第 3 章 Linux 字符界面操作基础

3.1 字符操作界面简介

内容提要

1. 字符界面的使用方法。
2. 本地登录和远程登录。
3. 学会使用 Putty。
4. 理解系统运行级别及其切换方法。
5. 掌握常用的系统关机和重启命令。

3.1.1 选择在字符界面下工作

为什么使用字符工作方式

Linux 是一种类 UNIX 操作系统。在 UNIX 发展的早期，类 UNIX 操作系统根本没有图形操作界面，只有字符工作模式。后来随着 GUI 的发展，人们在类 UNIX 操作系统上开发了 XWindow 系统，使类 UNIX 系统有了图形用户界面。虽然图形用户界面的操作更简单，但是字符操作方式仍然沿用至今，这主要是因为：

- 在字符操作方式下可以高效地完成所有的任务，尤其是系统管理任务。
- 系统管理任务通常在远程进行，而远程登录后进入的是字符工作方式。
- 由于使用字符界面不用启动图形工作环境，大大地节省了系统资源开销。

进入字符工作方式的方法

可以使用如下的 3 种方法进入字符工作方式：

- 在图形环境下开启终端窗口进入字符工作方式。
- 在系统启动后直接进入字符工作方式。
- 使用远程登录方式（Telnet 或 SSH）进入字符工作方式。

3.1.2 虚拟控制台和本地登录

虚拟控制台

如果在系统启动时直接进入字符工作方式，系统将提供多个（默认为 6 个）虚拟控制台。每个虚拟控制台可以独立使用，互不影响。

可以使用组合键“Alt+F1”~“Alt+F6”进行多个虚拟控制台之间的切换。如果用户使用 startx 命令在字符界面下启动了图形环境，那么可以使用组合键“Ctrl+Alt+F1”~“Ctrl+Alt+F6”切换字符虚拟终端，使用组合键“Ctrl+Alt+F7”切换到图形界面。

本地登录和注销

在图 3-1 所示的登录界面输入用户名和口令即可登录。

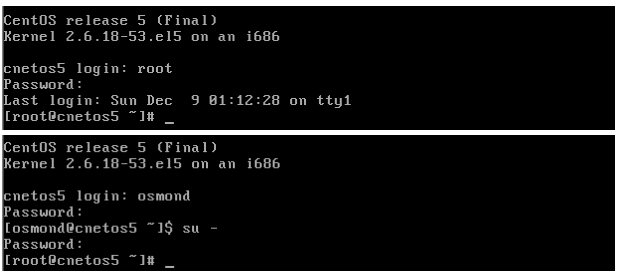

图 3-1　在字符界面下登录 Linux

若要注销登录，用户可以在当前的登录终端上输入 logout 命令，或使用“Ctrl+d”快捷键。超级用户的命令提示符是“#”，普通用户的命令提示符是“$”。

提示：

1．Linux 系统是严格区分大小写的，无论用户名，还是文件名或设备名都是如此。即：ABC、Abc、abc 代表三个不同的用户名或文件名。

2．基于安全的考虑，一般应该使用普通用户登录系统，不要使用 root 用户登录，当需要进行超级用户的工作时可以使用 su -命令切换为超级用户身份。初学者尤其要注意这一点。

3.1.3　远程登录 Linux 系统

在 Linux 环境下使用 ssh 登录远程 Linux 系统

Linux 下的 ssh 命令是 OpenSSH 的客户端程序。要登录远程 Linux 系统，必须保证远程 Linux 系统上启动了 OpenSSH 服务器。使用 ssh 命令登录远程 OpenSSH 服务器的命令格式是：

```
$ ssh 远程主机上的用户名@远程主机的 IP 地址或 FQDN
```

下面给出一个使用 ssh 命令登录远程 Linux 系统的操作步骤。以下为在 Linux 环境下使用 ssh 命令登录远程 CentOS 系统的示例。

```
## 以 root 身份登录 IP 地址为 192.168.0.101 的 CentOS 系统
osmond@ubuntu:~$  ssh root@192.168.0.101
The authenticity of host '192.168.0.101 (192.168.0.101)' can't be established.
RSA key fingerprint is 6e:0c:62:70:0a:b9:d7:fe:11:7f:78:b0:af:28:69:7c.
Are you sure you want to continue connecting (yes/no)? yes
## 如果第一次使用该账号进行 ssh 登录需确认密钥，选择“yes”才可继续登录过程
Warning: Permanently added '192.168.0.101' (RSA) to the list of known hosts.
root@192.168.0.101's password:
## 在此输入 root 用户在主机 192.168.0.101 上的口令，口令输入过程中没有回显
Last login: Sun Dec  9 01:18:13 2007
```

```
[root@cnetos5 ~]#
## 正确登录后出现 Shell 提示符
[root@cnetos5 ~]# logout
## 输入 logout 命令注销远程连接
Connection to 192.168.0.101 closed.
osmond@ubuntu:~$
## 返回本地 Shell
```

在 Windows 环境下使用 putty 登录远程 Linux 系统

在 Windows 下，用户可以用 putty 来远程登录 Linux 系统。putty 是一个绿色软件，无需安装，可直接使用。putty 支持 telnet、ssh、rlogin 等多种连接方式。

- putty 的下载地址是：

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html。

- putty 中文版的下载地址是：http://wrc.gro.clinux.org/putty/。

下面给出一个使用 putty 登录远程 Linux 系统的操作步骤。

步骤 1：双击 Windows 桌面上的 putty 图标，启动 putty。

步骤 2：对 putty 进行设置。

1．Session 设置

- 在 Host Name 一栏中添入远程主机的主机名或 IP 地址。
- 在 Protocol 一栏中选择 SSH 连接会话的类型。

2．Window 设置

- 在 Appearance 页中的 Font Settings 单击“Change…”按钮设置字体
- 在 Translation 页中的“Received data assumed to be in which character set:”下的下拉列表中选“UTF-8”。

3．保存当前会话配置

- 在 Session 页中的 Saved Sessions 下的文本框中起一个名字，如“CentOS”。
- 单击右边的“Save”按钮。

步骤 3：保存会话之后就可以用双击会话名的方法登录远程主机了，如双击会话名“CentOS”，如图 3-2 所示。

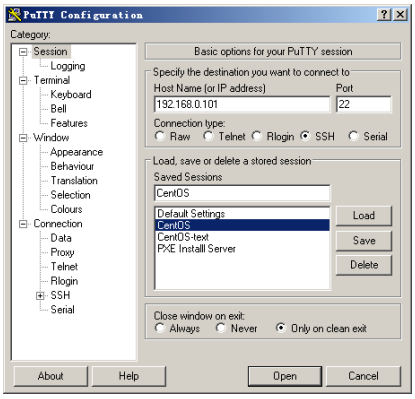

图 3-2　双击会话名连接远程 CentOS 系统

步骤 4：如果是第一次连接远程系统，putty 会提示在本地主机上没有远程系统的公共

密钥，询问用户是否要继续连接，如图 3-3 所示单击“是”继续。

步骤 5：建立与远程主机的连接之后，输入用户名和口令登录系统，如图 3-4 所示。

图 3-3　确认与远程系统建立连接

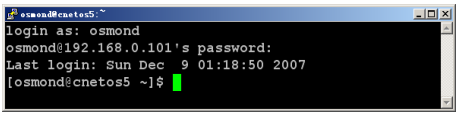

图 3-4　登录远程系统

至此用户已经通过 putty 成功登录到服务器，下面用户就可以如同在服务器控制终端上一样执行各种命令进行系统管理了。

3.1.4　系统运行级别与关机

系统运行级别

Linux 系统在任何时候都运行在一个指定的运行级上，并且不同的运行级的程序和服务各不相同，所要完成的工作和所要达到的目的也不一样。CentOS 设置了如表 3-1 所示的运行级，并且系统可以在这些运行级别之间进行切换，以完成不同的工作。

表 3-1　Linux 系统的运行级及说明

运　行　级	说　　明
0	所有进程将被终止，机器将有序的停止，关机时系统处于这个运行级别
1	单用户模式。用于系统维护，只有少数进程运行，同时所有服务也不启动
2	多用户模式。和运行级别 3 一样，只是网络文件系统（NFS）服务没被启动
3	多用户模式。允许多用户登录系统，是系统默认的启动级别
4	留给用户自定义的运行级别
5	多用户模式，并且在系统启动后运行 X-Window，给出一个图形化的登录窗口
6	所有进程被终止，系统重新启动

如果系统启动后进入字符登录界面，则说明系统默认的运行级别为 3；如果系统启动后进入图形登录界面，则说明系统默认的运行级别为 5。

如果已经启动了字符界面，用户想要进入图形界面可以使用如下命令切换：

```
$  startx &
```

查看和切换运行级

用户可以使用如下的命令查看当前系统的运行级：

```
$ runlevel
```

用户可以使用如下的命令切换运行级：

```
# init [0123456Ss]
```

在 init 命令后有一个参数，此参数是要切换到的运行级的代号，如：

- init 0 命令表示切换至运行级别 0，即关机。
- init 1 命令表示切换至运行级别 1，即进入单用户运行模式。
- init 6 命令表示切换至运行级别 6，即重新启动。

你也可以使用 telinit 命令，在 CentOS 中，telinit 命令是 init 命令的符号链接。

下面看一个使用 runlevel 和 init 命令的例子。

```
## 显示系统当前运行级别
#  runlevel
N 3
## 系统当前的运行级别为“3”，没有上一次运行级别（用“N”表示）
#  init 2
## 执行“init 2”命令后会在系统控制台中显示相应的停止启动服务信息
#  runlevel
3 2
## 系统当前运行级别已经为“2”，上一次的运行级别为“3”，转换运行级别成功
```

关机与重启命令

系统的关机和重新启动，实际上是进行运行级别的切换。此时可以使用 init 命令进行关机和重启，命令 init 用于立即关机或重启，但是在多用户系统中，若想给用户发送关机警告信息以便各个用户完成自己的工作并注销登录，则必须使用 shutdown、halt 和 reboot 等命令。

在多用户环境下，通常使用 shutdown 命令关闭和重新启动系统。shutdown 命令能够以一种比较安全的方式来关闭系统，所有登录到系统上的用户将被通知系统将要关闭，而新的登录操作将被阻止；同时所有的进程也将被通知系统将要关闭，这样有些程序，如 Vi 就能够及时保存用户编辑的文件并退出。

shutdown 命令的格式为：

```
shutdown [参数] time [warning-message]
```

其中：

- **time** 用于设置多长时间后执行 shutdown 指令，可以使用如下三种格式。
  - hh:mm ： 指定绝对时间，hh 用于指定小时，mm 用于指定分钟。
  - +m ： 指定相对时间，m 为数字，单位为分钟。
  - now ： 现在立刻进行，相当于+0。
- warning-message 用于设置发给用户的警告信息。

常用参数如下：

- **-t sec** ： 送出警告信息和删除信息之间要延迟多少秒再通知 init 执行运行级切换。
- **-k** ： 并不真的关闭系统，只是给每个用户发送警告信息。
- **-r** ： 关闭之后重新启动系统。
- **-h** ： 关闭之后停止系统。
- **-f** ： 重新启动后不用 fsck 检查磁盘。
- **-F** ： 重新启动后强制用 fsck 检查磁盘。

请见如下命令示例。

```
## 警告所有用户系统将在 5 分钟后重新启动系统
# shutdown -r +5 "System will be reboot in 5 ms, Please save your work.”
## 立即关闭系统
# shutdown -h now
## 立即重新启动系统，并在重新启动后强制用 fsck 检查磁盘
# shutdown -h -F now
```

3.2　Shell 和命令操作基础

内容提要

1．熟悉 Shell 及其功能。
2．熟悉 Shell 的元字符。
3．熟悉命令行格式。
4．学会使用命令帮助。

3.2.1　Shell 简介

什么是 Shell

- Shell 是系统的用户界面，提供了用户与内核进行交互操作的一种接口（命令解释器）。
- Shell 接收用户输入的命令并把它送入内核去执行。
- Shell 起着协调用户与系统的一致性和在用户与系统之间进行交互的作用。

Shell 在 Linux 系统中具有极其重要的地位，如图 3-5 所示。

图 3-5　Shell 在 Linux 系统中的地位

Shell 的功能

Shell 最重要的功能是命令解释，从这种意义上说，Shell 是一个命令解释器。Linux 系统中的所有可执行文件都可以作为 Shell 命令来执行。Linux 系统上可执行文件的分类见表 3-2。

表 3-2　Linux 系统中的可执行文件

类　别	说　明
Linux 命令	存放在/bin、/sbin 目录下的命令
内置命令	出于效率的考虑，将一些常用命令的解释程序构造在 Shell 内部
实用程序	存放在/usr/bin、/usr/sbin、/usr/share、/usr/local/bin 等目录下的实用程序或工具
用户程序	用户程序经过编译生成可执行文件后，也可作为 Shell 命令运行
Shell 脚本	由 Shell 语言编写的批处理文件

图 3-6 描述了 Shell 是如何完成命令解释的。

当用户提交了一个命令后，Shell 首先判断它是否为内置命令，如果是就通过 Shell 内部的解释器将其解释为系统功能调用并转交给内核执行；若是外部命令或实用程序就试着在硬盘中查找该命令并将其调入内存，再将其解释为系统功能调用并转交给内核执行。在查找该命令时有两种情况：

1．如果用户给出了命令的路径，Shell 就沿着用户给出的路径进行查找，若找到则调

入内存，若没找到则输出提示信息。

2．如果用户没有给出命令的路径，Shell 就在环境变量 PATH 所制定的路径中依次查找命令，若找到则调入内存，若没找到则输出提示信息。

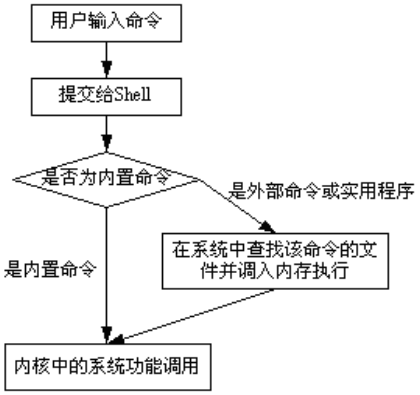

图 3-6　命令解释过程

提示：

1．内置命令是包含在 Shell 自身当中的，在编写 Shell 的时候就已经包含在内了，当用户登录系统后就会在内存中运行一个 Shell，由其自身负责解释内置命令。一些基本的命令如 cd、exit 等都是内置命令。用 help 命令可以查看内置命令的使用方法。

2．外部命令是存在于文件系统某个目录下的具体的可执行程序，如文件拷贝命令 cp，就是在/bin 目录下的一个可执行文件。用 man 或 info 命令可以查看外部命令的使用方法。外部命令也可以是某些商业或自由软件，如 mozilla 等。

此外，Shell 还具有如下的一些功能：

- 通配符
- 命令补全、别名机制、命令历史
- 重定向
- 管道
- 命令替换
- Shell 编程语言

Shell 的主要版本

表 3-3 列出了几种常见的 Shell 版本。

表 3-3　常见的 shell 版本

版　本	说　明
Bourne Again Shell（bash, bsh 的扩展）	bash 是大多数 Linux 系统的默认 Shell。bash 与 bsh 完全向后兼容，并且在 bsh 的基础上增加和增强了很多特性。bash 也包含了很多 C Shell 和 Korn Shell 的优点。bash 有很灵活和强大的编程接口，同时有很友好的用户界面
Korn Shell （ksh）	Korn Shell（ksh）由 Dave Korn 编写。它是 UNIX 系统上的标准 Shell。另外，在 Linux 环境下有一个专门为 Linux 系统编写的 Korn Shell 的扩展版本，即 Public Domain Korn Shell（pdksh）
tcsh （csh 的扩展）	是 C Shell 的扩展。tcsh 与 csh 完全向后兼容，但它包含了更多使用户感觉方便的新特性，其

	性能上最大的提高是在命令行编辑和历史浏览方面

Shell 元字符

在 Shell 中有一些具有特殊的意义字符，称为 Shell 元字符（Shell Metacharacters）。若不以特殊方式指明，Shell 并不会把它们当做普通文字符使用。

表 3-4 简单介绍了常用的 Shell 元字符的含义。

表 3-4　常用的 shell 元字符及含义

Shell 元字符	Shell 元字符的含义
*	代表任意字符串
?	代表任意字符
/	代表根目录或作为路径间隔符使用
\	转义字符。当命令的参数要用到保留字时，要在保留字前面加上转义字符
\<Enter>	续行符。可以使用续行符将一个命令行分写在多行上
$	变量值置换，如：$PATH 表示环境变量 PATH 的值
'	在'…'中间的字符都会被当做文字处理，指令、文件名、保留字等都不再具有原来的意义
"	在"…"中间的字符会被当做文字处理并允许变量值置换
`	命令替换，置换`…`中命令的执行结果
<	输入重定向字符
>	输出重定向字符
|	管道字符
&	后台执行字符。在一个命令之后加上字符“&”，该命令就会以后台方式执行
;	分割顺序执行的多个命令
( )	在子 Shell 中执行命令
{ }	在当前 Shell 中执行命令
!	执行命令历史记录中的命令
~	代表登录用户的宿主目录（自家目录）

3.2.2　命令操作基础

目录和文件名的命名规则

在 Linux 下可以使用长文件或目录名，也可以给目录和文件取任何名字，但必须遵循下列的规则：

- 除了/之外，所有的字符都可以用于目录和文件名。
- 有些字符最好不用，如空格符、制表符、退格符和字符：？，@#$&()\|；"'<>等。
- 避免使用+、－或.来作为普通文件名的第一个字符。
- 大小写敏感。
- 以“.”开头的文件或目录是隐含的。

命令基本格式

Shell 命令的一般格式为：

```
cmd [options] [arguments]
```

其中：

- cmd 是命令名。
- options 是选项。
- arguments 是参数，也即操作对象。

说明：

- 最简单的 Shell 命令只有命令名，复杂的 Shell 命令可以有多个选项和参数。
- 选项和参数都作为 Shell 命令执行时的输入，它们之间用空格分隔开。
- 单字符参数前使用一个减号（-），单词参数前使用两个减号（--）。
- 多个单字符参数前可以只使用一个减号。
- 操作对象（arguments）可以是文件也可以是目录，有些命令必须使用多个操作对象，如 cp 命令必须指定源操作对象和目标操作对象。
- 并非所有命令的格式都遵从以上规则，例如 dd、find 等。

以下是一些命令示例：

```
$ ls
$ ls -lRa  /home
$ cat  abc  xyz
$ ls --help
```

具有以上格式的字符串习惯地称为命令行，命令行是用户与 Shell 之间对话的一个基本单位。

通配符

通配符主要为了便于用户描述目录或文件而使用。常用的通配符如下。

- *：匹配任何字符和任何数目的字符。
- ?：匹配单一数目的任何字符。
- [ ]：匹配[ ]之内的任意一个字符。
- [!]：匹配除了[!]之外的任意一个字符，!表示非的意思。

> **警告：**1．“*”能匹配文件或目录名中的“.”。
>
> 2．“*”不能匹配首字符是“.”的文件或目录名。

通配符在指定一系列的文件名时非常有用，下面列举一些例子来说明通配符的使用。

- ls *.c ： 列出当前目录下的所有 C 语言源文件。
- ls /home/*/*.c ： 列出/home 目录下所有子目录中的所有 C 语言源文件。
- ls n*.conf ： 列出当前目录下的所有以字母 n 开始的 conf 文件。
- ls test?.dat ： 列出当前目录下的以 test 开始的，随后一个字符是任意的.dat 文件。
- ls [abc]* ： 列出当前目录下的首字符是 a 或 b 或 c 的所有文件。
- ls [!abc]* ： 列出当前目录下的首字符不是 a 或 b 或 c 的所有文件。
- ls [a-zA-Z]* ： 列出当前目录下的首字符是字母的所有文件。

3.2.3　获得命令帮助

使用 man 命令获得帮助

在系统中，用户可以非常容易的获得系统地帮助和支持，系统发行版本中为几乎每个

程序、工具、命令或系统调用编制了使用手册。要想查看某个命令的使用手册页，只要输入 man 后面跟该命令的名称即可。例如，输入如下命令将显示如图 3-7 所示的界面。

```
$ man ls
```

在此界面中可以查看有关 ls 命令的详细使用说明。用户可以使用↑、↓和 PgDn、PgUp 键进行翻阅，按 q 键退出。

一般来说，命令的使用手册页中会包括如表 3-5 中所示的组成信息。

表 3-5　命令使用手册页中的组成

手册项目	说　明
Name	命令的名称及简单说明
Synopsis	如何使用这个命令即命令行选项
Description	对这个命令及其选项的解释
Files	这个命令用到的文件清单和它们存放的位置
See Also	有关的使用手册页的清单
Diagnostics	特殊输出情况的说明
Bugs	编程漏洞
Author	命令程序的主要编写者和其他维护人员

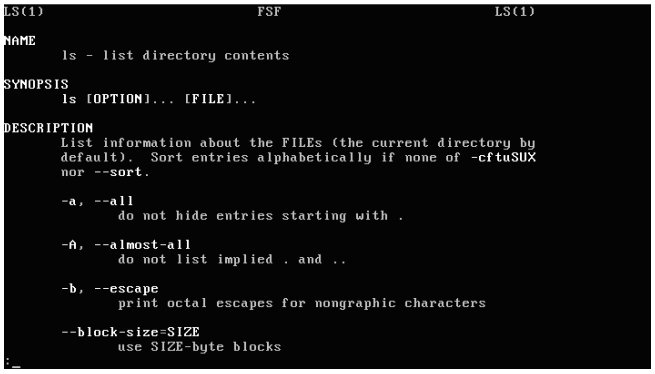

图 3-7　使用 man 获得命令帮助

另外，根据内容的不同可将手册页分为不同的类型，不同类型用一个数字（或字母）代表，各种类型的含义如表 3-6 所示。

表 3-6　命令使用手册页的类型

手册类型	说　明
man1	普通用户的可执行命令手册
man2	系统调用手册，内核函数的说明
man3	子程序手册，库函数的说明
man4	系统设备手册，“/dev”目录中设备文件的参考说明
man5	配置文件格式手册，大多为“/etc”目录下各种配置文件的格式描述
man6	游戏和趣味小程序的说明手册
man7	协议转换手册，也包括一些杂项
man8	系统管理工具手册，这些命令只有超级用户才可以执行

man9	Linux 系统例程手册

手册页按照不同的类型被存放在系统不同的目录下（/usr/share/man/man[1..9]）。例如，使用如下命令可以查看 passwd 命令的使用方法。

```
$ man 1 passwd
```

使用如下命令可以查看 passwd 配置文件的描述信息

```
$ man 5 passwd
```

使用 info 命令获得帮助

texinfo 是 Linux 系统中提供的另一种格式的帮助信息，它与手册页相比具有更强的交互性，如支持连接跳转功能等。通常使用 info 命令查看 texinfo 格式的帮助文档（info 文档存放在“/usr/share/info/”目录中）。例如，输入如下命令将显示如图 3-7 所示的界面。

```
$ info ls
```

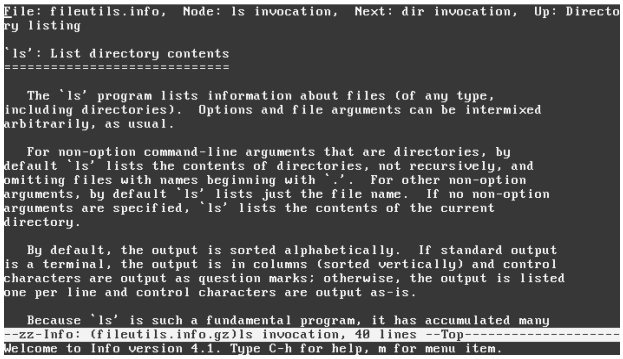

图 3-7　使用 info 获得命令帮助

用户可以使用↑、↓和 PgDn、PgUp 键进行翻阅，按 q 键退出。另外，用户可以使用 Ctrl+H 键进入 info 命令的帮助屏幕，学习 info 命令的更详细的使用方法。

3.3　文件概述

内容提要

1. 理解 Linux 系统中文件的概念。
2. 掌握 Linux 系统中文件的分类。
3. 熟悉 Linux 系统中的设备文件。
4. 理解 Linux 系统中的链接文件。

3.3.1　什么是文件

在 Linux 系统上，文件被看做是字节序列。这种概念使得所有的系统资源有了统一的标识，这些资源包括：普通文件或目录、磁盘设备、控制台（键盘、显示器）、打印机等。对这些资源的访问和处理都是通过字节序列的方式实现的。

3.3.2 文件的类型

Linux 系统下的文件类型包括：

- 普通文件（ - ）
- 目录（ d ）
- 符号链接（ l ）
- 字符设备文件（ c ）
- 块设备文件（ b ）
- 套接字（ s ）
- 命名管道（ p ）

普通文件

普通文件仅仅就是字节序列，Linux 并没有对其内容规定任何的结构。普通文件可以是程序源代码（C、C++、Python、Perl 等）、可执行文件（文件编辑器、数据库系统、出版工具、绘图工具等）、图片、声音、图像等。Linux 不会区别对待这些文件，只有处理这些文件的应用程序才会对根据文件的内容为它们赋予相应的含义。

在 DOS 或 Winodws 环境中，所有的文件名的后缀就能表示该文件的类型，如：*.exe 表示可执行文件，*.bat 表示批处理文件。在 Linux 环境下，只要是可执行的文件并具有可执行属性它就能执行，不管其文件名后缀是什么。但是对一些数据文件一般也遵循一些文件名后缀规则。

- 系统文件
  - *.conf: 配置文件。
  - *.rpm: rpm 包。
  - *.deb: deb 包。
  - *.a: 一种存档文件。
  - *.lock: 一种锁定文件。
  - *~: 备份文件。
  - .*: 隐含文件。
- 程序与脚本
  - *.c: C 语言源程序文件。
  - *.cpp: C++语言源程序文件。
  - *.h: C 或 C++的头文件。
  - *.o: 程序对象文件。
  - *.pl: Perl 语言源程序文件。
  - *.php: PHP 语言源程序文件。
  - *.python: Python 语言源程序文件。
  - *.tcl: TCL 脚本文件。
  - *.so，*.lib: 库文件。
  - *.sql: SQL 语言文件。
- 其他格式文件

- *.txt: 无格式的 ASCII 码文件。
- *.html，*.htm: 静态 Web 页。
- *.ps: PostScript 文件。
- *.mp3: mp3 文件。
- *.au: 一种声音文件。
- *.wav: 一种声音文件。
- *.xpm: 一种图像文件。
- *.jpg: 一种图形、图像文件。
- *.gif: 一种图形、图像文件。
- *.png: 一种图形、图像文件。

目录和硬链接

目录文件是由一组目录项组成，目录项可以是对其他文件的指向也可以是其下的子目录指向。

实际上，一个文件的名称是存储在其父目录中的，而并非同文件内容本身存储在一起。

将两个文件名（存储在其父目录的目录项中）指向硬盘上一个存储空间，对两个文件中的任何一个的内容进行修改都会影响到另一个文件，这种链接关系称为硬链接，硬链接文件实际上就是在某目录中创建目录项，从而使不止一个目录可以引用到同一个文件。它可以由 ln 命令建立。首先查看一下目录中的文件情况：

```
$ ls -l
-rwxr-xr-x  1 Mike   users      58 2006-07-01 10:05 file1
$ cat file1
This is file1.
```

使用 ln 命令建立文件 file1 的硬链接文件 file2：

```
$ ln file1 file2
```

该命令产生一个新的文件 file2，它和已经存在的文件 file1 建立起硬链接关系：

```
$ cat file2
This is file1.
$ ls -l
-rwxr-xr-x  2 Mike   users      58 2006-07-01 10:05 file1
-rwxr-xr-x  2 Mike   users      58 2006-07-01 10:07 file2
```

可以看出，file2 和 file1 的大小相同，内容相同。再看详细信息的第 2 列，原来 file1 的链接数是 1，说明这一块硬盘存储空间只有 file1 一个文件指向它，而建立起 file1 和 file2 的硬链接关系之后，这块硬盘空间就有 file1 和 file2 两个文件同时指向它，所以 file1 和 file2 的链接数就都变为了 2。因为两个文件指向一块硬盘空间，所以如果现在修改 file2 的内容为“This is file2.”，再查看 file1 的内容，就会有：

```
$ cat file1
This is file2.
```

如果删除其中的一个文件（不管是哪一个），就是删除了该文件和硬盘空间的指向关系，该硬盘空间不会释放，另外一个文件的内容也不会发生改变，但是目录详细信息中的链接数会减少，请见如下信息。

```
$ rm -f file1
$ ls -l
```

```
-rwxr-xr-x  1 Mike   users      58 2006-07-01 10:07 file2
$ cat file2
This is file2.
```

硬链接并不是一种特殊类型的文件，只是在文件系统中允许多个目录项指向同一个文件。

符号链接

符号链接又称软链接，是指将一个文件指向另外一个文件的文件名。这种符号链接的关系由 ln -s 命令行来建立。首先查看一下目录中的文件信息：

```
$ ls -l
-rwxr-xr-x  1 Mike   users      58 2006-07-01 10:05 file1
$ cat file1
This is file1.
```

使用 ln 命令和-s 选项建立文件 file1 的符号链接文件 file2：

```
$ ln -s file1 file2
```

该命令产生一个新的文件 file2，它和已经存在的文件 file1 建立起符号链接关系：

```
$ cat file2
This is file1.
$ ls -l
-rwxr-xr-x  1 Mike   users      58 2006-07-01 10:15 file1
lrwxrwxrwx  1 Mike   users       5 2006-07-01 10:17 file2 -> file1
```

可以看出 file2 这个文件很小，因为它只是记录了要指向的文件名而已，请注意那个从文件 file2 指向文件 file1 的指针。

为什么 cat 命令显示的 file2 的内容与 file1 相同呢？因为 cat 命令在寻找 file2 的内容时，发现 file2 是一个符号链接文件，就根据 file2 记录的文件名找到了 file1 文件，然后将 file1 的内容显示出来。

明白了 file1 和 file2 的符号链接关系，就可以理解为什么 file1 的链接数仍然为 1，这是因为 file1 指向的硬盘空间仍然只有 file1 一个文件在指向。

如果现在删除了 file2，对 file1 并不产生任何影响；而如果删除了 file1，那么 file2 就因无法找到文件名称为 file1 的文件而成为死链接。

```
$ rm -f file1
$ ls -l
lrwxrwxrwx  1 Mike   users       5 2006-07-01 10:17 file2 -> file1
$ cat file2
cat: file2: No such file or directory
```

设备文件

设备是指计算机中的外围硬件装置，即除了 CPU 和内存以外的所有设备。通常，设备中含有数据寄存器或数据缓存器、设备控制器，它们用于完成设备同 CPU 或内存的数据交换。

在 Linux 下，为了屏蔽用户对设备访问的复杂性，采用了设备文件，即可以通过像访问普通文件一样的方式来对设备进行访问读写。

设备文件用来访问硬件设备，包括硬盘、光驱、打印机等。每个硬件设备至少与一个设备文件相关联。设备文件分为：字符设备（如：键盘）和块设备（如：磁盘）。Linux 下设备名以文件系统中的设备文件的形式存在。所有的设备文件存放在/dev 目录下。

表 3-7 列出了常用设备列表。

表 3-7　常用设备文件

设备文件	说　明
/dev/hd*	IDE 硬盘设备，如 hda1 表示第一块 IDE 硬盘的第一个分区；hdb2 表示第二块 IDE 硬盘的第二个分区
/dev/sd*	SCSI 硬盘设备，如 sda1 表示第一块 SCSI 硬盘的第一个分区；sdb2 表示第二块 SCSI 硬盘的第二个分区

续表

设备文件	说　明
/dev/lp*	表示并口设备，如 lp0 表示第一个并口设备；lp1 表示第二个并口设备
/dev/tty*	终端设备
/dev/console	系统控制台
/dev/scd*	SCSI 光驱设备
/dev/ppp*	ppp 设备
/dev/isdn*	isdn 设备
/dev/null	空设备
/dev/zero	零设备

在/dev 目录下有许多链接文件，使用这些链接能够方便地使用系统中的设备。例如，可以通过/dev/cdrom 而不是/dev/hdc 来访问光驱。

套接字和命名管道

套接字和命名管道是 Linux 环境下实现进程间通信（IPC）的机制。

命名管道（FIFO）文件允许运行在同一台计算机上的两个进程之间进行通信。套接字（socket）允许运行在不同计算机上的进程之间相互通信。

套接字和命名管道通常是在进程运行时创建或删除的，一般无需系统管理员干预。

3.4　文件与目录操作命令

内容提要

1．掌握常用的文件操作命令。

2．掌握常用的目录操作命令。

3.4.1　目录操作命令

ls 命令

- 功能说明：显示文件和目录列表。
- 命令格式：ls [参数] [<文件或目录> …]
- 常用参数：
  - -a：不隐藏任何以“.”字符开始的条目。
  - -b：用八进制形式显示非打印字符。
  - -R：递归列出所有子目录。
  - -d：当遇到目录时，列出目录本身而非目录内的文件，并且不跟随符号链接。
  - -F：在条目后加上文件类型的指示符号（*, /, =, @, |，其中的一个）。
  - -l：使用较长格式列出信息。
  - -L：当显示符号链接的文件信息时，显示符号链接所指示的对象而并非符号链接本身的信息。
  - -x：逐行列出项目而不是逐栏列出。

- -1：每行只列出一个文件。
- -r：依相反次序排列。
- -S：根据文件大小排序。
- -X：根据扩展名排序。
- -c：根据状态改变时间（ctime）排序。
- -t：根据最后修改时间（mtime）排序。
- -u：根据最后访问时间（atime）排序。

- 使用示例：

```
$ ls
$ ls -a
$ ls -F
$ ls -l
$ ls -R
$ ls -Sl
$ ls -rl
$ ls -cl
$ ls -tl
$ ls -ul
$ ls some/dir/file
$ ls some/dir/
$ ls -d some/dir/
```

tree 命令

- 功能说明：显示文件和目录树。
- 命令格式：tree [参数] [<目录>]
- 常用参数：
  - -a：不隐藏任何以“.”字符开始的条目。
  - -d：只显示目录不显示文件。
  - -f：每个文件都显示路径。
  - -F：在条目后加上文件类型的指示符号（*，/，=，@，|，其中的一个）。
  - -r：依相反次序排列。
  - -t：根据最后修改时间（mtime）排序。
  - -L n：只显示 n 层目录（n 为数字）。
  - --dirsfirst：目录显示在前文件显示在后。
- 使用示例：

```
$ tree
$ tree -d
$ tree -F
$ tree -L 3
$ tree /some/dir/
```

pwd 命令

- 功能说明：显示当前工作目录。
- 命令格式：pwd [参数]
- 常用参数：
  - -P：若目录是一个符号链接，将显示物理路径而非符号链接。
- 使用示例：

```
$ pwd
$ pwd -P
```

cd 命令

- 功能说明：切换目录。
- 命令格式：cd [参数] [<目录>]
- 常用参数：
  - -P：若目录是一个符号链接，将显示物理路径而非符号链接。
- 使用示例：

```
$ cd /some/dir/
$ cd -P Examples
$ cd
$ cd ~
$ cd ..
$ cd ../..
$ cd -
```

mkdir 命令

- 功能说明：创建目录。
- 命令格式：mkdir [参数] <目录>
- 常用参数：
  - -p：创建目录树，需要时创建上层目录，如目录已存在也不视作错误。
- 使用示例：

```
$ mkdir somedir/
$ mkdir -p some/path/dir/
```

rmdir 命令

- 功能说明：删除空目录。
- 命令格式： rmdir [参数] <目录>
- 常用参数：
  - -p：删除目录，然后尝试删除指定路径中的所有上层目录。例如：rmdir -p a/b/c 的效果等于 rmdir a/b/c a/b a。
- 使用示例：

```
$ rmdir somedir/
$ rmdir -p some/path/dir/
```

3.4.2 文件操作命令

touch 命令

- 功能说明：生成新的空文件或更改现有文件的时间戳。
- 命令格式：touch [参数] <文件> …
- 常用参数：
  - -a：只更改访问时间。
  - -m：只更改修改时间。
  - -t <STAMP>：使用[[CC]YY]MMDDhhmm[.ss]格式的时间而非当前时间。
  - -r <参考文件或目录>：使用指定文件的时间属性而非当前时间。

- 使用示例：

```
$ touch newfile
$ touch file
$ touch -a file
$ touch -m file
$ touch -t 200701311200 file
```

GNU/Linux 的文件有 3 种类型的时间戳：

- mtime: 最后修改时间（ls –lt）
- ctime: 状态改变时间（ls –lc）
- atime: 最后访问时间（ls –lu）

> **注意：**
>
> 1．ctime **并非文件创建时间。**
>
> 2．**覆盖一个文件会改变所有三类时间**：mtime、ctime **和** atime。
>
> 3．**改变文件的访问权限或拥有者会改变文件的** ctime **和** atime。
>
> 4．**读文件会改变文件的** atime。

cp 命令

- 功能说明：复制文件或目录。
- 命令格式：cp [参数] <源> <目标>
- 常用参数：
  - -a：等价于 –dpR。
  - -d：当复制符号链接的源文件时，目标文件也将创建符号链接且指向源文件所链接的原始文件。
  - -f：强制复制，不管目标是否存在。
  - -i：交互式复制，覆盖文件前需要确认。
  - -p：在复制文件过程中保留文件属性，包括属主、组、权限与时间戳。
  - -R，-r：递归地复制目录及目录内的所有项目。
  - -l：对源文件创建硬链接，而非复制文件，也可以使用 ln 命令进行。
  - -s：对源文件创建符号链接，而非复制文件，也可以使用 ln -s 命令进行。
  - -u：只有当源文件的修改时间（ctime）比目标文件更新时或目标尚不存在时才进行复制。
- 使用示例：

```
$ cp file1  file2
$ cp some/dir/file1   someother/dir/
$ cp some/dir/file1   someother/dir/file2
$ cp some/dir/file .
$ cp some/dir/files  someother/dir/
$ cp some/dir/file1   some/dir/file2  some/dir/file3  someother/dir/
$ cp -r  some/dir/ someother/dir/
$ cp -au  some/dir/ someother/dir/
```

mv 命令

- 功能说明：移动文件或目录、文件或目录改名。

- 命令格式：mv [参数] <源> <目标>
- 常用参数：
  - -f：强制移动，不管目标是否存在。
  - -i：交互式移动，覆盖文件前需要确认。
  - -u：只有当源文件的修改时间（ctime）比目标文件更新时或目标尚不存在时才进行移动。
- 使用示例：

```
$ mv /some/dir/file1 /someother/dir/
$ mv /some/dir/file1 /someother/dir/file2
$ mv /some/dir/files /someother/dir/
$ mv file newname_file
$ mv dir newname_dir
```

rm 命令

- 功能说明：删除文件或目录。
- 命令格式：rm [参数] <文件> …
- 常用参数：
  - -f：略过不存在的文件，不显示任何信息。
  - -i：进行任何删除操作前必须先确认。
  - -r，-R：递归删除该目录下的所有目录层。
- 使用示例：

```
$ rm /some/dir/file1
$ rm -i /some/dir/file1
$ rm -f /some/dir/file1
$ rm -rf /some/dir/
```

注意：

1．默认时，rm 不会删除目录。使用-recursive（-r 或–R）选项可删除每个给定的目录，以及其下所有的内容。

2．要删除第一个字符为“-”的文件（例如“-foo”），请使用以下其中一种方法：

```
$ rm -- -foo
$ rm ./-f
```

ln 命令

- 功能说明：创建链接文件。
- 命令格式：ln [参数] <被链接的文件> <链接文件名>
- 常用参数：
  - -s：创建符号链接，而非硬链接。
  - -f：强行创建链接，不论其是否存在。
  - -i：覆盖原有文件之前先询问用户。
- 使用示例：

```
$ ln somefile hardlinkfile
$ ln -s somefile softlinkfile
$ ln -s somedir softlinkfile
```

symlinks 命令

- 功能说明：检查目录中的符号链接，并显示符号链接类型。
- 命令格式：symlinks [参数] <目录> [<目录>…]
- 链接类型：
  - absolute：使用绝对路径的符号链接。
  - dangling：原始文件已经不存在的符号链接。
  - lengthy：符号链接的路径中包含了多余的“../”。
  - messy：符号链接的路径中包含了多余的“/”。
  - other_fs：原始文件位于其他文件系统中。
  - relative：使用相对路径的符号链接。
- 常用参数：
  - -c：将 absolute/messy 类型的符号链接转换为 relative 类型。
  - -d：删除 dangling 类型的符号链接。
  - -r：检查目录下所有子目录中的符号链接。
  - -s：缩短 lengthy 类型的符号链接。
  - -v：显示所有类型的符号链接。
- 使用示例：

```
$ symlinks -v  -r .
$ symlinks -v /usr/bin
$ symlinks -c somedir
$ symlinks -s -d somedir
```

3.4.3　文件打包压缩命令

gzip 命令

- 功能说明：.gz 文件的压缩和解压缩程序。
- 命令格式：gzip [参数] <文件> …
- 常用参数：
  - -a: 使用 ASCII 文字模式。
  - -c: 把压缩后的文件输出到标准输出设备，不改动原始文件。
  - -d: 解开压缩文件。
  - -f: 强行压缩文件，不理会文件名称或硬链接是否存在以及该文件是否为符号链接。
  - -l: 列出压缩文件的相关信息。
  - -L: 显示版本与版权信息。
  - -n: 压缩文件时，不保存原来的文件名称及时间戳。
  - -N：压缩文件时，保存原来的文件名称及时间戳，这是默认的。
  - -q：不显示警告信息。
  - -r：递归处理，将指定目录下的所有文件及子目录一同处理。
  - -t：测试压缩文件是否正确无误。
  - -v：显示指令执行过程。
  - -V：显示 gzip 版本信息。

- -<压缩率>：压缩率是一个介于 1～9 的数值，默认值为“6”，数值越大压缩率越高。
- --best：此参数的效果和指定“-9”参数相同。
- --fast：此参数的效果和指定“-1”参数相同。

- 使用示例：

```
$ gzip filename
$ gzip -v file1 file2
$ gzip -c file1 file2 > foo.gz
$ gzip -l *.gz
$ gzip -d filename.gz
```

bzip2 命令

- 功能说明：.bz2 文件的压缩和解压缩程序。
- 命令格式：bzip2 [参数] <文件> …
- 常用参数：
  - -c：把压缩后的文件输出到标准输出设备，不改动原始文件。
  - -d：解开压缩文件。
  - -f：强行压缩文件。
  - -k：保留原始文件，默认在压缩或解压缩后会删除原始的文件。
  - -s：降低程序执行时内存的使用量，但会加长执行时间。
  - -t：测试压缩文件是否正确无误。
  - -v：显示指令执行过程。
  - -V：显示 bzip2 版本信息。
  - -<压缩等级>：压缩等级是一个介于 1～9 的数值，指定压缩时的区块大小。
  - --repetitive-best：若文件中有重复出现的资料时，可利用此参数提高压缩效果。
  - --repetitive-fast：若文件中有重复出现的资料时，可利用此参数加快执行速度。
- 使用示例：

```
$ bzip2 filename
$ bzip2 -vk file1 file2
$ bzip2 -c file1 file2 > foo.bz2
$ bzip2 -t *.bz2
$ bzip2 -d filename.bz2
```

> **提示：bzip2 以区块的方式来压缩文件，每个区块视为独立的单位。因此，当某一区块损坏时，便可利用 bzip2recover 命令试着将文件中的区块分隔开来，以便解压缩正常的区块。通常只适用于压缩文件很大的情况。**

tar 命令

- 功能说明：文件打包和解包。
- 命令格式：tar [参数] <目录> …
- 常用参数：
  - -f name：使用 name 指定存档文件名或设备名。
  - -v：列出处理的详细信息。

- -c：用于创建一个新的存档文件。
- -x：从归档文件中恢复备份文件。
- -t：用于列出一个存档文件中的文件名。
- -z：用 GNU 的 gzip 压缩文件或解压。
- -Z：用 compress 压缩文件或解压。
- -j：用 bzip2 压缩文件或解压。

- 使用示例：

```
$ tar -cvf myball.tar somedirname
$ tar -tf myball.tar
$ tar -xvf myball.tar
$ tar -zcvf myball.tar.gz somedirname
$ tar -ztf myball.tar.gz
$ tar -zxvf myball.tar.gz
$ tar -jcvf myball.tar.bz2 somedirname
$ tar -jtf myball.tar.bz2
$ tar -jxvf myball.tar.bz2
$ (cd /source/directory && tar cpf - . ) | (cd /dest/directory && tar xvfp -)
```

提示：

压缩文件（compressed file）和归档文件（archive file）的异同

- 相同：都是文件和目录的一个集合。
- 不同：
  - 归档文件所占用的磁盘空间是其中所有文件和目录的总和。
  - 一般情况下，压缩文件所占用的磁盘空间比其中所有文件和目录的总和要少。
  - 归档文件不是压缩文件，但是压缩文件可以是归档文件。

3.5　文本处理命令

内容提要

1．掌握常用的文本显示命令。

2．掌握常用的文本处理命令。

cat 命令

- 功能说明：用于从文件头到文件尾方向滚屏显示文本文件内容。
- 命令格式：cat [参数] [<文件> …]
- 常用参数：
  - -n：由 1 开始对所有输出的行进行编号。
  - -b：和-n 相似，只不过对于空行不编号。
  - -s：当遇到有连续两行以上的空行时，使用一个空行代替。
- 使用示例：

```
$ cat file
$ cat -n file
```

提示：

1．系统还提供了一个 tac 命令，用于从文件尾到文件头显示文件内容。

2．系统还提供了一个 rev 命令，与 tac 不同，它并不反转行序，而是把每行的内容反转。

3．可以使用 cat 命令连接多个文本文件，如：

```
$ cat file1 file2 > files
```

more 命令

- 功能说明：从文件头到文件尾分屏显示文本文件内容。
- 命令格式：more [参数] [<文件> …]
- 常用参数：
  - -d：显示提示信息“[Press space to continue, 'q' to quit.]”。
  - -s：当遇到有连续两行以上的空行时，使用一个空行代替。
  - +num：从第 num 行开始显示。
- 使用示例：

```
$ more file
$ more +10 file
```

提示：

1．系统还提供了一个 less 命令，用于双向显示分屏显示文本文件内容。

2．less 的功能比 more 丰富的多，可以使用如下命令查看其内置功能

```
$ less --help
```

head 命令

- 功能说明：显示文本文件的头部的若干行。
- 命令格式：head [参数] [<文件> …]
- 常用参数：
  - **-n**：显示前 n 行，不指定此参数显示前 10 行。
- 使用示例：

```
$ head file
$ head -5 file
```

tail 命令

- 功能说明：显示文本文件的尾部的若干行。
- 命令格式：tail [参数] [<文件> …]
- 常用参数：
  - -n：显示后 n 行，不指定此参数显示后 10 行。
  - +n：从第 n 行显示到文件尾。
  - -F：用于跟踪显示不断增长的文件结尾内容（通常用于显示日志文件）。
- 使用示例：

```
$ tail file
$ tail -5 file
```

```
$ tail +5 file
$ tail -F /var/log/messages
```

cut 命令

- 功能说明：纵向切割出文本指定的部分并写到标准输出。
- 命令格式：cut [参数] [<文件> …]
- 常用参数：
  - -b<LIST>：只列出<LIST>指定的字节。
  - -c<LIST>：只列出<LIST>指定的字符。
  - -f<LIST>：只列出<LIST>指定的字段；并打印所有不包含分界符的行，除非 -s 选项被指定。
  - -s：不打印没有包含分界符的行。
  - -d<DELIM>：DELIM 是分界符，使用指定<DELIM>代替制表符作为区域分界。
  - --complement：补足选中的字节、字符或字段的占位。
  - --output-delimiter=<STRING>：使用指定<STRING>作为输出分界符默认时采用输入的分界符。

提示：LIST 的语法

选用-b、-c 或-f 中一个或若干个选项时，每个<LIST>都由一个范围域或是由逗号分隔开的多个范围域组成。被选中的输入会按照与读入时相同的次序写到屏幕，每个输入只会被输出一次。每个范围域可以是以下中的任何一种：

- N：第 N 个字节、字符或字段，从 1 开始计数。
- N-：从第 N 个字节、字符或字段，直到行尾。
- N-M：从第 N 个到第 M 个已包含的字节、字符或字段。
- -M：从第一个到第 M 个字节、字符或字段。

- 使用示例：

```
$ cut -b-10 file
$ cut -c5- file
$ cut -c5-10,15-20 file
$ cut -f1,3,5 file
$ cut -f2-4 file
$ cut -f2-4 -d' ' file
$ cut -f1,2-4,6 -d' ' -s file
```

paste 命令

- 功能说明：纵向合并多个文本并写到标准输出。
- 命令格式：paste [参数] [<文件> …]
- 常用参数：
  - -d<DELIM>：DELIM 是分界符，使用指定<DELIM>代替制表符作为区域分界。
  - -s：不使用平行的行目输出模式，而是每个文件占用一行。
- 使用示例：

```
$ paste file1 file2
```

```
$ paste -s file1 file2
$ paste -d' ' file1 file2
```

sort 命令

- 功能说明：以行为单位对文件进行排序。
- 命令格式：sort [参数] [<文件> …]
- 常用参数：
  - -b：忽略前导的空格。
  - -d：只考虑空格、字母和数字。
  - -f：忽略字母的大小写。
  - -i：只考虑可打印字符。
  - -M：排序月份，（未知词）<“JAN”< … <“DEC”。
  - -n：根据字符串的数值进行排序。
  - -r：逆向排序。
  - -u：对相同的行只输出一行。
  - +n：n 为数字，对指定的列进行排序，+0 表示第 1 列，以空格或制表符作为列的间隔符。
- 使用示例：

```
$ sort file
$ sort -bd file
$ sort -bn file
$ sort -r file
$ sort -u file
$ sort +5 file
$ sort +5 -rb file
$ sort file1 file2
$ sort -br file1 file2
```

> **警告：1．本地环境变量会影响排序结果。**
>
> **2．如果希望以字节的自然值获得最传统的排序结果，请设定 LC_ALL=C。**

uniq 命令

- 功能说明：删除文本文件中相邻的重复的行并写到标准输出。
- 命令格式：uniq [参数] [<输入文件> [<输出文件>]]
- 常用参数：
  - -c：在每行前加上表示相应行目出现次数的前缀编号。
  - -d：只显示重复的行。
  - -i：忽略大小写差异。
  - -u：只显示出现一次的行。
  - -s<N>：<N>为数字，对各行前<N>个字符不作比较。
  - -w<N>：<N>为数字，对各行第<N>个字符以后的内容不作比较。
- 使用示例：

```
$ uniq file
$ uniq -i file
$ uniq -cd file
$ uniq -u file
```

wc 命令

- 功能说明：统计指定文本文件的行数、字数、字符数。
- 命令格式：wc [参数] [<文件> …]
- 常用参数：
  - -c：统计输出字节数。
  - -l：统计输出行数。
  - -L：统计输出最长一行的长度。
  - -w：统计输出单词数。
- 使用示例：

```
$ wc file
$ wc -l file
$ wc -w file
$ wc -c file
$ wc -L file
```

expand 命令

- 功能说明：将文件中的制表符转换为空格，写到标准输出。
- 命令格式：expand [参数] [<文件> …]
- 常用参数：
  - -i：不转换非空格后的制表符。
  - -t<N>：设定每个制表符为指定<N>的宽度，而不是默认的 8。
- 使用示例：

```
$ expand file
$ expand -t4 file
```

unexpand 命令

- 功能说明：将文件中的空格转换为制表符，写到标准输出。
- 命令格式：unexpand [参数] [<文件> …]
- 常用参数：
  - -a：转换所有空格字符而不仅仅是字母首部的空格。
  - --first-only：只转换首部的空格字符序列（覆盖-a 选项）。
  - -t<N>：设定每个制表符为指定<N>的宽度，而不是默认的 8（激活-a 选项）。
- 使用示例：

```
$ unexpand file
$ unexpand -t4 file
```

iconv 命令

- 功能说明：将文件从一种编码转换成另一种编码。
- 命令格式：iconv [参数] <输入文件>
- 常用参数：
  - -f <encoding>：指定原始文本编码。
  - -t <encoding>：指定要转换的编码。
  - -l：列出所有已知编码字符集。
  - -c：忽略输出中的无效字符。

- -o <output file>：指定输出文件，而不是在标准输出上显示。

- 使用示例：

```
$ iconv -l
$ iconv -f ISO-8859-1 -t UTF-8 -o outputfile inputfile
$ iconv -f GB2312 -t UTF-8 -o outputfile inputfile
$ iconv -f GBK -t UTF-8 -o outputfile inputfile
$ iconv -f BIG5 -t UTF-8 -o outputfile inputfile
$ iconv -f UTF-8 -t GB2312 -o outputfile inputfile
```

dos2unix 命令

- 功能说明：将 DOS 格式的文本文件转换成 UNIX 格式的文本文件。
- 命令格式：dos2unix [参数] <文件> [<输出文件>]
- 常用参数：
  - -k：不改变文件的时间戳。
  - -n：新文件模式，即不改变原文件将转换结果保存到指定的输出文件。
- 使用示例：

```
$ dos2unix dosfile
$ dos2unix -n dosfile linuxfile
$ dos2unix -k *
$ dos2unix -k -n dosfile linuxfile
```

> 提示：系统还提供了一个 unix2dos 命令，用于将 UNIX 格式的文本文件转换成 DOS 格式的文本文件。使用方法与 dos2unix 命令一样。

3.6　信息显示命令

内容提要

掌握常用的信息显示命令的使用。

uname 命令

- 功能说明：显示系统信息。
- 命令格式：uname [参数]
- 常用参数：
  - -a：显示全部的信息。
  - -s：显示内核名称（kernel name）。
  - -r：显示内核版本（kernel release）。
  - -v：显示内核版本类型及发布时间（kernel version）。
  - -m：显示计算机系统架构类型，等同于 arch 命令。
  - -n：输出网络节点上的主机名。
  - -o：显示操作系统名称。
- 使用示例：

```
$ uname
$ uname -o
$ uname -r
$ uname -m
$ uname -a
```

hostname 命令

- 功能说明：显示与主机名相关的信息。
- 命令格式：hostname [参数]
- 常用参数：

- -f：显示 FQDN (Fully Qualified Domain Name)。
- -d：显示 DNS 域名，等同于 dnsdomainname 命令。
- -i：显示主机名对应的 IP 地址。

- 使用示例：

```
$ hostname
$ hostname -f
$ hostname -i
```

dmesg 命令

- 功能说明：显示开机信息。
- 命令格式：dmesg [参数]
- 常用参数：-c：显示信息后清除 ring buffer 中的内容
- 使用示例：

```
$ dmesg
```

> **提示：**
>
> 1．kernel 会将开机信息存储在 ring buffer 中。
>
> 2．若开机时来不及查看信息，可利用 dmesg 来查看。
>
> 3．开机信息亦保存在/var/log/dmesg 中。

uptime 命令

- 功能说明：显示从开机到当前的时间。
- 命令格式：uptime
- 使用示例：

```
$ uptime
```

file 命令

- 功能说明：显示文件类型。
- 命令格式：file [参数] [<文件或目录> …]
- 常用参数：
  - -z：探测压缩过的文件类型。
  - -L：直接显示符号链接所指向的文件类型。
  - -f namefile：从文件 namefile 中读取要分析的文件名列表，让 file 依次辨识这些文件的类型。namefile 的格式为每行一个文件名。如果要检查标准输入，使用 - 作为 namefile 参数。
  - -v：在标准输出后显示版本信息。
- 使用示例：

```
$ file ~/.bashrc
$ file /usr/bin/passwd
$ file -L /bin/sh
$ file -z somefile.gz
$ file /etc/passwd /bin/ls /dev/{tty0,hda}
```

stat 命令

- 功能说明：从 inode 中提取并显示文件状态信息或文件系统信息。
- 命令格式：stat [参数] [<文件> …]
- 常用参数：
  - -f：显示文件系统信息。
  - -t：使用简洁格式输出信息。
  - -L：跟随链接。
- 使用示例：

```
$ stat /etc/passwd
$ stat -f /home
```

du 命令

- 功能说明：统计文件的磁盘用量，目录取总用量。
- 命令格式：du [参数]
- 常用参数：
  - -c：显示所有项目相加后的总用量。
  - -h：使用人类习惯的方式显示。
  - -s：只分别计算命令列中每个参数所占的总用量。
  - -S：不包括子目录的占用量。
  - -x：忽略不同文件系统上的目录。
  - --max-depth=N：值统计 N 层目录，N 为整数。
- 使用示例：

```
$ du
$ du -h
$ du -sS
$ du -sc /root /home
# du --max-depth=3 -x / |sort -n
```

df 命令

- 功能说明：查看磁盘剩余情况。
- 命令格式：df [参数]
  - -h：以人类容易理解的格式显示。
  - -i：显示 inode 的使用量而非块的使用量。
  - -l：仅显示本地的文件系统。
  - -t <fstype>：仅显示指定类型文件系统的使用量。
  - -x <fstype>：不显示指定类型文件系统的使用量。
- 使用示例：

```
$ df -h
$ df -i
$ df -lh
$ df -h -t ext3
$ df -h -x ext3
```

top 命令

- 功能说明：实时显示进程任务。

- 命令格式：top [参数]
- 常用参数：
  - -c：显示每个进程的完整指令。
  - -u <user>：只显示指定用户账号的进程。
  - -i：忽略僵死进程。
  - -s：使用安全模式消除互动模式下的潜在威胁。
- 使用示例：

```
$ top -s
$ top -s -u www-data
```

free 命令

- 功能说明：显示内存使用状态。
- 命令格式：free [参数]
- 常用参数：
  - -b：以 Byte 为单位显示内存使用情况。
  - -k：以 KB 为单位显示内存使用情况。
  - -m：以 MB 为单位显示内存使用情况。
  - -s <sec>：持续观察内存使用状况，<sec>为时间间隔秒数。
  - -t：显示内存总和。
- 使用示例：

```
$ free
$ free -mt
$ free -s 5
```

w 命令

- 功能说明：显示登录用户。
- 命令格式：w [参数]
- 常用参数：-s：显示短信息。
- 使用示例：

```
$ w
$ w -s
```

date 命令

- 功能说明：显示和设置日期和时间。
- 命令格式：date [参数]
- 常用参数：
  - -R：以 RFC 2822 规范输出日期和时间。
  - -u：显示 UTC 时间。
  - -s <STRING>：设置日期和时间 。
- 使用示例：

```
$ date
$ date -R
```

cal 命令

- 功能说明：显示月历。
- 命令格式：cat [参数] [月] [年]
- 常用参数：
  - -m：使用周一作为一周的第一天。
  - -y：显示一年的日历。
  - -3：显示上月、当前月、下月的月历。
- 使用示例：

```
$ cal
$ cal -3
$ cal -y
$ cal -y 2005
$ cal 11 2007
$ cal 9 1752
```

提示：1752 年 9 月第 3 日起改用西洋新历（因为这时大部分的国家都采用了新历），有 11 天被去除，所以该月份的月历有些不同。在此之前使用的是西洋旧历。

which 命令

- 功能说明：在环境变量 PATH 设置的目录下查找指定文件的位置。
- 命令格式：which [参数] <文件> …
- 常用参数：-a 显示所有匹配的路径。
- 使用示例：

```
$ which find
$ which -a find
```

whereis 命令

- 功能说明：在特定目录中查找符合条件的文件（包括二进制文件、手册页文件、源码文件）。
- 命令格式：whereis [参数] <文件> …
- 常用参数：
  - -b：只查找二进制文件。
  - -m：只查找手册页文件。
  - -s：只查找源码文件。
  - -u：查找不包含指定类型的文件。
  - -B <directory>：只在指定的目录下查找二进制文件。
  - -M <directory>：只在指定的目录下查找手册页文件。
  - -S <directory>：只在指定的目录下查找源码文件。
- 使用示例：

```
$ whereis ls
$ whereis -m ls
```

locale 命令

- 功能说明：显示本地支持的语言系统信息。

- 命令格式：locale [参数]
- 常用参数：
  - -a：显示本地支持的语言系统。
  - -m：显示所有支持的语言编码系统。
- 使用示例：

```
$ locale
$ locale -a
```

提示：

1．用户可以通过设置语系相关的环境变量修改当前的语言系统。

2．超级用户也可以通过修改配置文件/etc/sysconfig/i18n 改变语言系统。

apropos 命令

- 功能说明：使用正则表达式搜索手册页名称和描述。
- 命令格式：apropos [参数] <正则表达式>
- 常用参数：
  - -e：对给出的关键字进行精确匹配。
  - -w：指定关键字中包含通配符。
- 使用示例：

```
$ apropos apache
$ apropos -w ls*
```

提示：whatis 命令与 apropos 命令的功能类似。

3.7 基本网络操作命令

内容提要

1．掌握常用的远程登录命令的使用。

2．掌握常用的文件传输命令的使用。

3．掌握常用的字符界面浏览器的使用。

telnet 命令

- 功能说明：用 Telnet 协议与另一个主机通信。
- 命令格式：telnet [参数] [<主机名[端口]>]
- 常用参数：
  - -8：允许使用 8 位字符资料，包括输入与输出。
  - -a：自动注册进入远程系统。
  - -d：设置调试触发器的初值为 TRUE。
  - -E：禁止转移字符功能。
  - -e escapechar：设置转移字符。
  - -l user：指定连接远程系统的用户名。

- -n tracefile：打开记录文件，记录跟踪信息。
- -S tos：为 telnet 连接设置 IP TOS（type-of-service）选项。

- 使用示例：

```
$ telnet 192.168.0.100
$ telnet localhost 25
```

提示：

1 .若只键入 telnet 并回车 ,将进入 Telnet 的交互模式。可以在交互提示符 telnet> 下输入“?”查看交互模式下所有可用命令的说明。

2．telnet 可发送除了“转义字符”（escape）的任何字符到远程主机上。因为“转义字符”字符在 telnet 中是客户机的一个特殊的命令模式，它的默认值是 Ctrl-]。

ssh 命令

- 功能说明：使用 SSH 协议登录远程主机的客户端。
- 命令格式：ssh [参数] [-l login_name] [hostname | [username@]hostname] [command]
- 常用参数：
  - -1：强制 ssh 使用协议版本 1。
  - -2：强制 ssh 使用协议版本 2。
  - -4：强制 ssh 使用 IPv4 地址。
  - -6：强制 ssh 使用 IPv6 地址。
  - -v：冗余模式。打印关于运行情况的调试信息。在调试连接、认证和配置问题时非常有用。
  - -q：安静模式。抑制所有的警告和讯息信息。只有严重的错误才会被显示。
- 使用示例：

```
$ ssh -l osmond 192.168.0.100
$ ssh osmond@192.168.0.100
$ ssh osmond@192.168.0.100 "ls ~"
```

scp 命令

- 功能说明：基于 SSH 协议在本地主机和远程主机之间复制文件。
- 命令格式：scp [[user@]host1:]file1 [...] [[user@]host2:]file2
- 使用示例：

```
$ scp localfile osmond@192.168.0.100:~/remotefile
$ scp osmond@192.168.0.101:remotefile  localfile
$ scp identity.pub osmond@192.168.0.100:.ssh/authorized_keys
```

ftp 命令

- 功能说明：ftp 字符界面客户端。
- 命令格式：ftp [参数] [主机名]
- 常用参数：
  - -p：使用被动模式（passive mode），这是默认值。
  - -i：在 mget 期间关闭交互模式。

- -n：不使用自动登录。
- -g：关闭本地主机文件名称支持特殊字符的扩充特性。
- -d：详细显示指令执行过程，便于排错或分析程序执行的情形。
- -v：显示远程服务器的所有响应信息。

- 使用示例：

```
$ ftp 192.168.0.100
```

ftp 的交互命令如表 3-8 所示。

表 3-8　ftp 的交互命令

交互命令	说　明
?	用来列出 ftp 子命令
! <shell-command>	执行本地 Shell 命令
pwd	显示远程主机上的当前目录
ls	使用 UNIX 命令列出当前远程目录的内容
cd	在远程主机中切换目录
lcd	在本地主机中切换目录
get	从远程主机当前目录下下载一个文件
mget	从远程主机当前目录下下载多个文件（文件名中可包含通配符）
put	上传一个文件到远程主机的当前目录
mput	上传多个文件到远程主机的当前目录
mkdir	在远程主机上创建目录
rmdir	删除远程主机上的目录
chmod	修改远程主机上文件或目录的权限
open	打开一个新的 FTP 连接
close	关闭 FTP 连接
bye	断开与远程主机的连接

lftp 命令

- 功能说明：一个功能强大的字符界面文档传输工具。
- 命令格式：lftp [参数] [主机名]
- 常用参数：
  - -p <port>：用于指定连接的端口。
  - -u <user>[,<pass>]：使用指定的用户名或口令进行 FTP 身份验证。
  - -e <commands>：执行命令后并不退出。
  - -c <commands>：执行命令后退出。
  - -f <script_file>：执行文件中的命令后退出。
  - -d：使用调试模式，便于排错或分析程序执行的情形。
- 使用示例：

```
$ lftp 192.168.0.100
```

- **lftp** 的交互命令

lftp 除了支持传统 ftp 的交互命令之外，还做了如下许多扩充。

1．命令别名和历史

- alias [<name> [<value>]]：定义或清除别名。
- history -w file|-r file|-c|-l [cnt]：显示、操作历史文件。

2．远程文件目录操作

- cat [-b] <files>：滚屏显示文件的内容。
- more <files>：分屏显示文件的内容。
- zcat <files>：滚屏显示.gz 文件的内容。
- zmore <files>：分屏显示.gz 文件的内容。
- mv <file1> <file2>：文件改名。
- rm [-r] [-f] <files>：删除文件。
- mrm <files>：删除文件（可用通配符）。
- du [opts] <dirs>：显示整个目录的容量。
- find [directory]：递归显示指定目录的所有文件（用于 ls －R 失效时）。

3．上传和下载

- get [opts] <rfile> [-o <lfile>]：下载文件，可以改名后存储在本地。
- mget [opts] <files>：下载多个文件。
- pget [opts] <rfile> [-o <lfile>]：多线程下载。
- reget rfile [-o lfile]：下载续传。
- put [opts] <lfile> [-o <rfile>]：上传文件，可以改名后存储在远程。
- mput [opts] <files>：上传多个文件。
- reput lfile [-o rfile]：上传续传。

4．连接会话和队列管理

- scache [<session_no>]：显示所有连接会话或切换至指定的连接会话。
- queue [opts] [<cmd>]：将命令置于队列等待执行。
- jobs [-v]：显示后台执行的作业。
- wait [<jobno>] | all：将后台进程换到前台执行（fg 是 wait 的别名）。
- kill all | <job_no>：删除后台作业。
- exit bg：退出 lftp 后，所有任务移至后台继续执行。

5．Cache 管理

- cache [SUBCMD]：管理 lftp 的 cache。
- [re]ls [args]：读取 cache 显示远程文件列表，rels 不读取 cache。
- [re]cls [opts] [path/][pattern] : cls 提供了比 ls 更丰富的列表功能。
- [re]nlist [<args>] : nlist 只显示文件名（没有颜色区分）。

6．站点镜像

- mirror [opts] [remote [local]]：用于实现站点镜像。

7．书签管理

bookmark [SUBCMD]：用于管理书签。

8．环境参数设置

set [opts] [<var> [<val>]]：用户设置 lftp 的环境参数。

提示：lftp 的功能

- 支持 ftp、ftps、http、https、hftp、fish 等传输协议；
- 支持 FXP（在两个 FTP 服务器之间传输文件）；
- 支持代理；
- 支持多线程传输；
- 支持传输队列（queue）；
- 支持书签；
- 类似 bash，它提供后台命令、nohop 模式、命令历史、命令别名、命令补齐和作业控制支持。
- 支持镜像（mirror）；

wget 命令

- 功能说明：wget 使用 HTTP 和 FTP 协议，支持代理服务器和断点续传，是基于控制台最强大的下载工具。
  - 不使用交互界面的 wget 可以在后台工作，这意味着使用者在下载过程中可以不登录，只需在开始和结束下载时登录。
  - wget 的递归功能允许其查看 HTML 文件和 FTP 目录树结构，并在本地建立与远程站点上相同层次关系的目录结构，是镜像网页的首选。
  - 在 wget 通过 FTP 下载时，具有文件名通配符匹配和目录递归镜像功能。wget 可以读出并储存 HTTP 和 FTP 站点给出的时间戳，从而判断远程文件的更新状况，这使得 wget 对 FTP 站点和 HTTP 站点的镜像同样出色。
  - 断点续传的功能使得 wget 在缓慢和不稳定的连接状态下表现依然出色。
  - 支持代理服务器的特性使得 wget 在使用中减小网络负载、加速下载以及配合防火墙使用成为可能。与此同时，支持被动 FTP 下载也是许多用户选择 wget 的原因之一。
- 命令格式：wget [参数] <URL>
- 常用参数：

1．启动选项

- -V：显示 wget 的版本。
- -h：显示 wget 的使用说明。
- -b：启动之后转入后台执行，日志文件写在当前目录下“wget-log”文件中。
- -e <COMMAND>：执行一个.wgetrc 里面的<COMMAND>指令。

2．日志文件与输入文件选项

- -o <FILE>：将命令的输出写入指定的<FILE>文件。
- -a <FILE>：将命令的输出以追加方式写入指定的<FILE>文件。

- -d：显示调试信息。
- -q：以安静模式执行（无输出）。
- -v：输出详细信息。
- -nv：关闭详细信息输出，但不是安静模式。
- -i <FILE>：从指定的<FILE>文件中读取 URL。
- -F：把输入文件视为 HTML 文件（与-i 参数同时使用）。
- -B <URL>：与-F 一同使用，优先考虑-i 所指定文件中的 URL。

3．下载选项

- -t <NUMBER>：当 wget 无法与服务器建立连接时，尝试连接<NUMBER>次，（0 表示无限制）。
- -O <FILE>：将下载的文件保存为指定的 <FILE>。
- -nc：不覆盖已有的文件。
- -c：续传文件。
- -N：不取回比本地旧的文件，只下载更新的文件。
- -S：显示服务器响应。
- -T <SECONDS>：设定响应超时的秒数为<SECONDS>。
- -w <SECONDS>：在两次尝试之间等待<SECONDS>秒。
- -Y：通过代理服务器进行连接。
- -Q <quota>：限制下载文件的总大小最多不能超过<quota>，单位为字节，可以使用 k，m 后缀。
- --limit-rate=<RATE>：限定下载传输率，单位为字节，可以使用 k,m 后缀。

4．目录选项

- -nd：不下载目录结构，把从服务器所有指定目录下载的文件都堆到当前目录里。
- -x：创建与远程完全一致的目录结构。
- -nH：不创建以目标主机域名为目录名的目录，将目标主机的目录结构直接下到当前目录下。
- -P <PREFIX>：将文件保存到目录 PREFIX/…。
- --cut-dirs=<NUMBER>：忽略<NUMBER>层远程目录。

5．HTTP 选项

- --http-user=<USER>：指定 HTTP 用户验证的用户。
- --http-passwd=<PASS>：指定 HTTP 用户验证的用户口令。
- --no-cache：禁用服务器端的数据缓存（默认情况下为允许）。
- --proxy-user=<USER>：指定 Proxy 使用者为<USER>。
- --proxy-passwd=<PASS>：指定 Proxy 使用者口令为<PASS>。
- -E：将所有类型为 application/xhtml+xml 或 text/html 的文档以.html 扩展名保存。
- -U <AGENT>：设定代理的名称为<AGENT>而不是 Wget/VERSION。

6．FTP 选项

- --ftp-user=<USER>：指定 FTP 用户。

- --ftp-passwd=<PASS>：指定 FTP 用户口令。
- --no-remove-listing：不移除临时的.listing 文件。
- --no-glob：关闭文件名的 globbing 机制（即，不使用通配符）。
- --no-passive-ftp：关闭默认的被动传输模式（即，使用主动传输模式）。
- --retr-symlinks：在递归时，将链接指向文件（而不是目录）。

7．使用递归方式获取选项

- -r：打开递归下载。
- -l <NUMBER>：指定最大递归深度为<NUMBER>（inf 或 0 代表无穷）。
- --delete-after：删除下载完毕的本地文件。
- -k：转换非相对链接为相对链接。
- -m：开启适合用来镜像的选项，等价于-N -r -l inf --no-remove-listing。
- -p：用以确保所有用于显示被下载网页的元素都被下载，如图像，声音以及网页中用到的样式表。

8．递归方式的允许与拒绝选项

- -A <LIST>：在<LIST>指定允许下载的扩展文件名（用逗号间隔）。
- -R <LIST>：在<LIST>指定禁止下载的扩展文件名（用逗号间隔）。
- -D <LIST>：在<LIST>指定允许下载的域名。
- --exclude-domains=<LIST>：在<LIST>指定禁止下载的域名。
- --follow-ftp：跟踪 HTML 文档中的 FTP 链接。
- --follow-tags=<LIST>：在<LIST>中指定用逗号分隔的被跟踪的 HTML 标签列表。
- --ignore-tags=<LIST>：在<LIST>中指定用逗号分隔的不被跟踪的 HTML 标签列表。
- -H：当递归时转到外部主机。
- -L：仅仅跟踪相对链接。
- -I <LIST>：在<LIST>中指定允许下载的目录列表。
- -X <LIST>：在<LIST>中指定禁止下载的目录列表。
- -np：不要追溯到父目录，即只下载目标站点指定目录及其子目录的内容。
- 使用示例：

1．下载单个文件。

```
$ wget http://osmond.cn/cdbe/CDBE.zip
```

2．以续传方式后台下载单个文件。

```
$ wget -bc http://ftp.hostrino.com/pub/centos/5.0/isos/i386/CentOS-5.0-i386-bin-1of6.iso
```

3．只下载单一 HTML 文件，确保影响着页面显示的所有元素均被下载，并重新建立链接。

```
$ wget -p -k http://osmond.cn/cdbe/manual/index.html
```

4．下载 http://osmond.cn/cdbe/manual/ 目录下的所有文件。

```
$ wget -p -np -r -k http://osmond.cn/cdbe/manual/index.html
```

5．在本地镜像网站 http://www.xyz.edu.cn 的内容。

```
$ wget -m -l4 -t0 http://www.xyz.edu.cn
```

6．只下载网站指定的目录，避免向远程主机的其他目录扩散，并拒绝下载 gif 和 jpg 文件。

```
$ wget -r -L -R gif,jpg  http://www.xyz.edu.cn/doc/
```

7．递归下载 http://www.xyz.edu.cn 下的 blog 和 wiki 目录并将文件后缀存为 html。

```
$ wget -r -k -p -np -E -I blog,wiki http://www.xyz.edu.cn
```

8．递归下载 http://ayo.freshrpms.net/redhat/9/i386/updates/RPMS 的所有文件到当前目录。

```
$ wget -r -nH  -nd http://ayo.freshrpms.net/redhat/9/i386/updates/RPMS
```

9．批量下载（首先将每个要下载文件的 URL 写一行，生成文件 download.txt）。

```
$ wget -i download.txt -o download.log
```

10．使用代理下载。

```
$ wget -Y -i download.txt -o download.log
```

提示：如何设置代理

- **在环境变量中设定代理**

```
$ export PROXY=111.111.111.111:8080
```

- **在 ~/.wgetrc 中设定代理**

```
http_proxy = 111.111.111.111:8080
ftp_proxy = 111.111.111.111:8080
```

w3m 命令

- 功能说明：字符界面浏览器。
- 命令格式：w3m [参数] [URL 或 filename]
- 常用参数：
  - -I <charset>：说明文档字符集。
  - -O <charset>：指定文档显示字符集。
  - -T <type>：指定文档 content-type。
  - -F：自动呈现 frame。
  - -dump：格式化后输出至 stdout。
  - --dump_source：格式化源码后输出至 stdout。
  - +<num>：将光标定位到 <num> 行。
  - -num：显示行号。
  - -help：显示使用帮助信息。
  - -version：显示 w3m 的版本。
- 使用示例：

```
$ w3m README.html
$ w3m http://www.google.com
$ cat foo.html | w3m -dump -T text/html >foo.txt
```

提示：

1．w3m 的更多用法参见 http://w3m.sourceforge.net/MANUAL.en.html

2．字符界面的浏览器除了 w3m 之外，还有 lynx、elinks 等。