，w 是 width 的第 1 个字母，n 是字符的数目，系统默认是 75 个字符，也可以用-u 选项将文件中的空格统一化，即每个单词之间使用一个空格分隔，每个句子使用两个空格分隔。

先对例 3.72 中的文件 windata.spaces 做一些处理，在单词之间无规律地添加不同数目的空格，然后统一空格字符的个数。

**例 3.75** 统一空格字符的个数，每行 48 个字符。

```
liuhui@liuhui-VirtualBox:~$ fmt -u -w48 windata.spaces > windata.fmt
```

例 3.75 运行结果没有显示，再次使用 cat -A 命令查看文件内容。

**例 3.76** 查看统一以后的文件。

```
liuhui@liuhui-VirtualBox:~$ fmt -u -w48 windata.spaces > windata.fmt
liuhui@liuhui-VirtualBox:~$ cat -A windata.fmt
This is a file edited in windows7.  The$
separator is "Tab".  We wash convert Tab$
to space.  Let's try it!$
liuhui@liuhui-VirtualBox:~$
```

仔细观察例 3.76 的运行结果，可以看出每个单词之间都由一个空格分隔，多余的空格去掉了；每个句子以两个空格分开，段落以 $ 作为结束符。

## 3.3.6 文件归档、压缩及解压缩

### 1. tar 命令

为了保证文件的安全，经常要对文件进行备份和归档。Linux 系统的标准归档命令是 tar，tar 命令的功能是将多个文件(包括目录)放在一起存放到一个磁带或磁盘归档文件中，并且将来可以根据需要只还原归档文件中某些指定的文件。tar 命令默认不进行文件的压缩，但 tar 命令本身支持压缩和解压缩算法，内部的压缩和解压缩算法是 gzip 和 gunzip 或 bzip2 和 bunzip2。

tar 命令的语法格式如下。

```
tar [参数] 文件名
```

**注意**

> 归档后的文件名要使用相对路径。在 tar 命令中必须至少使用如下选项中的一个。

(1) c：创建一个新的归档文件。
(2) t：列出归档文件中内容的目录。
(3) x：从归档文件中抽取文件。
(4) f：指定归档文件或磁带。

tar 命令中还有以下 3 个可选的选项。

(1) v：显示所打包的文件的详细信息。
(2) z：使用 gzip 压缩算法来压缩打包后的文件。
(3) j：使用 bzip2 压缩算法来压缩打包后的文件。

**注意**

> tar 命令中所有选项之前都不能使用前导符号-。

普通用户 liuhui 登录到默认的家目录，家目录中有一个子目录 Pictures，含有若干图片文件，现在把这个子目录归档成 pictures.tar 文件，使用 c 选项创建一个新的归档文件，使用 v 选项在创建的过程中显示所有打包的文件和目录，使用 f 选项指定归档的文件名为 pictures.tar。

**例 3.77** 归档新文件。

```
liuhui@liuhui-VirtualBox:~$ tar cvf pictures.tar Pictures
Pictures/
Pictures/pic9.jpg
Pictures/pic5.jpg
Pictures/pic1.jpeg
Pictures/pic6.jpg
Pictures/pic7.jpg
Pictures/pic8.jpg
Pictures/pic4.jpg
Pictures/pic2.jpg
Pictures/pic3.jpg
Pictures/Screenshot from 2019-09-12 14:36:57.png
liuhui@liuhui-VirtualBox:~$
```

例 3.77 运行结果显示了归档后新文件中包含的所有文件和目录，最后一个文件是个截图文件，使用的默认文件名比较长。也可以使用 tar 命令显示 pictures.tar 中包含的所有文件。

**例 3.78** 用 t 选项列出归档文件包含的所有文件。

```
liuhui@liuhui-VirtualBox:~$ tar tvf pictures.tar
drwxr-xr-x liuhui/liuhui     0 2019-09-22 11:04 Pictures/
-rw-rw-r-- liuhui/liuhui 19251 2019-09-22 11:02 Pictures/pic9.jpg
-rw-rw-r-- liuhui/liuhui 29262 2019-09-22 11:02 Pictures/pic5.jpg
-rw-rw-r-- liuhui/liuhui 21813 2019-09-22 11:01 Pictures/pic1.jpeg
-rw-rw-r-- liuhui/liuhui 20394 2019-09-22 11:03 Pictures/pic6.jpg
-rw-rw-r-- liuhui/liuhui 15352 2019-09-22 11:03 Pictures/pic7.jpg
-rw-rw-r-- liuhui/liuhui 21541 2019-09-22 11:04 Pictures/pic8.jpg
-rw-rw-r-- liuhui/liuhui 26702 2019-09-22 11:02 Pictures/pic4.jpg
-rw-rw-r-- liuhui/liuhui 29734 2019-09-22 11:02 Pictures/pic2.jpg
-rw-rw-r-- liuhui/liuhui 23132 2019-09-22 11:02 Pictures/pic3.jpg
-rw-rw-r-- liuhui/liuhui 15937 2019-09-12 14:37 Pictures/Screenshot
from 2019-09-12 14:36:57.png
liuhui@liuhui-VirtualBox:~$
```

### 2. 恢复文件

接下来再把归档打包的文件解开，并恢复到原来的位置，使用 tar 命令的 xvf 选项在工作目录中抽取打包的文件，所以需要将工作目录切换到打包时所在的目录。为了更加清楚地演示打包文件的恢复过程，先把原来 Pictures 子目录的所有内容全部删除。

**例 3.79** 使用 rm -r 命令删除目录和目录中的所有内容。

```
liuhui@liuhui-VirtualBox:~$ rm -r Pictures
```

例 3.79 运行结果没有显示，可以使用 ls 命令验证查看。

**例 3.80** 验证工作目录中没有 Pictures 子目录。

```
liuhui@liuhui-VirtualBox:~$ ls -l pic*
-rw-rw-r-- 1 liuhui liuhui 235520  9月 22 11:08 pictures.tar
liuhui@liuhui-VirtualBox:~$
```

例 3.80 运行结果表明目录中只有归档文件 pictures.tar，也可以在图形界面下直接查看有无目录 Pictures。

**例 3.81** 恢复归档时被删除的原始文件。

```
liuhui@liuhui-VirtualBox:~$ tar xvf pictures.tar
Pictures/
Pictures/pic9.jpg
Pictures/pic5.jpg
Pictures/pic1.jpeg
Pictures/pic6.jpg
Pictures/pic7.jpg
Pictures/pic8.jpg
Pictures/pic4.jpg
Pictures/pic2.jpg
Pictures/pic3.jpg
Pictures/Screenshot from 2019-09-12 14:36:57.png
liuhui@liuhui-VirtualBox:~$
```

例 3.81 运行结果表明被删除的文件通过事先归档的文件又被恢复到了工作目录下。

### 3. gzip 命令

在 Linux 系统中有两组常用的压缩命令，第 1 组是 gzip 和 gunzip，第 2 组是 bzip2 和 bunzip2。gzip 对正文文件的压缩比一般超过 75%，通常 bzip2 对归档文件的压缩比要优于 gzip，比较新的 Linux 版本才支持 bzip2 和 bunzip2。

gzip 命令可以用来压缩文件，压缩后的结果会存在一个文件中，使用原来的文件名加上 .gz 作为扩展名，压缩文件会保留原文件的访问及修改时间、所有权和访问权限，原文件将会从文件结构中删除。gzip 命令只能对文件进行压缩，对目录不能压缩。

gzip 命令的语法格式如下。

```
gzip [选项] [压缩文件名…]
```

gzip 命令的几个经常使用的选项如下。

- -v：在屏幕上显示文件的压缩比。
- -c：保留原文件并新创建一个压缩文件。

1）压缩、解压缩普通文件

普通用户 liuhui 的家目录/home/liuhui 下有一个目录 hlprivacy1，该目录下有 Linux 系

统下编辑的多个文件和一个目录 linuxdata，用 gzip 命令来压缩文件，压缩之前先查看文件的详细信息。

**例 3.82** 查看要压缩的文件。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ ls -l
total 24
-rw-rw-r-- 1 liuhui liuhui  121  9月 17 11:15 d1.txt
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 10:31 d1.txt~
-rw-rw-r-- 1 liuhui liuhui    0  9月 17 10:40 jerrymos1
drwxrwxr-x 2 liuhui liuhui 4096  9月 22 13:02 linuxdata
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 11:18 play.txt
-rw-rw-r-- 1 liuhui liuhui   78  9月 17 10:34 tomcat1~
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 11:29 tomcat1.txt
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

**例 3.83** 压缩文件并查看已经压缩的文件。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ gzip d1.txt
liuhui@liuhui-VirtualBox:~/hlprivacy1$ ls -l d*
-rw-rw-r-- 1 liuhui liuhui 116  9月 17 10:31 d1.txt~
-rw-rw-r-- 1 liuhui liuhui 114  9月 17 11:15 d1.txt.gz
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

例 3.83 运行结果表明，目录下确实生成了一个名为 d1.txt.gz 的压缩文件，但是原有的 d1.txt 文件不见了，文件的大小也发生了变化，且变小了，证明了 gzip 确实将文件进行了压缩。该目录下还有一个 d1.txt～文件，这表明在 hlprivacy1 目录下曾经对这个文件 d1.txt 做过修改，或者刚刚删除了这个文件，也就是“～”代表草稿文件，现在已经保存完毕了，可以不必理会它。另外要注意，在 Linux 中，文件的后缀不表示类型，后缀是文件名的一部分，所以在输入时要连后缀一起输入作为文件名。

利用 gunzip 命令可以将原文件恢复。

**例 3.84** 解压文件。

```
liuhui@liuhui-VirtualBox:~$ gunzip d1.txt.gz
```

例 3.84 运行结果没有显示，可以用 ls 命令查看验证。

**例 3.85** 查看解压后的文件。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ gunzip d1.txt.gz
liuhui@liuhui-VirtualBox:~/hlprivacy1$ ls -l
total 24
-rw-rw-r-- 1 liuhui liuhui  121  9月 17 11:15 d1.txt
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 10:31 d1.txt~
-rw-rw-r-- 1 liuhui liuhui    0  9月 17 10:40 jerrymos1
drwxrwxr-x 2 liuhui liuhui 4096  9月 22 13:02 linuxdata
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 11:18 play.txt
-rw-rw-r-- 1 liuhui liuhui   78  9月 17 10:34 tomcat1~
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 11:29 tomcat1.txt
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

例 3.85 运行结果表明，被压缩的文件 d1.txt.gz 已经被恢复成原文件 d1.txt 了，与例 3.82 的运行结果比较可以发现恢复的文件和原来的一模一样。同时，压缩形成的文件 d1.txt.gz 消失了。能否在压缩文件的同时保留原始文件呢？

**例 3.86** 压缩文件时保留原文件 play.txt。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ gzip  -cv play.txt
play.txt:       k◆]play.txtMOAn◆@
                                  ◆◆◆◆\Qs◆◆sB◆zU◆◆◆◆yC^◆◆)_◆◆P◆<◆◆X◆c◆d\
n◆PE    BK◆&◆<&◆iU◆j◆◆4◆M?◆3◆◆B◆q1S◆◆$◆◆F◆!4
                                          ◆DU◆83◆z◆◆6(c  Q@◆◆_◆◆◆◆0◆7]◆]
4◆◆◆◆◆◆◆N)◆◆*{◆◆◆◆      ◆◆i◆C◆◆◆7#◆◆     ◆`◆◆◆◆y◆(◆◆◆◆
◆31.2%
liuhui@liuhui-VirtualBox:~/hlprivacy1$ gzip  -cv play.txt > play.txt.gz
play.txt:        31.2%
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

例 3.86 运行过程中会显示压缩比，这是由于使用了-v 选项；运行结果表明，需要用重定向符指定压缩后的文件，否则不能正常压缩，且有乱码提示。

**例 3.87**　验证压缩操作后是否保留了原文件。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ ls -l pl*
-rw-rw-r-- 1 liuhui liuhui 266  9月 22 13:23 play.txt
-rw-rw-r-- 1 liuhui liuhui 116  9月 17 11:18 play.txt~
-rw-rw-r-- 1 liuhui liuhui 210  9月 22 13:39 play.txt.gz
liuhui@liuhui-VirtualBox:~/hlprivacy1$ _
```

例 3.87 运行结果可见，工作目录下除了原文件 play.txt 以外，还有一个压缩文件 play.txt.gz。

2）压缩图像文件

对于普通文件，经过压缩后文件变小了很多，压缩比可达 32%；对于图像等以二进制形式保存的文件同样也可以压缩。

**例 3.88**　压缩图像文件。

```
liuhui@liuhui-VirtualBox:~/Pictures$ gzip -cv pic1 > pic1.gz
gzip: pic1: No such file or directory
liuhui@liuhui-VirtualBox:~/Pictures$ gzip -cv pic1.jpeg  > pic1.gz
pic1.jpeg:        0.9%
liuhui@liuhui-VirtualBox:~/Pictures$ _
```

压缩图像文件的运行过程比较慢，结果表明只压缩了 0.9%。对于图像文件，一般不执行压缩操作，因为图像文件的数据格式，不管是 jpg、bmp，还是 png、gif，本身就已采用了压缩格式。

3）压缩目录

在文件传送中，有时一次需要传送多个文件，在 Windows 系统中可以把多个文件放在一个文件夹中，压缩后变为一个文件再传送，方便双方收发。在 Linux 系统中是否也可以对目录进行压缩呢？

普通用户 liuhui 的家目录/home/liuhui 下有一个目录 hlprivacy1，该目录下有 Linux 系统下编辑的多个文件和一个目录 linuxdata，里面有多个文件，对这个目录执行压缩操作。

**例 3.89**　压缩目录。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ gzip linuxdata
gzip: linuxdata is a directory -- ignored
liuhui@liuhui-VirtualBox:~/hlprivacy1$ _
```

例 3.89 运行结果提示不能压缩目录。

### 4. bzip2 命令

bzip2 采用新的压缩算法，压缩效果比 gzip 压缩算法更好。若没有加上任何参数，bzip2 命令压缩完文件后会产生后缀为.bz2 的压缩文件，并删除原始的文件。bzip 和 gzip 一样，都不支持压缩目录。其语法格式如下。

```
bzip2  [参数]  [文件名]
```

常用参数如下。

（1）-r：查找指定目录并压缩或解压缩其中的所有文件。

（2）-k：压缩并保留原文件。

（3）-c 或-stdout：将压缩与解压缩的结果送到标准输出。

(4) -d 或-decompress：执行解压缩。

(5) -f 或-force：bzip2 在压缩或解压缩时，若输出文件与现有文件同名且没有参数，默认不会覆盖现有文件，使用参数-f 可以强制覆盖原文件。

普通用户 liuhui 登录到目录/home/liuhui/hlprivacy1，该目录下有 Linux 系统下编辑的文件 tomcat1.txt，用 bzip2 命令来压缩该文件之前先查看文件的详细信息。

**例 3.90** 查看要压缩的文件。

```
liuhui@liuhui-VirtualBox:~$ cd hlprivacy1
liuhui@liuhui-VirtualBox:~/hlprivacy1$ ls -l
total 36
-rw-rw-r-- 1 liuhui liuhui  121  9月 17 11:15 d1.txt
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 10:31 d1.txt~
-rw-rw-r-- 1 liuhui liuhui    0  9月 17 10:40 jerrymos1
drwxrwxr-x 2 liuhui liuhui 4096  9月 22 13:02 linuxdata
-rw-rw-r-- 1 liuhui liuhui  266  9月 22 13:23 play.txt
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 11:18 play.txt~
-rw-rw-r-- 1 liuhui liuhui  210  9月 22 13:39 play.txt.gz
-rw-rw-r-- 1 liuhui liuhui   78  9月 17 10:34 tomcat1~
-rw-rw-r-- 1 liuhui liuhui  116  9月 17 11:29 tomcat1.txt
-rw-rw-r-- 1 liuhui liuhui  127  9月 22 13:40 tomcat.gz
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

**例 3.91** 用 bzip2 命令压缩文件。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ bzip2 tomcat1.txt
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

例 3.91 运行结果没有显示，可以使用 ls 命令来查看，也可以使用图形界面，在桌面上双击 Files 图标直接查看文件，发现 tomcat1.txt 文件没有了，出现了 tomcat1.txt.bz2 文件，如图 3.5 所示。

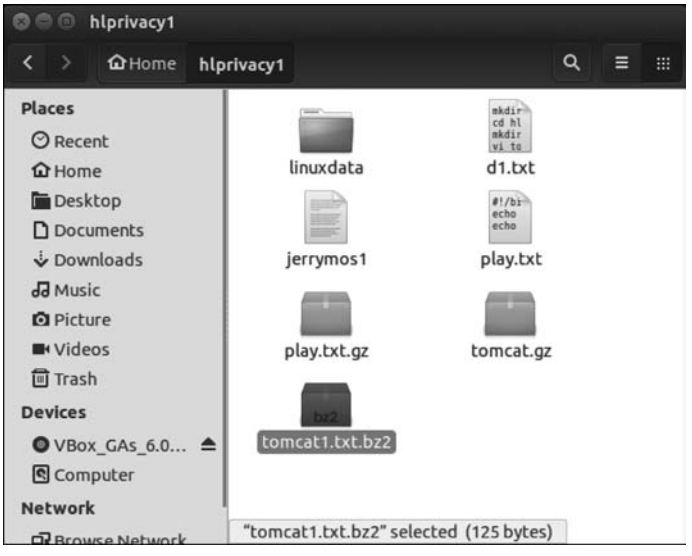



图 3.5 bzip2 命令的执行结果

**例 3.92** 解压例 3.91 中压缩的文件。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ bzip2 -d tomcat1.txt.bz2
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

例 3.92 运行结果没有显示，可以使用 ls 命令来查看，也可以使用图形界面，在桌面上双击 Files 图标直接查看文件，发现 tomcat1.txt.bz2 文件没有了，出现了 tomcat1.txt.。

**例 3.93**　压缩文件时保留原文件。

```
liuhui@liuhui-VirtualBox:~$ bzip2 -k  jerrymos1
```

例 3.93 运行结果没有显示，可以用 ls 命令验证查看。

**例 3.94**　查看原文件是否被保留。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ bzip2 -k jerrymos1
liuhui@liuhui-VirtualBox:~/hlprivacy1$ ls -l jer*
-rw-rw-r-- 1 liuhui liuhui  0  9月 17 10:40 jerrymos1
-rw-rw-r-- 1 liuhui liuhui 14  9月 17 10:40 jerrymos1.bz2
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

例 3.94 运行结果表明原文件被保留了。

## 3.4　文件输入输出

在系统默认情况下，Shell 从键盘读命令的输入，并将命令的输出显示到显示器上。但在命令或者 Shell 脚本中，使用输入输出重定向符可以改变实际操作时读入数据和显示数据的方式。

### 3.4.1　文件描述符

文件描述符是 Linux 系统内部使用的一个文件代号，它决定从哪里读入命令所需的输入和将命令产生的输出及错误显示送到哪里。

文件描述符有 0、1 和 2，共 3 个编码，具体含义如下。

(1) 0：标准输入，文件描述符的缩写为 stdin。

(2) 1：标准输出，文件描述符的缩写为 stdout。

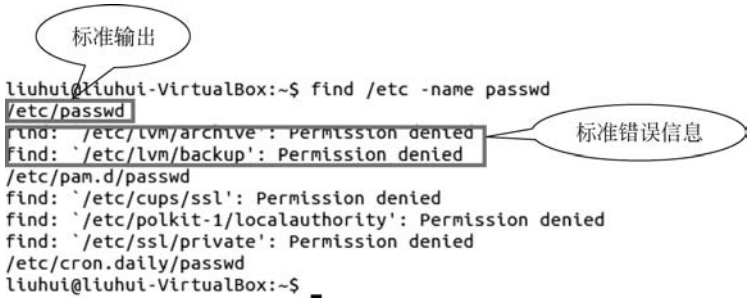
(3) 2：标准错误(信息)，文件描述符的缩写为 stderr。

这些号码存储在/dev/std * 系统文件中，可以用 ls 命令来查看。

**例 3.95**　查看文件描述符。

```
liuhui@liuhui-VirtualBox:~/hlprivacy1$ ls -l /dev/std*
lrwxrwxrwx 1 root root 15  9月 23 19:28 /dev/stderr -> /proc/self/fd/2
lrwxrwxrwx 1 root root 15  9月 23 19:28 /dev/stdin -> /proc/self/fd/0
lrwxrwxrwx 1 root root 15  9月 23 19:28 /dev/stdout -> /proc/self/fd/1
liuhui@liuhui-VirtualBox:~/hlprivacy1$
```

**例 3.96**　标准输出和标准错误信息。



## 3.4.2 输入输出重定向和转换

### 1. 输出重定向

>符号为输出重定向符号；>>也是输出重定向符号，二者的区别如下所述。

（1）>：覆盖原文件的内容，每次都删除原有文件的内容，从文件的开头写入数据。

（2）>>：在原文件之后追加内容，原文件的内容保留，新增加的内容紧跟在原文件内容的后面。

输出重定向命令语法如下。

```
command  >  filename
command  >>  filename
```

功能说明：把 command 命令的运行结果写入 filename 文件中，而不是把命令的执行结果显示在显示器上。

**例 3.97** 从/etc 目录开始搜寻名为 passwd 的文件，在屏幕上只显示标准错误信息，将标准输出重定向到一个名为 output.std 的文件中。

```
liuhui@liuhui-VirtualBox:~$ find /etc -name passwd 1>output.std
find: `/etc/lvm/archive': Permission denied
find: `/etc/lvm/backup': Permission denied
find: `/etc/cups/ssl': Permission denied
find: `/etc/polkit-1/localauthority': Permission denied
find: `/etc/ssl/private': Permission denied
liuhui@liuhui-VirtualBox:~$ su -
Password:
root@liuhui-VirtualBox:~# find /etc -name passwd 1>output.std
root@liuhui-VirtualBox:~# cat output.std
/etc/passwd
/etc/pam.d/passwd
/etc/cron.daily/passwd
root@liuhui-VirtualBox:~#
```

例 3.97 运行结果表明，第 1 次在普通用户 liuhui 的家目录下执行查询/etc 目录下的文件的 find 命令时，系统提示无权操作的错误信息，因为命令中只规定把标准输出重定向到 output.std 文件中，没有定义系统错误信息的输出，所以使用默认操作输出到显示器上。更换为 root 用户后，再执行 find /etc 命令，显示器上没有任何运行结果；于是使用 cat 命令查看重定向的文件 output.std，可以看到查找到的 passwd 文件在/etc 目录下有 3 处。

**例 3.98** 将 find 命令的标准输出写入 output.std 文件，将标准错误信息写入 err.std 文件。

```
root@liuhui-VirtualBox:~# find /etc -name passwd 1>output.std 2>>  err.std
root@liuhui-VirtualBox:~# cat output.std
/etc/passwd
/etc/pam.d/passwd
/etc/cron.daily/passwd
root@liuhui-VirtualBox:~# cat err.std
```

### 2. 输入重定向

使用<符号可以从文件中读取数据到命令中，断开键盘和“命令”的标准输入之间的关联，然后将输入文件关联到标准输入。

输入重定向命令语法如下。

```
command < filename
```

功能说明：command 命令的输入来自 filename 文件。

**例 3.99** 从文件中读取数据存入变量中。

```
liuhui@liuhui-VirtualBox:~$ cat inpt4
liuhui users
shiephl users
tomcat1 users

liuhui@liuhui-VirtualBox:~$ read var1  <  inpt4
liuhui@liuhui-VirtualBox:~$ echo $var1
liuhui users
liuhui@liuhui-VirtualBox:~$
```

例 3.99 中，从 inpt4 文件中读取一行数据存入变量 var1，然后用 echo 输出变量的值加以验证。

### 3. 转换命令

输入重定向经常用于对文件格式的更改，例如从 Windows 系统中传入的文件，以回车符\r 和换行符\n 结束一行；而在 Linux 系统中，使用\n 结束一行。从 Windows 系统导入文件到 Linux 中时，不可能手工修改结束符号，于是 Linux 提供了 tr(translate)命令，可以转换、压缩和删除来自标准输入的字符，并将结果写到标准输出设备上。tr 命令的输入不接受文件名形式的参数，即不能用文件名作为它的输入参数，只能以输入重定向<或者管道|来指定输入的数据来源。

tr 命令的语法如下。

```
tr OPTION… SET1 [SET2]
```

一个 SET 就是一串字符，包括特殊的反斜杠转义字符。

tr 命令常用选项如下。

(1) -d 或--delete：删除所有属于第 1 个字符串 SET1 的字符。

(2) -s 或--squeeze-repeats：用第 2 个字符串 SET2 代替第 1 个字符串 SET1，或者把连续重复的字符以单独一个字符表示。

(3) -t 或--truncate-set1：删除第 1 个字符串 SET1 较第 2 个字符串 SET2 多出的字符。

tr 命令通常与管道或其他命令结合使用，可以执行删除重复字符、将大写转换为小写以及替换和删除基本字符等操作。

**例 3.100** 查看从 Windows 系统传来的文件 possible.txt 中的结束符。

```
liuhui@liuhui-VirtualBox:~$ cat -A possible.txt
dream it possible^M$
I will run I will soar^M$
I'm undefeated^M$
Jumping out of my skin pull the chord^M$
Yeah I believe it^M$
The past is everything^M$
So I'll dream ^M$
It's not until you fall that_you flyliuhui@liuhui-VirtualBox:~$
```

例 3.100 运行结果表明，Windows 系统下的文件以\r 和\n 作为结束符，结果以^M $ 标注，并且最后一行没有换行符。Linux 系统下的结束符只有\n 一个。

**例 3.101** 更改结束符并存为新文件。

```
liuhui@liuhui-VirtualBox:~$ tr -d '\r'  < possible.txt  > poss3.lnx
liuhui@liuhui-VirtualBox:~$ cat -A  poss3.lnx
dream it possible$
```

```
I will run I will soar$
I'm undefeated$
Jumping out of my skin pull the chord$
Yeah I believe it$
The past is everything$
So I'll dream $
It's not until you fall that you flyliuhui@liuhui-VirtualBox:~$
```

例 3.101 中，从 possible.txt 文件读取数据，利用 tr -d 命令删除\r 字符，把删除\r 字符以后的文件以新文件保存到 poss3.lnx 文件中，然后再查看结束符，确实变成了 $，即只有\n 作为结束符。

删除\r 字符，还可以以 tr -s 命令来实现，即把\r\n 替换为\n，如例 3.102 所示。

**例 3.102** 用 tr -s 命令把\r\n 替换为\n 并存入新文件 poss2.lnx 中。

```
liuhui@liuhui-VirtualBox:~$ tr -s '\r\n'  '\n'  < possible.txt  > poss2.lnx
liuhui@liuhui-VirtualBox:~$ cat -A poss2.lnx
dream it possible$
I will run I will soar$
I'm undefeated$
Jumping out of my skin pull the chord$
Yeah I believe it$
The past is everything$
So I'll dream $
It's not until you fall that you flyliuhui@liuhui-VirtualBox:~$
liuhui@liuhui-VirtualBox:~$
```

例 3.102 运行结果表明，用\n 代替\r\n 后，poss2.lnx 文件以\n 作为结束符，结果以 $ 标注。

**例 3.103** 将 jerrymos1 文件中所有大写字母转换成小写并保存到新文件 jrymos 中。

```
liuhui@liuhui-VirtualBox:~$ tr '[A-Z]'  '[a-z]' < jerrymos1   >  jrymos
liuhui@liuhui-VirtualBox:~$
```

执行命令后，可以用 cat 命令查看转换以后的文件中是否还有大写字母，也可以在图形界面中直接打开文件查看。

**例 3.104** 删除文件中的多个空格，以一个空格代替之。

```
liuhui@liuhui-VirtualBox:~$ tr -s '   '  ' ' < db2  > dbdt2
liuhui@liuhui-VirtualBox:~$ cat dbdt2
stdid stdnam stdage stdmaj
201912 zhangsan 18 secut
201913 tomcat 18 secut
201914 jerrmos 19 secut
201915 shiepstd 19 netwk
201915 supstd 20 netwk
liuhui@liuhui-VirtualBox:~$
```

## 3.4.3 剪切和粘贴

### 1. 剪切

剪切(cut)命令是从一个文件中剪切掉某些正文字段，并将它们送到标准输出显示。实际上 cut 命令是一个文件维护的命令，其语法格式如下。

```
cut [选项]…[文件名]…
```

其中常用的选项如下。

(1) -f：定义要截取的字段列。

(2) -c：要剪切的字符。

(3) -d：说明字段的分隔符(默认为 Tab)。

例 3.104 已经把 db2 文件中的多个空格分隔符整理为一个空格符作为分隔符后存入新文件 dbdt2 中，下面取出文件中的某一列数据，在命令中用-d 选项说明分隔符为空格。

**例 3.105** 从 dbdt2 文件中选取第 2 个字段的数据并存入文件 dbname 文件中。

```
liuhui@liuhui-VirtualBox:~$ cut -f2 -d ' '  dbdt2  > dbname
liuhui@liuhui-VirtualBox:~$ cat dbname
stdnam
zhangsan
tomcat
jerrmos
shiepstd
supstd
liuhui@liuhui-VirtualBox:~$ _
```

文件 dbdata 中的数据以逗号“,”作为分隔符，现在想取出里面第 1 列 stdid 的数据，则需要在命令中指定分隔符为“,”。

**例 3.106** 取出 dbdata 文件的第一列 stdid 的数据，并存入 dbid 文件中。

```
liuhui@liuhui-VirtualBox:~$ cut -f1 -d ','  < dbdata  > dbid
liuhui@liuhui-VirtualBox:~$ cat dbid
stdid
201912
201913
201914
201915
201915
liuhui@liuhui-VirtualBox:~$
```

### 2. 粘贴

粘贴(paste)命令主要用于将多个文件的内容合并输出，合并动作为按行将不同文件的行信息放在一行，即把每个文件的一行数据作为一列，把多个文件在宽度上合并在一起。默认情况下，不同文件的数据用空格或 Tab 键进行分隔，并把合并的结果写到标准输出上。

paste 命令的语法格式如下。

```
paste -d -s -file1 file2
```

paste 命令的选项含义说明如下。

(1) -d：指定不同于空格或 Tab 键的域分隔符。

(2) -s：将每个文件合并成行而不是按行粘贴。

(3) -：使用标准输入。

如果命令中没有文件名，或文件名使用了-，paste 将从标准输入设备读入数据，将多个文件合并成一个文件；如果在命令中使用了-d 选项，将更改输出的分隔符，默认分隔符是 Tab 字符。

**例 3.107** 把例 3.105 和例 3.106 中生成的文件 dbname 和 dbid 横向合并，合并两个文件成一个文件，两列数据之间以＃分隔。

```
liuhui@liuhui-VirtualBox:~$ paste -d '#' dbname  dbid  > dbstd
liuhui@liuhui-VirtualBox:~$ cat dbstd
stdnam#stdid
zhangsan#201912
tomcat#201913
jerrmos#201914
shiepstd#201915
supstd#201915
liuhui@liuhui-VirtualBox:~$
```

**例 3.108** 把第 1 个文件的所有数据放一行，第 2 个文件的所有数据放另一行。

```
liuhui@liuhui-VirtualBox:~$ paste -s  dbname  dbid  > dbstd2
liuhui@liuhui-VirtualBox:~$ cat dbstd2
stdnam  zhangsan        tomcat  jerrmos shiepstd        supstd
stdid   201912          201913          201914          201915          201915
liuhui@liuhui-VirtualBox:~$
```

由例 3.108 可以看出，paste 命令是对文件在宽度上做扩展，即被合并的每个文件的数据作为一列，从左向右排列成新文件。前文介绍的输出重定向>、>>和 cat f1 f2 > newfile 是在文件行上的合并，使文件长度增加，即纵向合并。

## 3.4.4 排序和管道操作

### 1. 排序

排序(sort)命令的功能是对正文进行排序并将结果送到标准输出设备，原有文件的内容不会发生变化，其正文数据既可以是一个文件，也可以是另一个命令的输出。

sort 命令的语法格式如下。

```
sort [选项] [文件名]
```

sort 命令中常用的选项如下。

(1) -r：进行反向排序(降序)。

(2) -f：忽略字符的大小写。

(3) -n：以数字的顺序进行排序。

(4) -u：去掉输出中的重复行。

(5) -t：如-t c，表示以字符 c 作为分隔符。

(6) -k：如-k N，表示按第 N 个字段排序。

(7) -k N1，N2：表示先按第 N1 个字段排序，之后再按第 N2 个字段排序。

前面曾经查看过系统的密码文件/etc/passwd，里面详细列出了用户名、标识、权限代码、家目录等信息，各列数据以：分隔。

**例 3.109** 查看 passwd 文件的部分内容。

```
liuhui@liuhui-VirtualBox:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
```

**例 3.110** 对 passwd 文件的第 3 列数据进行排序显示。

```
liuhui@liuhui-VirtualBox:~$ sort -t: -k3  /etc/passwd
root:x:0:0:root:/root:/bin/bash
liuhui:x:1000:1000:liuhui,,,:/home/liuhui:/bin/bash
syslog:x:100:103::/home/syslog:/bin/false
shiephl:x:1001:1001::/home/shiephl:
tomcat:x:1002:100::/home/tomcat:/bin/bash
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

例 3.110 运行结果表明，以：作为分隔符，按第 3 列数据的升序进行了排序，系统默认以字符串的 ASCII 码作为排序依据。如果想把第 3 列的数据，如 1、100、13、3 等，看作数字，则可以在命令中使用-n 选项把这些值当作数字。

**例 3.111** 使用-n 选项按整数数字排序。

```
liuhui@liuhui-VirtualBox:~$ sort -t: -k3 -n  /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats
-
```

例 3.111 运行结果表明，-n 选项可以使命令把数字字符当作整数来进行大小的排序。

### 2. 管道操作

管道操作符 | 可以连接两个或多个 Linux 命令，其语法格式如下。

```
命令 1 | 命令 2 …
```

功能说明：将命令 1 的标准输出重定向为命令 2 的标准输入；标准错误信息(stderr)并不通过管道传播，命令 1 的错误信息不会传给命令 2，命令 2 的错误信息也不会传给下一个命令等。任何两个命令之间都可以插入管道操作符，管道操作符之前的命令把输出写到标准输出设备上，管道操作符之后的命令再把标准输出设备上的输出当作它的输入。简言之，就是把管道操作符|左边的命令运行结果作为右边命令的输入。

**例 3.112** 统计系统上工作的用户有多少。

```
liuhui@liuhui-VirtualBox:~$ who | wc -l
2
liuhui@liuhui-VirtualBox:~$ _
```

如果系统上用户很少，直接用 who 命令就可以看出有几个用户；但如果系统上用户很多，有几百甚至几千个，那么用管道|让系统统计就会省很多事。

**例 3.113** 统计在 Linux 系统上一共创建了多少用户。

```
liuhui@liuhui-VirtualBox:~$ cat /etc/passwd | wc -l
39
liuhui@liuhui-VirtualBox:~$ _
```

**例 3.114** 利用 wc 命令统计给定目录下的文件个数。

```
liuhui@liuhui-VirtualBox:~$ read dirpath
backup
liuhui@liuhui-VirtualBox:~$ find $dirpath -type f | wc -l
33
liuhui@liuhui-VirtualBox:~$ _
```

例 3.114 中，dirpath 为目录路径，由 read 读入。find 命令找出文件，通过管道把这些文件的文件名传输给 wc 命令，统计出文件名的个数，即给定目录下文件的个数。

如果想列出系统目录/bin 下所有文件和目录细节，运行结果可能会显示好几页，需要按 PgUp、PgDn 或者上下方向键来翻屏。如果把 ls 命令的输出通过管道送到 more 命令中，就可以分页查看相关信息了。

**例 3.115** 将命令 ls 的结果输入 more 命令。

```
liuhui@liuhui-VirtualBox:~$ ls -lF /bin | more
total 10736
-rwxr-xr-x 1 root root 1029720 10月  7  2014 bash*
-rwxr-xr-x 1 root root   31288 10月 22  2014 bunzip2*
-rwxr-xr-x 1 root root 1931720  8月  8  2014 busybox*
-rwxr-xr-x 1 root root   31288 10月 22  2014 bzcat*
lrwxrwxrwx 1 root root       6  8月 29 15:03 bzcmp -> bzdiff*
-rwxr-xr-x 1 root root    2140 10月 22  2014 bzdiff*
lrwxrwxrwx 1 root root       6  8月 29 15:03 bzegrep -> bzgrep*
-rwxr-xr-x 1 root root    4877 10月 22  2014 bzexe*
lrwxrwxrwx 1 root root       6  8月 29 15:03 bzfgrep -> bzgrep*
-rwxr-xr-x 1 root root    3642 10月 22  2014 bzgrep*
-rwxr-xr-x 1 root root  150912  9月  8  2014 cp*
-rwxr-xr-x 1 root root  137272  4月 30  2014 cpio*
--More--
```

## 3.5 命令行的执行方式

### 3.5.1 命令的顺序执行和并发执行

#### 1. 命令的顺序执行

用分号来分隔两个命令，命令格式如下。

```
command1;command2;command3;…;commandN
```

说明：各命令的执行结果不会影响其他命令的执行。换句话说，每个命令都会执行，但不保证每个命令都执行成功。

**例 3.116** 顺序执行命令。

```
liuhui@liuhui-VirtualBox:~$ echo "date:" ; date ; echo "user:" ; whoami
date:
2019年 09月 25日 星期三 19:47:50 CST
user:
liuhui
liuhui@liuhui-VirtualBox:~$ _
```

**例 3.117** 依次显示目录名和内容。

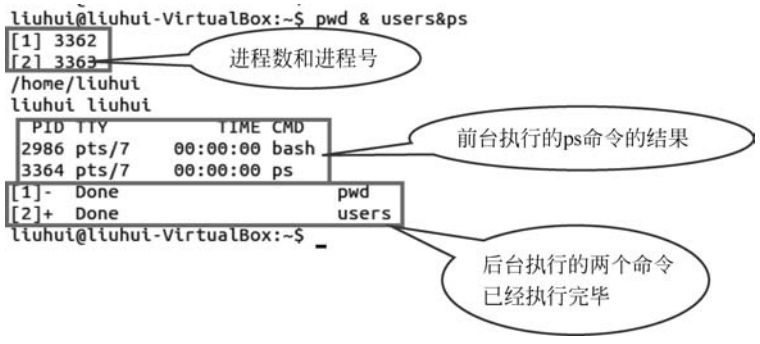
```
liuhui@liuhui-VirtualBox:~$ pwd ; users; ps; echo " commands in order"
/home/liuhui
liuhui liuhui
  PID TTY          TIME CMD
 2986 pts/7    00:00:00 bash
 3360 pts/7    00:00:00 ps
 commands in order
liuhui@liuhui-VirtualBox:~$
```

### 2. 命令的并发执行

命令并发执行的语法格式如下。

```
command1& command2& command3& … . commandN&
```

**例 3.118**　几个命令同时执行。




## 3.5.2　命令行中的&&和||操作

### 1. 命令行中的&&操作

&&：只有在前面的所有命令都执行成功的情况下才执行后一条命令，这样可以保证所有命令都成功执行，命令语法格式如下。

```
command1 && command2 && command3 && … && commandN
```

**例 3.119**　先创建一个目录 newdir，然后在该目录中创建两个新文件 d1 和 d2。

```
liuhui@liuhui-VirtualBox:~$ mkdir newdir && cd  newdir && touch d1 & touchd2
[1] 3379
touchd2: command not found
[1]+  Done                    mkdir newdir && cd newdir && touch d1
liuhui@liuhui-VirtualBox:~$ mkdir newdir && cd  newdir && touch d1 & touch d2
[1] 3389
liuhui@liuhui-VirtualBox:~$ mkdir: cannot create directory ‘newdir’: File exists
^C
[1]+  Exit 1                  mkdir newdir && cd newdir && touch d1
liuhui@liuhui-VirtualBox:~$ mkdir newdir && cd  newdir && touch d1 & touch d2
[1] 3394
liuhui@liuhui-VirtualBox:~$ ls -l newdir
total 0
-rw-rw-r-- 1 liuhui liuhui 0  9月 25 20:17 d1
[1]+  Done                    mkdir newdir && cd newdir && touch d1
```

例 3.119 运行结果表明，先用 && 命令的执行方式创建了新目录并在新目录中成功新建文件 d1，因为命令 touch d1 和 touch d2 是并行执行的，但输入命令时输错了，所以 touch d2 命令没有执行。第 2 次输入的命令都正确了，但由于第 1 次部分命令正确执行了，已经有目录 newdir 了，第 2 次希望创建新目录与第 1 次同名的命令没有成功执行，那么后面的命令就不会被执行，也就不提示新建的文件重名了。

**例 3.120**　查看当前目录下是否有 sample 文件，如果有，则删除该文件，并提示文件已被删除。

```
liuhui@liuhui-VirtualBox:~$ ls sample && rm sample && echo  "sample is deleted"
ls: cannot access sample: No such file or directory
liuhui@liuhui-VirtualBox:~$
```

当前工作目录下没有 sample 这个文件，所以第 1 个命令就没有正确执行，后面的命令也不执行。

### 2. 命令行中的||操作

||操作允许持续执行一系列命令，直到有一条命令成功为止，其后的命令将不再被执行。命令语法格式如下。

```
command1 || command2 || command3 || … || commandN
```

**例 3.121** 查看当前目录下是否有 sample 文件，如果有，则后面的命令就不执行；如果没有，则新建该文件，并提示文件创建成功。

```
liuhui@liuhui-VirtualBox:~$ ls sample || touch sample && echo "new file is created"
ls: cannot access sample: No such file or directory
new file is created
liuhui@liuhui-VirtualBox:~$ ls sample || touch sample && echo "new file is created"
sample
new file is created
liuhui@liuhui-VirtualBox:~$
liuhui@liuhui-VirtualBox:~$ ls sample || (touch sample && echo "new file is created")
sample
liuhui@liuhui-VirtualBox:~$
```

例 3.121 运行结果表明，命令||和 && 可以组合使用，第 1 次执行时没有 sample 文件，所以提示无此文件，然后执行 touch 命令创建新文件；第 2 次再次执行此命令，因为第 1 次已经新建了 sample 文件，所以第 2 次可以正确执行第 1 个命令；echo 命令是同时执行的，所以也提示新建了文件，但这不是预期的结果。所以，应该把后两个命令连在一起与第 1 个命令||或操作。再尝试第 3 次的执行，这才是预期的结果。

## 3.5.3 命令的后台执行及转换

### 1. 命令的后台执行命令&

在终端或控制台工作时，可能不希望由于运行一个作业而占据屏幕，因为可能还有更重要的事情要做，比如阅读电子邮件。对于密集访问磁盘的进程，用户更希望它能够在每天的非负荷高峰时间段运行(如凌晨)，为了使这些进程能够在后台运行，也就是不在终端屏幕上运行，可以让命令在后台运行。命令在后台运行以后，终端上就会显示一个进程号，表示该命令运行的进程编号，用户可以用该进程号来监控该进程，或杀死它。

命令的前台和后台执行命令的语法如下。

(1) command：在前台执行，就是一般的命令运行形式。

(2) command&：在后台执行，就是并行执行，在运行结果中会提示[1]3379 或[1]done，就表示这个命令在后台的进程号。

适合在后台运行的命令有 find、费时的排序及一些 Shell 脚本。在后台运行作业时要当心，需要用户交互的命令不要放在后台执行，因为在后台运行时看不到运行过程，就不会输入数据，这样机器就会“傻等”直到接收到数据。同时，作业在后台运行一样会将结果输出到屏幕上，干扰用户的工作。如果放在后台运行的作业会产生大量的输出，最好使用如下命令把它的输出重定向到某个文件中，这样，所有标准输出和错误输出都将被重定向到一个叫做 out.file 的文件中。

```
command  >  out.file  2>&1  &
```

command>out.file：将 command 的输出重定向到 out.file 文件，即输出内容不打印到屏幕上，而是输出到 out.file 文件中。

2>&1：将标准错误信息重定向到标准输出，这里的标准输出已经重定向到了 out.file 文件，所以将标准错误信息也输出到了 out.file 文件中。最后一个 & 是让该命令在后台执行。

试想 2>1 代表什么？2 与>结合代表错误重定向，而 1 则代表错误重定向到文件 1，而不代表标准输出；如果换成 2>&1，& 与 1 结合就代表标准输出了，就变成错误重定向到标准输出。

**例 3.122** 在工作目录下查找名字为 t1.txt 的文件，如果找到，则把结果写入工作目录的 file1.pt 文件中；如果有错误，则把错误信息写入/tmp/file1.err 文件中。

```
liuhui@liuhui-VirtualBox:~$ find   t1.txt 1> file1.pt  2> /tmp/file1.err
liuhui@liuhui-VirtualBox:~$ finde   t1.txt 1> file1.pt  2> /tmp/file1.err
liuhui@liuhui-VirtualBox:~$ find    t1.txt 1> file1.pt  2> /tmp/file1.err &
[1] 3206
liuhui@liuhui-VirtualBox:~$ findd    t1.txt 1> file1.pt  2> /tmp/file1.err &
[2] 3219
[1]   Done                    find t1.txt > file1.pt 2> /tmp/file1.err
liuhui@liuhui-VirtualBox:~$
[2]+  Exit 127                findd t1.txt > file1.pt 2> /tmp/file1.err
```

为了对比，例 3.122 先在前台执行命令，然后再用后台执行的方式做对比，可以看到在后台运行的命令会在终端上提示命令运行时对应的进程号和运行的命令名。为了对比错误信息，这里故意把命令的名字输入错误。

**例 3.123** 查看 file1.pt 和 file1.err 命令的运行结果文件。

```
liuhui@liuhui-VirtualBox:~$ cat /tmp/file1.err
No command 'findd' found, did you mean:
 Command 'findg' from package 'ncl-ncarg' (universe)
 Command 'findv' from package 'polylib-utils' (universe)
 Command 'find' from package 'findutils' (main)
findd: command not found
[1]+  Done                    find t1.txt > file1.pt 2> /tmp/file1.err
liuhui@liuhui-VirtualBox:~$ cat file1.pt
liuhui@liuhui-VirtualBox:~$
```

### 2. 不中断命令 nohup

使用命令 & 后，作业被提交到后台运行，当前控制台没有被占用，但是一旦把当前控制台关掉（例如，退出账户），作业就会停止运行。nohup 命令可以在退出账户之后继续运行相应的进程。nohup 就是不挂起（no hang up）的意思。该命令的一般形式如下。

```
nohup command &
```

如果使用 nohup 命令提交作业，那么在默认情况下该作业的所有输出都被重定向到一个名为 nohup.out 的文件中，可以使用重定向符把结果输出到指定的其他文件，如下示例。

```
nohup command > myout.file 2>&1 &
```

使用了 nohup 之后，还是有可能在当前账户非正常退出或者结束时，命令的执行还是自动结束了，所以在使用 nohup 让命令在后台运行之后，需要使用 exit 正常退出当前账户，才能保证命令一直在后台运行。

**例 3.124** 不中断的后台作业。

```
liuhui@liuhui-VirtualBox:~$ nohup find t2.txt > t2.out  2>&1 &
[1] 3377
liuhui@liuhui-VirtualBox:~$ nohup findd t2.txt > t2.out  2>&1 &
[2] 3389
[1]   Exit 1                  nohup find t2.txt > t2.out 2>&1
liuhui@liuhui-VirtualBox:~$
```


按 Ctrl+Z 快捷键可以将一个正在前台执行的命令放到后台，并且处于暂停状态。

按 Ctrl+C 快捷键可以终止前台命令。当一个命令死机或者想提前终止命令的执行时，按 Ctrl+C 快捷键就可以终止命令的执行。

### 3. fg 和 bg 命令

fg 命令可把后台的进程移到前台执行，bg 命令可把前台的进程移到后台执行。

如果前台运行的一个程序需要很长的时间，但是需要干其他事情，就可以按 Ctrl+Z 快捷键挂起这个程序，然后可以看到如下系统提示(方括号中的是作业号)。

```
[1]+  Stopped                 top
liuhui@liuhui-VirtualBox:~$
```

然后可以把程序调度到后台执行，命令如下。bg 后面的数字为作业号。

```
liuhui@liuhui-virtualBox: ~ $ bg 1
```

如果想把它调回到前台运行，可以用如下命令。

```
liuhui@liuhui-virtualBox: ~ $ fg 1
```

例如，先运行一个耗时较长的命令，然后使用 top 命令查看服务器负载信息，实时动态刷新显示服务器状态信息，且可以通过交互式命令自定义显示内容。

**例 3.125** 查看被终止的命令工作号。

```
liuhui@liuhui-VirtualBox:~$ top

top - 21:13:53 up  1:41,  2 users,  load average: 0.16, 0.09, 0.12
Tasks: 170 total,   1 running, 169 sleeping,   0 stopped,   0 zombie
%Cpu(s):  6.9 us,  0.9 sy,  0.0 ni, 91.8 id,  0.3 wa,  0.0 hi,  0.0 si,  0.0
KiB Mem:   1797636 total,  1090488 used,   707148 free,    79056 buffers
KiB Swap:  1843196 total,        0 used,  1843196 free.   386596 cached Mem

[1]+  Stopped                 top
liuhui@liuhui-VirtualBox:~$
```

例 3.125 运行结果界面会显示“[1]+ Stopped top”，1 表示是 top 命令的工作号，然后用 bg 命令可将 top 命令转为后台运行。

**例 3.126** 将 top 命令转为后台执行。

```
liuhui@liuhui-VirtualBox:~$ bg 1
[1]+ top &
liuhui@liuhui-VirtualBox:~$
```

因为 top 命令可以监控服务器的状态信息，转到后台运行后，就看不到运行结果了，但命令一直处于运行状态，如果想使用其他命令，需要按 Ctrl+C 快捷键退出当前的命令，然后才

可以开始一个新的命令。所以在例 3.126 运行 bg 命令后按 Ctrl+C 快捷键退出当前的 bg 命令,然后再使用 fg 1 命令把 top 命令转为前台运行,输入 fg 1 后 top 命令会接着上面的继续运行。

```
liuhui@liuhui-virtualBox: ~ $ fg 1
```

#### 4. jobs 命令

jobs 命令可以显示所有挂起的和后台进程的作业号,确定哪一个是当前的进程。在 jobs 命令的输出里面,当前进程前面有一个 + 标志,而其他进程通常前面加一个 - 来标志。

jobs 命令语法如下。

```
jobs [option] [ %jobID]
```

功能说明:显示所有在 jobID 中指明的被挂起的和后台进程的状态;如果没有列表,则显示当前进程的状态。可选参数 jobID 可以是以%符号开头、以空格符分隔的一串作业号。

常用选项说明如下。

-l:显示该作业的 PID。

jobs -l:显示所有任务的 PID,jobs 命令的状态可以是 running、stopped、terminated。

重新输入 top 命令来监控服务器的状态信息,然后按 Ctrl+Z 快捷键暂停 top 命令的执行,按 Ctrl+C 快捷键退出 top 命令,接着输入 jobs 命令查看当前的进程作业号。

**例 3.127** jobs 命令的使用。

```
[1]+  Stopped                 top
liuhui@liuhui-VirtualBox:~$ ^C
liuhui@liuhui-VirtualBox:~$ jobs
[1]+  Stopped                 top
liuhui@liuhui-VirtualBox:~$ _
```

例 3.127 运行结果表明 top 命令被中断了。

## 3.6 文件系统挂载和卸载

### 3.6.1 文件系统挂载

Linux 系统中,要访问根目录以外的文件,需要将其“关联”到根目录下的某个目录中,这种关联操作就是“挂载”,这个目录就是“挂载点”,解除此关联关系的过程称为“卸载”。


作为“挂载点”的目录需要满足以下 3 个要求。

- 目录事先存在,可以用 mkdir 命令新建目录。
- 挂载点目录不可被其他进程使用。
- 挂载点下原有文件被隐藏。

#### 1. 挂载命令

挂载命令的语法格式如下。

```
mount [ - fnrsvw] [ - t vfstype] [ - o options] device dir
```

device：指明要挂载的设备。

dir：挂载点，执行挂载操作之前必须事先存在。建议使用空目录，因为挂载后原有目录中的内容会被隐藏。

mount 命令的常用选项如下。

(1) -t：vsftype，指定要挂载的设备上的文件系统类型。

(2) -r：readonly，只读挂载。

(3) -w：read and write，读写挂载。

(4) -n：不更新/etc/mtab。

(5) -a：自动挂载所有支持自动挂载的设备，定义在/etc/fstab 文件中，且挂载选项中有“自动挂载”功能。

(6) -o：特殊选项。常用的特殊选项如下。

- rw/ro：读写/只读，定义文件挂载时是否具有读写权限，默认为 rw(读写)。
- async/ sync：异步/同步 I/O，默认为 async(异步)。
- atime /noatime：更新访问时间/不更新访问时间，定义访问分区文件时是否更新文件的访问时间，默认为 atime(更新)。
- auto/ noauto：自动/手动，定义执行 mount -a 命令时，是否自动挂载/etc/fstab 文件内容，默认为 auto(自动)。

**注意**

查看内核追踪到的已挂载的所有设备，可用命令 cat /proc/mounts。上述选项可多个同时使用，彼此间使用逗号分隔；默认挂载选项包括 rw、suid、dev、exec、auto、nouser 及 async。

### 2. 挂载移动硬盘

对 Linux 系统而言，USB 接口的移动硬盘是被当作 SCSI 设备对待的。插入移动硬盘之前，应先用 fdisk -l 或 more /proc/partitions 命令查看系统硬盘的分区情况。

**注意**

fdisk 命令需要 root 用户的权限，普通用户无权使用，故应该先用 su - 将用户切换为 root 用户，或者使用 sudo fdisk -l 命令短暂使用 root 权限。

**例 3.128** 查看分区情况。

```
root@liuhui-VirtualBox:~# more /proc/partitions
major minor  #blocks  name

  11        0      75354 sr0
   8        0   41943040 sda
   8        1     248832 sda1
   8        2          1 sda2
   8        5   41691136 sda5
 252        0   39845888 dm-0
 252        1    1843200 dm-1
root@liuhui-VirtualBox:~# _
```

接入移动硬盘后，再用 fdisk -l 或 more /proc/partitions 命令查看系统的硬盘和硬盘分区情况，应该可以发现多了一个 SCSI 硬盘/dev/sdb1，这就是移动硬盘的逻辑分区。

**例 3.129** 查看接入移动硬盘后系统中硬盘的分区情况。

```
root@liuhui-VirtualBox:~# fdisk -l

Disk /dev/sda: 40 GiB, 42949672960 bytes, 83886080 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x06f61fae

Device     Boot  Start      End  Sectors  Size Id Type
/dev/sda1  *      2048   499711   497664  243M 83 Linux
/dev/sda2       501758 83884031 83382274 39.8G  5 Extended
/dev/sda5       501760 83884031 83382272 39.8G 8e Linux LVM

/dev/sdb1  *       63 3907024064 3907024002  1.8T  7 HPFS/NTFS/exFAT

root@liuhui-VirtualBox:~# _
```

如果接入移动硬盘后无法看到多出来的逻辑分区，则需要先安装 USB 设备的驱动程序。安装方法是：在“设置”→“USB 设备”中添加 USB 筛选器，会自动安装 USB 的驱动程序。安装成功后，在图形界面中可以直接查看 U 盘内容，在终端用 fdisk -l 命令就可以看到 sdb 盘符了，然后使用 mount 命令就可以在终端使用移动硬盘了，挂载命令执行之前，要确保存在移动硬盘的挂载点，即确保挂载目录存在，如果不存在，则应先创建目录。

**例 3.130** 创建挂载目录。

```
root@liuhui-VirtualBox:~# mkdir /mnt/usbsdb1
root@liuhui-VirtualBox:~#
```

**例 3.131** 挂载移动硬盘。

```
root@liuhui-VirtualBox:~# mount -t vfat /dev/sdb1  /mnt/usbsdb1
mount: /dev/sdb1 is already mounted or /mnt/usbsdb1 busy
       /dev/sdb1 is already mounted on /media/liuhui/0000F6990009976F
root@liuhui-VirtualBox:~#
```

对于例 3.131 来说，因为安装的是 Ubuntu 18.01，并且安装了增强功能，所以在接入移动硬盘时系统自动安装了驱动程序，也就是该移动硬盘可以直接使用，所以在挂载时提示已经挂载成功，挂载的位置在/media/liuhui/0000F6，最后的一长串数字是移动硬盘的卷标。因为事先没有设置该移动硬盘的卷标，所以系统给它分配了一个标识符。命令运行时，系统提示/mnt/usbdb1 忙，是因为挂载的移动硬盘容量太大，这里的移动硬盘容量是 2.0TB，无法挂载在/mnt 目录下，系统将其自动挂载在了/media/liuhui/，如图 3.6 所示。

在图形界面中可以直接查看并打开相关文件。

**注意**

对 NTFS 格式的磁盘分区应使用 mount -t ntfs 参数，对 FAT32 格式的磁盘分区应使用 mount -t vfat 参数。若汉字文件名显示为乱码或不显示，可以使用如下命令。

```
#mount -t ntfs -o iocharset=cp936 /dev/sdc1 /mnt/usbhd1
#mount -t vfat -o iocharset=cp936 /dev/sdc5 /mnt/usbhd2
```

### 3. 挂载 U 盘

对 Linux 系统而言，U 盘也是被当作 SCSI 设备对待的，使用方法和移动硬盘完全一样。

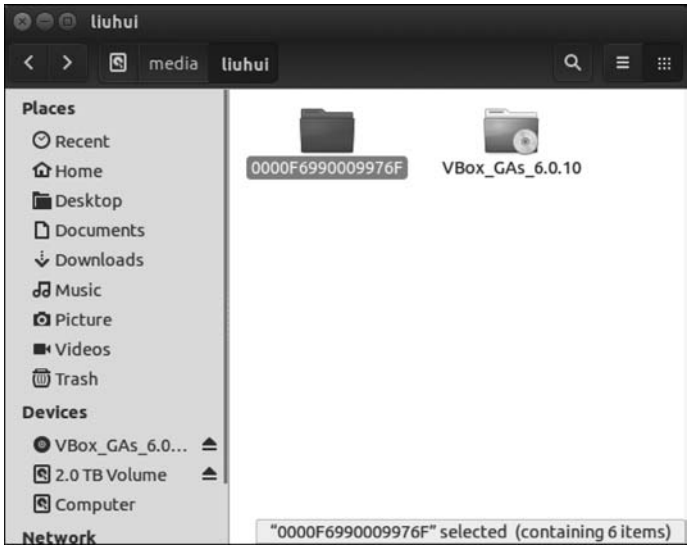



图 3.6　大容量移动硬盘的挂载点

插入 U 盘之前和插入之后，应先用 fdisk -l 或 more /proc/partitions 命令查看系统中硬盘的分区情况。

**例 3.132**　查看插入 U 盘后硬盘的分区情况。

```
/dev/sdb1  *       16 3903486 3903471  1.9G  b W95 FAT32

root@liuhui-VirtualBox:~# ls -l /dev/sdb1
brw-rw---- 1 root disk 8, 17 10月  3 11:19 /dev/sdb1
root@liuhui-VirtualBox:~#
```

由于本系统安装了增强功能，因此在接入 U 盘时系统自动安装了驱动程序，也就是该 U 盘可以直接使用。但是 Linux 系统把 U 盘当作块文件，不能直接打开，所以还需要把 U 盘挂载在挂载点上。

**例 3.133**　挂载 U 盘。

```
root@liuhui-VirtualBox:~# mount -t vfat /dev/sdb1  /mnt/usbsdb1
root@liuhui-VirtualBox:~# ls -l /mnt/usbsdb1
total 1803104
drwx------ 2 liuhui liuhui      4096 10月  3 11:28 2
-rw-r--r-- 1 liuhui liuhui     84994  6月  5 15:22 2018252学习方法主题班会.pptx
-rw-r--r-- 1 liuhui liuhui   7552266 11月  7  2018 2018指津答案.pdf
drwx------ 2 liuhui liuhui      4096  5月  5 13:17 2019市教委重点课程
-rw-r--r-- 1 liuhui liuhui     58368  5月 21 15:37 2019试卷模板.doc
drwx------ 5 liuhui liuhui      4096  4月 23 10:06 linuxhl资料
-rw-r--r-- 1 liuhui liuhui 926285824  9月 23  2015 Office 2010.iso
-rw-r--r-- 1 liuhui liuhui 912098920 12月  7  2017 Office2010安装程序(64位VOL版)
.rar
drwx------ 3 liuhui liuhui      4096  5月 21 13:24 RECYCLER
```

挂载后，在终端中，该 U 盘就是 usbsdb1，系统不认识原来的卷标 HL。但在图形界面，左侧的 Devices 中显示的还是原来的卷标 HL。在终端可以使用针对文件的各种操作命令，也可以在图形界面直接查看并打开，如图 3.7 所示。

## 3.6.2　文件系统卸载

卸载文件系统可使用 umount 命令。


图 3.7 图形界面下查看 U 盘信息

**例 3.134** 卸载移动硬盘并查看分区情况。

```
root@liuhui-VirtualBox:~# umount /dev/sdb1
root@liuhui-VirtualBox:~#
root@liuhui-VirtualBox:~# more /proc/partitions
major minor  #blocks  name

  11        0      75354 sr0
   8        0   41943040 sda
   8        1     248832 sda1
   8        2          1 sda2
   8        5   41691136 sda5
 252        0   39845888 dm-0
 252        1    1843200 dm-1
root@liuhui-VirtualBox:~# _
```

**例 3.135** 卸载 U 盘。

```
root@liuhui-VirtualBox:~# umount /dev/sdb1
root@liuhui-VirtualBox:~# ls -l /mnt/usbsdb1
total 0
root@liuhui-VirtualBox:~# umount /mnt/usbsdb1
umount: /mnt/usbsdb1: not mounted
root@liuhui-VirtualBox:~#
```

在终端执行 umount 命令后，即使没有拔出 U 盘，在终端中也看不到 U 盘及里面的内容了。在终端使用 fdisk 和 more 命令，还可以看到 sdb1，即 U 盘，在图形界面中还可以查看 U 盘及其内容。

**例 3.136** 验证卸载但不拔出 U 盘的情况。

```
root@liuhui-VirtualBox:~# fdisk -l

/dev/sdb1  *       16 3903486 3903471  1.9G  b W95 FAT32

root@liuhui-VirtualBox:~# more /proc/partitions
major minor  #blocks  name

  11        0      75354 sr0
   8        0   41943040 sda
   8        1     248832 sda1
   8        2          1 sda2
   8        5   41691136 sda5
 252        0   39845888 dm-0
 252        1    1843200 dm-1
   8       16    1951743 sdb
   8       17    1951735 sdb1
root@liuhui-VirtualBox:~# _
```

这是由于U盘还插在USB接口上，fdisk -l命令是查看逻辑分区的，因此可以看到sdb1的逻辑分区；但又由于没有挂载U盘，因此无法看到U盘的内容。

## 上机实验：Linux文件系统命令的使用

### 1. 实验目的

(1) 通过对文件的各种操作，掌握Linux环境中文件系统的基本思想，理解一切皆文件的理念。

(2) 通过对mkdir、cp、cd、ls、mv、chmod及rm等文件系统命令的操作，掌握Linux系统中文件系统命令的用法。

(3) 通过对文件系统的挂载使用，理解驱动程序的原理。

### 2. 实验任务

(1) 文件的新建、复制命令的使用。

(2) 文件内容的搜索命令cat、head、grep、find的使用。

(3) 文件内容的比较命令diff、sdiff、expand的使用。

(4) 文件的输入输出重定向、排序等命令的使用。

(5) 文件系统的挂载命令的使用。

### 3. 实验环境

装有Windows系统的计算机；虚拟机安装VirtualBox+ Linux Ubuntu操作系统。

### 4. 实验题目

**任务1**：文件的新建命令touch、复制命令cp、文件类型的查看命令file的使用。

在当前用户的家目录下新建一个文件，文件名是自己的学号，把这个文件复制到一个目录下，查看该文件的类型。

**任务2**：文件编辑器gedit、vi、emacs等的使用。

分别使用上述3种文本编辑器，在任务1中新建的文件中输入内容，然后保存，体会3种编辑器的使用不同之处。

**任务3**：文件内容的搜索命令cat、head、grep、find的使用。

在任务2中输入内容的文件中搜索相关的关键词。

**任务4**：文件内容的比较命令diff、sdiff、expand的使用。

新建一个文件，输入内容，内容与任务2中的内容差别小一点，使用diff、sdiff命令判断两个文件的异同；再把新建的文件复制一份，比较与原文件之间的异同。

**任务5**：文件的归档tar、解压缩gzip等命令的使用。

把当前用户家目录下的文件归档为一个文件，再把当前用户家目录下的文件压缩，查看这两次操作后文件大小的异同。

**任务 6**：文件系统的挂载命令 mount 的使用。

把自己的 U 盘插入计算机，在文件系统中查看 U 盘；是否可以在终端中直接查看 U 盘内容？使用 mount 命令挂载，然后再查看，有何异同？

**5. 实验心得**

总结上机中遇到的问题及解决问题过程中的收获、心得体会等。