高等院校计算机应用系列教材

Linux 编程

(第二版)(微课版)

刘文果　主　编

丁　凯　徐钦桂　钟雪峰　谭　伟　副主编

清华大学出版社

北京

内 容 简 介

本书将 Linux 系统编程应用与操作系统原理深度融合，先从 Linux C 案例程序出发，提出问题，引入操作系统的概念和原理，讨论解决问题的理论和方法，再从理论回归实践，分析和解决编程应用问题，破解传统上理论教学和实践教学脱节的问题，取得了很好的教学效果。本书主要内容包括 Linux 基本操作、Shell 编程、系统 I/O 编程、文件系统、进程控制原理、多进程并发编程、信号机制、线程概念、多线程并发编程、同步互斥概念、基于信号量与 P/V 操作解决同步互斥问题、经典同步问题、网络编程、并发网络应用编程等。本书安排了大量的程序示例、课后习题，旨在训练读者理论运用和解决问题的能力，精心设计了很多绘图，使抽象的概念、原理和技术看得见。

本书内容全面、结构合理、思路清晰、语言简洁、示例丰富。本书既可作为高等院校计算机类专业有关操作系统和 Linux 编程等课程的教材，又可作为 C 程序、嵌入式开发工程师的参考资料。

本书配套的电子课件、示例源代码、习题答案和实验指导可以到 http://www.tupwk.com.cn/downpage 网站下载，也可以扫描前言中的“配套资源”二维码获取。扫描前言中的“看视频”二维码可以直接观看教学视频。

图书在版编目(CIP)数据

Linux 编程：微课版 / 刘文果主编. —2 版.—北京：清华大学出版社，2024.4
高等院校计算机应用系列教材
ISBN 978-7-302-65807-8

Ⅰ. ① L… Ⅱ. ① 刘… Ⅲ. ① Linux 操作系统—程序设计－高等学校－教材 Ⅳ. ① TP316. 85

中国国家版本馆 CIP 数据核字(2024)第 055772 号

责任编辑：胡辰浩
封面设计：高娟妮
版式设计：妙思品位
责任校对：马遥遥
责任印制：刘海龙

出版发行：清华大学出版社
网　　址：https://www.tup.com.cn，https://www.wqxuetang.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-83470000　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市君旺印务有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：22.5　　**字　　数**：604 千字
版　　次：2019 年 1 月第 1 版　　2024 年 6 月第 2 版　　**印　　次**：2024 年 6 月第 1 次印刷
定　　价：79.00 元

产品编号：102743-01

前　言

Linux是一种技术先进、功能强大、性能优越、应用广泛的操作系统，也是当今大多数云计算、大数据平台的节点用操作系统。要掌握 Linux 系统原理和编程技术需要具备操作系统原理知识，而学习操作系统原理，又需要通过 Linux 编程来巩固和应用理论知识。以往这两方面的教学脱节严重，致使教学效果不及预期。

本书是作者从事多年有关操作系统课程教学研究与改革成果的结晶，针对过去理论原理和编程实践脱节的问题，将操作系统理论和 Linux 编程实践进行深度融合，以 Linux 系统编程为主线，纳入操作系统中的进程管理、线程机制、信号量与 P/V 操作、进程间通信、文件系统等部分内容，将理论和实践有机结合，要想熟练掌握操作系统与 Linux 编程，不仅要深入理解相关的概念和原理，还要用操作系统理论知识去分析问题，在 Linux 环境下编写系统和网络通信应用程序。

本书先介绍操作系统的操作使用、Shell 编程、文件管理操作，使读者获得初步的感知，然后介绍系统内部结构、原理和编程等内容，使学习过程自然而不唐突。对于文件系统、进程管理与控制、线程管理、进程间通信、网络编程，都是从学生看得见、摸得着的命令操作和 C 程序运行结果开始，提出问题，引发学生讨论，引入操作系统的概念、内部结构、理论原理和解决方案。书中的绘图使抽象的原理看得见，逐步引导学生用理论知识去解决更多、更复杂的应用问题。本教材于 2017 年开始应用于我校有关操作系统课程的教学，经过两年的完善，2019 年出版第一版，2023 年出版 Mooc 视频版。多年的教学实践表明，采用本书内容和教学方案，有效破解了多年来操作系统课程难教难学的问题。

本书既可作为有关操作系统课程的主要教材，又可独立作为有关操作系统实验或 Linux 系统编程的教材，书中提供了大量的微课视频、PPT 课件、示例源代码、习题答案、实验指导等教学资源。

本书内容全面、结构合理、思路清晰、语言简洁、示例丰富。每章的开头概述了本章的学习目标。每章的正文都结合所讲述的关键技术和难点，穿插了大量有价值的示例程序，安排了有针对性的思考和练习，帮助学生理解相关概念。每章末尾都安排了丰富的课后习题，培养学生分析和解决问题的能力。

在编写本书的过程中，我们参考了相关文献，在此向这些文献的作者深表感谢。由于我们水平有限，书中难免有不足之处，恳请专家和广大读者批评指正。我们的电话是 010-62796045，电子邮箱是 992116@qq.com。

本书配套的电子课件、示例源代码、习题答案和实验指导可以到 http://www.tupwk.com.cn/downpage 网站下载，也可以扫描下方的“配套资源”二维码获取。扫描下方的“看视频”二维码可以直接观看教学视频。

扫描下载

配套资源

扫一扫

看视频

作　者

2023 年 11 月

目　录

第 1 章

Linux 系统文件操作

本章主要介绍 Linux 系统的基本知识，包括 Linux 系统简介、Linux 系统的目录结构、文件类型、文件权限、Linux 命令格式以及文件目录的基本操作，为在 Linux 环境下进行编程打下基础。

本章学习目标：

- 了解 UNIX 与 Linux 系统的基本特点和发展历程
- 理解 Linux 系统的目录结构
- 掌握 Linux 系统的安装、启动、登录方法
- 掌握 Linux 文件属性和权限的概念
- 掌握 Linux 文件路径的概念和通配符的含义
- 掌握常用的 Linux 文件与目录操作命令
- 掌握 Linux 文件的打包及解包方法
- 理解 I/O 重定向和管道的功能及基本概念

1.1 UNIX/Linux 操作系统简介

1.1.1 UNIX 简介

1969 年，Bell Labs(贝尔实验室)的 Ken Thompson 和 Dennis Ritchie 出于兴趣开发了一种多用户、多任务、多层次的操作系统 UNIX，1971 年完成第一版的开发，实现了多任务管理、文件操作、网络通信功能。这一版本的 UNIX 性能卓越。

1973 年，Dennis Ritchie 创造了 C 语言，与 Ken Thompson 一起用 C 语言重写了 UNIX 的第三版内核，使代码维护和移植变得便利，UNIX 得到了科研机构与企业的大力支持，该版本后来逐渐形成了 UNIX AT&T System V Release 4(SVR4)和 BSD 两个版本系列：

- 加利福尼亚大学 Berkeley 分校于 1978 年开发出研究版本 BSD UNIX，于 1994 年开发出 4.4 BSD 版本，并成为现代 BSD 基本版本。
- AT&T 于 1983 年开发出商业版本 System V 版本 1，其后的 System V 版本 4(称为 SVR4)大获成功，基于 SVR4 造就了 IBM 的 AIX 和 HP 的 HP-UX。

UNIX 系统在金融、教育、科研、军事等领域获得广泛应用，成为大学师生研究、学习操作系统原理的首选实例系统。UNIX 的主要版本有以下几种。

(1) AIX：IBM 基于 SVR4 开发的一套 UNIX 操作系统，其性能高、安全性高、可靠性强，被广泛用于银行领域。

(2) Solaris：Sun Microsystems 于 1982 年推出基于 BSD UNIX 的 Sun OS，随后的新版本 Solaris 在接口上逐渐向 SVR4 靠拢，其性能高、处理能力强，具有 GUI，在高校、科研院所用得较多。

(3) HP-UX：惠普(HP)公司以 SVR4 为基础研发的类 UNIX 操作系统。

(4) IRIX：SGI 公司以 SVR4 与 BSD 延伸程序为基础研发的 UNIX 操作系统，具有很强的图形处理功能，在游戏设计中使用广泛的三维图形编程库 OpenGL 就基于此系统。

尽管 UNIX 系统具有技术先进、性能高、安全性高等优点，但 UNIX 的不同版本间并不兼容，这给应用开发带来极大负担。搭建 UNIX 系统也涉及非常昂贵的费用，计算机硬件、UNIX 系统、开发工具、应用软件都需要分别计费。通常，搭建一套带开发系统的 UNIX 工作站的费用达数十万元，很多用户负担不起，很多学校买不起。另外，UNIX 系统源代码不开放，这给学习、研究带来不便。UNIX 厂商间的恶性竞争削弱了 UNIX 系统的技术优势，而与此同时，微软公司的 Windows 操作系统却得到了迅速发展，占据了桌面领域的市场。

1.1.2 Linux 概述

为方便广大师生学习、研究 UNIX 系统，1991 年，芬兰的林纳斯·托瓦兹(Linus Torvalds)开发了一套多用户、多任务、多线程的类 UNIX 操作系统，即 Linux。Linux 继承了 UNIX 系统强大的功能和性能，采用与 UNIX 系统兼容的操作命令，兼容 UNIX 编程接口规范 POSIX。只要学会了操作使用 Linux，一般就可操作 UNIX 系统；只要掌握了 Linux 环境编程，就能在 UNIX 环境下进行编程开发。

Linux 系统运行于廉价的 PC 上，可免费使用，开放源代码，鼓励广大师生使用和开发 Linux 环境的配套软件，这样可使 Linux 环境的各种开发工具(如 gcc)和应用软件变得非常齐全，而且全部开放源代码、免费使用、免费升级。

2001 年，Linux 2.4 版本内核发布。2003 年，Linux 2.6 版本内核发布，使 Linux 逐渐成为一种成熟的操作系统，在很多关键领域得到应用。目前，常见的 Linux 内核版本有 Linux 2.4、Linux 2.6、Linux 3.2、Linux 4.6 等。

Linux 系统自 1991 年诞生以来，借助 Internet，通过世界各地编程爱好者的共同努力，已成为如今使用最多的一种类 UNIX 操作系统。Linux 可安装在各种计算机硬件设备上，如个人计算机、大型机、超级计算机、Android 手机、平板电脑、路由器，世界上运算最快的 10 台超级计算机全部运行 Linux 操作系统。目前主流大数据平台 Hadoop 的每个节点都是一个 Linux 系统。

Linux 系统开源、完全免费、安全性高、可靠性强、支持多种平台；它功能强大，其目录结构、基本命令与 UNIX 一致，可为未来学习 UNIX 系统打下良好的基础。不同厂商将 Linux 内核与外围实用程序和文档封装，提供安装界面和系统配置、管理工具等，形成发行系统，目前主要发行版本包括 Red Hat Enterprise、Fedora、Ubuntu 等。不同 Linux 发行版本都采用至今仍由 Linux 系统创始人林纳斯·托瓦兹维护的同一个内核版本，在某种发行版本下编写的源程序和可执行程序不加修改即可在另一种 Linux 发行版本下运行，完全克服了 UNIX 不同发行版本间的不兼容问题。

思考与练习题 1.1 UNIX 系统得以发展的主要原因是什么？进入 20 世纪 90 年代和 21 世纪后，阻碍 UNIX 进一步发展的原因又是什么？

思考与练习题 1.2　近年来，Linux 系统从诞生到得以广泛应用的原因是什么？

1.2 Linux 系统目录结构

图 1-1 是 Linux 系统目录结构，它与 UNIX(如 IBM AIX、Sun Solaris)系统的目录结构基本一致。与 Windows 系统不同，Linux 系统目录结构是一棵树，树根是/，每个文件和目录的路径都是以“/”开始的一条路径，如/home/can。Linux 系统没有盘符概念，除了根分区，其他硬盘分区的文件系统都挂载到某个以“/”开始的目录路径下。Linux 系统主要目录的路径、描述，以及 Windows 环境下的对等目录或设施如表 1-1 所示。

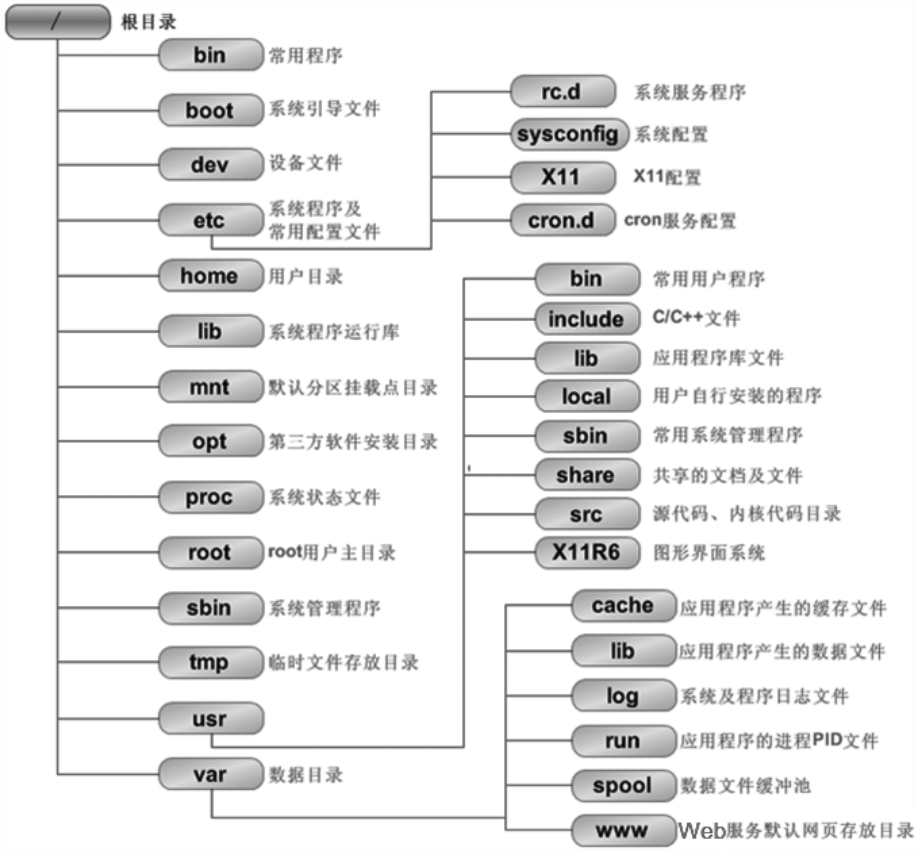

图 1-1　Linux 系统目录结构

Linux/UNIX 采用以上目录结构规范的好处有：

(1) 用户创建的文件全部放在/home 目录下，一来便于实施用户权限管理，二来可创建一个专门用于保存用户文件的分区，挂载到/home 目录下，方便管理；

(2) 可创建专用系统分区，保存 Linux 系统文件，以只读方式挂载在/usr 目录下，可防止恶意用户或病毒破坏系统目录，提高系统安全性；

(3) 可创建一个专用分区，保存动态增长的文件，以读写方式挂载到/var 目录下，万一该分区受

到破坏，整个系统将不受影响，从而提高系统可靠性；

(4) 这种目录结构规范来自 UNIX 系统，所有的 UNIX 和 Linux 采用的目录结构与上述规范大体相似，这使 UNIX/Linux 系统具有很好的向前兼容性，同时也方便人们学习。

表 1-1　Linux 目录结构说明

目录	描述	助记单词	Windows 系统对应目录或设施
/	根目录，所有的目录、文件、设备都在根目录/下，根目录/就是 Linux 文件系统的组织者，也是顶级目录		
/bin	bin 就是单词binary的英文缩写，含义是二进制。在一般的系统中，可以在这个目录下找到 Linux 常用的用户命令	binary (二进制)	C:\WINDOWS\system32
/boot	Linux 内核及系统引导过程所需的文件所在的目录，比如 vmlinuz initrd.img 文件就位于这个目录中。一般情况下，GRUB 或 LILO 系统引导管理器也位于这个目录中	boot	
/dev	dev 是单词device的英文缩写，这个目录中包含 Linux 系统中使用的所有设备文件，是访问这些外部设备的一个入口，使用户能够用操作文件的方法操作这些外部设备	device	
/etc	目录/etc 中存放各种系统配置文件，其中还有子目录，系统网络配置文件、文件系统、X 图形界面配置、设备配置、用户设置信息等都位于这个目录中	etcetera	注册表
/home	普通用户的家目录，如果建立一个用户，用户名是“xx”，那么在/home 目录下就有一条对应的/home/xx 路径，它是用来存放用户文件的家目录		C:\Documents and Settings
/include 或 /usr/include	C/C++源程序所需的系统头文件默认所在的目录		
/lib 或 /usr/lib	lib 是 library 的缩写。这个目录用来存放系统的库文件。几乎所有的应用程序都会用到这个目录中的库文件	library (库)	C:\WINDOWS\system32
/lost+found	在 ext2 或 ext3 文件系统中，当系统意外崩溃或机器意外关机而产生一些文件碎片时，将它们放在此目录下		
/mnt	该目录一般用于管理存储设备的挂载目录，比如/mnt/cdrom子目录下挂载 cdrom、/mnt/C 下挂载 Windows C 盘	mount	
/opt	该目录主要存放那些可选的程序	option	
/proc	可以在这个目录下获取系统及各种进程的信息，这些信息保存在内存中，由系统自己产生	process	注册表
/root	Linux 超级权限用户root的主目录		
/sbin 或 /usr/sbin	该目录用来存放系统管理用的命令与程序，执行其中的命令需要具备超级权限用户 root 的权限	system binary	
/selinux	SELinux的一些配置文件目录，SELinux 可以让 Linux 更加安全	secure linux	
/srv	服务启动后所需访问的数据目录，如 www 服务启动后，读取的网页数据就可以放在/srv/www 中	server	
/tmp	临时文件目录，用来存放不同程序执行时产生的临时文件。有些用户程序在运行过程中会产生临时文件	temporary	C:\Windows\Temp

(续表)

目录	描述	助记单词	Windows 系统 对应目录或设施
/usr	这是系统存放程序的目录，如命令、帮助文件等，大多数 Linux 发行版官方提供的软件包就安装在该目录中	UNIX System Resources	C:\Program Files
/var	这个目录中的内容经常变动，其名称为单词 variable 的缩写，/var 下的子目录/var/log 用于存放系统日志；/var/www 子目录用于存放 Apache服务器网站的文件；/var/lib 子目录用于存放一些库文件，如 MySQL 的库文件及MySQL数据库	variable	

思考与练习题 1.3　Linux 目录结构有何意义？/home、/usr、/etc、/var 等目录的用途是什么，这样安排有何好处？

思考与练习题 1.4　简述/etc、/home/guest、/root、/var、/usr、/usr/lib、/bin、/tmp、/usr/include、/usr/sbin、/boot 等目录中可保存何种用途的文件。

1.3　Linux 系统的安装、启动、登录、用户界面与命令格式

1.3.1　在 VMware 中用快照快速安装 Linux 虚拟机系统

假设安装于 D 盘，具体步骤如下：

(1) 下载 VMWare；

(2) 安装 VMware(安装过程中需要重启计算机)；

(3) 下载 Ubuntu 快照并将其解压缩到 D 盘，目录名为 D:\ubuntu；

(4) 启动 VMware，输入序列号；

(5) 选择 File→Open，选择打开 D:\ubuntu 下的.vmx 文件，双击 My Computer 下的 Linux 或 Ubuntu，启动 Ubuntu Linux；

(6) 选择 VM→Update VMWare Tools Installation，更新 VMWare 工具，启用可在主机与 Linux 虚拟机之间以拖放方式复制文件的功能。

1.3.2　启动与登录 Linux

1. 启动与登录系统

学习本节时，读者可观看“Linux 系统启动与登录(以普通用户身份登录).exe”视频。与 Windows 系统一样，为安全起见，Linux 系统启动后，会提示用户输入用户名与密码以完成登录，这样才能使用系统执行任务。Linux 系统有两类用户：普通用户与超级用户(管理用户)。超级用户为 root，在系统安装过程中创建，可执行系统管理、维护、软件安装等工作，具有执行任何操作的权限，相当于 Windows 下的管理用户 Administrator。超级用户 root 的家目录为/root。

启动 VMWare，选择 Ubuntu 并启动 Linux 系统后，以普通用户 can 的身份登录，输入密码 123456 后进入系统。打开命令窗口，输入用户命令，命令提示符为“$”，视频中还演示了创建 hello.c 文件的过程。

如果以 root 用户身份登录，打开终端窗口后就可执行任何操作，root 用户的密码也是 123456。为了能够与普通用户的操作环境有明显区别，root 用户的命令提示符是“#”。

2. 如何切换成 root 用户身份

在普通用户打开的以“$”为提示符的终端窗口中，输入命令“su -”，接着输入 root 用户的密码，就可以暂时切换成 root 用户身份，命令提示符也会变成“#”，之后就可执行需要 root 用户权限的各种操作和命令了。执行完需要 root 用户权限的操作后，执行“exit”命令，可退出 root 登录身份，恢复成原来的普通用户身份。

3. Linux 系统的一般操作方式

一般以普通用户身份登录 Linux 系统，通过在终端窗口中输入操作命令来使用系统。Linux 系统提倡使用终端命令来执行各种操作的原因是，Linux 操作命令的种类异常丰富，功能强大，如果设计成由桌面系统启动，反而过于复杂，操作不便。

以普通用户身份登录的原因是，普通用户的操作权限受限，凡涉及系统配置、管理、维护的命令都无权执行，Linux 系统遭受攻击的威胁较小，系统安全性和可靠性较高。例如，当普通用户因误操作执行硬盘格式化或对系统文件执行删除操作时，系统将拒绝执行，这样就可避免不必要的损失；又如，当普通用户意外双击执行病毒或木马程序时，病毒或木马程序的影响仅限于该用户的文件，而不至于危及整个系统的安全。

1.3.3 三种系统操作方法

Linux 提供了三种系统操作方法。

1. 图形界面

一般启动 Linux 系统后直接进入图形用户界面，通过选择主菜单或单击文件管理器，可执行系统管理和文件操作，图 1-2 是 Ubuntu 下的文件管理器界面，可通过下拉菜单、弹出菜单等执行文件、目录操作。

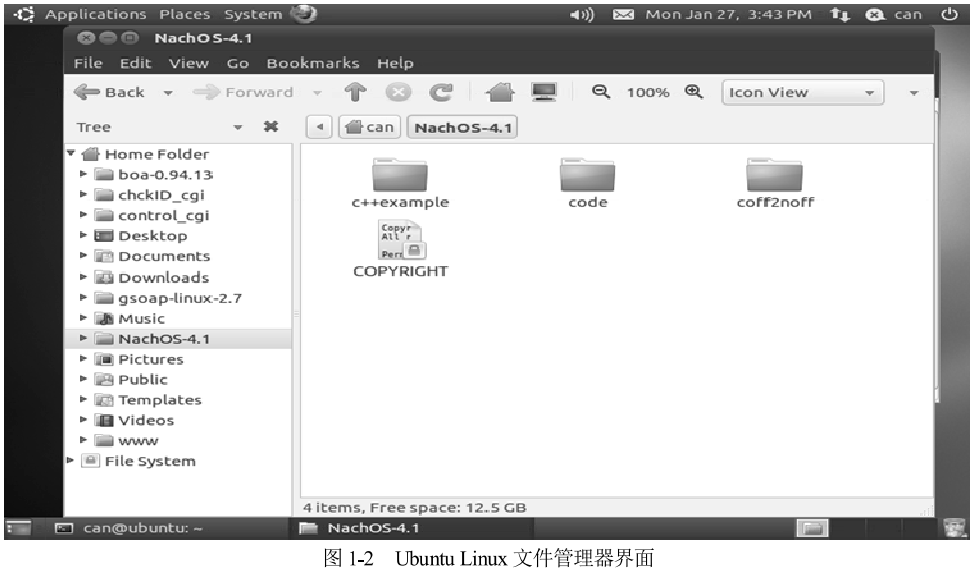

图 1-2 Ubuntu Linux 文件管理器界面

2. 命令界面

由于 Linux 系统的功能非常强大，系统操作种类异常丰富，而桌面环境通常难以支持太多操作功能，因此 Linux 的多数功能难以通过图形界面运行。为此，Linux 系统提供了大量的操作命令，可以通过在终端窗口或命令窗口中输入命令来操作 Linux 系统，命令输出也显示在终端窗口中。对计算机专业人员来说，往往会更多地使用命令界面进行各种开发工作，因为这样效率更高。在图 1-3 中，输入命令 *ls* 可显示当前目录下的文件列表，而输入 *cat /etc/passwd* 则显示用户数据库文件/etc/passwd 的内容。

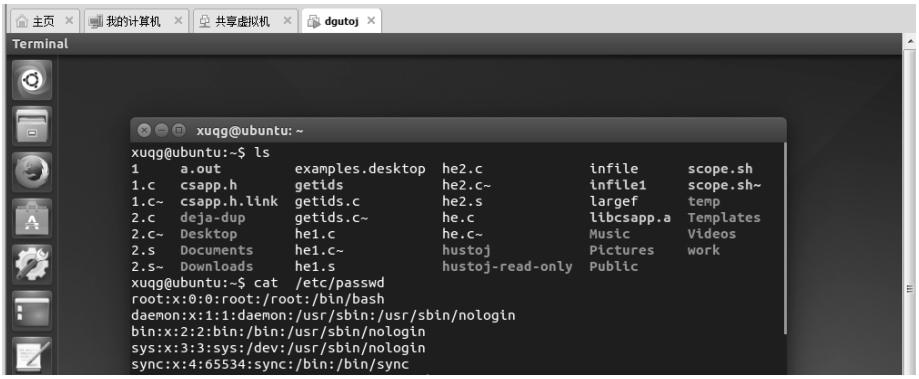

图 1-3　Linux 命令界面

3. 编程接口

编程接口是指在 C/C++语言程序(也包括其他程序)中调用 Linux 系统功能的方法，一般是通过一些称为系统调用的库函数来实现的。例如：文件操作流接口——fopen、fread、fwrite、fclose、fseek，在程序中调用这些函数可分别打开、读、写、关闭文件，以及移动读写指针；文件与设备操作系统调用接口—— open、read、write、close，在程序中调用这些函数可分别打开、读、写、关闭文件或设备。本书后面各章主要讨论如何使用 Linux 库函数进行 Linux 系统编程。

1.3.4　Linux 命令格式和说明

1. Linux 命令格式

Linux 系统命令遵循统一的格式，一般形式为：

```
$　命令名 选项 参数 1 参数 2 ……
```

命令名、选项、参数间以空格分隔，在很多命令中，选项与参数都有默认值，各部分说明如下：

1) 命令名

一般是由小写英文字母构成的字符串，往往是表示相应功能的英文单词的缩写。例如，date 表示日期；who 表示谁在系统中；cp 是 copy 的缩写，表示复制文件等。Linux 系统支持的用户命令主要放在目录/usr/bin 和/bin 下。命令名中出现大写字母的命令一般都不是正确的系统命令。

2) 选项

选项是对命令的特别定义，以“-”开始，指示命令按特定模式执行，生成输出。例如：加-l 选项的 ls 命令“***ls -l***”表示以长格式显示文件列表，每个文件一行，显示文件名与属性；加-a 选项的 ls 命令

“***ls -a***”表示文件名以点(.)开头的隐藏文件也要显示出来；若同时使用多个选项，多个选项可用一个“-”连起来，“***ls -l -a***”命令与“***ls -la***”命令的功能完全相同，显示包括隐藏文件的当前目录文件属性。

不同选项的书写一般没有先后限制，但如果选项本身带有参数，那么选项和参数必须在一起，中间不允许插入其他命令参数或命令选项。Linux 命令的选项一般位于命令参数之前，有时也放在命令参数之后。例如，按长格式显示包括隐藏文件在内的文件目录列表的命令“***ls -l -a***”也可写成“***ls -a -l***”。在将 C 程序 hello.c 编译成可执行程序 hello 的命令“***gcc -o hello hello.c***”中，命令选项“-o hello”用于指定输出文件名，命令参数是源程序文件名“hello.c”，允许二者交换，可将命令写成“***gcc hello.c -o hello***”，但不能写成“***gcc -o hello.c hello***”，因为命令选项名“-o”与其参数“hello”被命令参数 hello.c 隔开了。

3) 参数

参数用于提供命令运行的信息或命令执行过程中使用的文件名。通常，参数是一些文件名，告诉命令从哪里可以得到输入，以及把输出送到什么地方，例如，命令“***cp file1 file2***”将文件 *file1* 复制到文件 *file2*。

思考与练习题 1.5　分别指出命令“***ls -l -a /home/can***”与“***gcc -o he he.c***”中的命令名、选项与参数，它们的含义是什么？

2. 命令说明

1) 判断命令是否成功执行

一条 Linux 命令仅在命令名、选项、参数全部正确时，才能正确执行，否则执行将失败，并显示出错信息，出错信息的格式为“命令名: 出错描述”。以下是命令执行失败的示例：

```
$ LS                                    #命令名错误，显示目录列表的命令是小写的 ls
bash: LS: command not found             #显示命令 LS 未找到错误
$ ls   -P                               #命令 ls 无选项-P
ls: invalid option – P
$ ls   -l   PP                          #文件 PP 不存在
ls: cannot access PP: No such file or directory
```

2) 命令输出

如果命令执行后有输出，成功执行后的输出信息会紧接着命令串显示。由于很多 Linux 命令没有输出，如下面将要介绍的目录与文件操作命令 cd、mkdir、rmdir、rm、mv，因此它们在执行完毕后，将立即显示命令提示符“$”或“#”。由于含有错误的命令一定会显示出错信息，因此一条命令如果执行后没有任何输出而是立即显示命令提示符，则说明该命令的执行一定是成功的。例如：

```
$ cd                                    #该命令无任何输出，执行必然成功
$
```

3) 联机帮助

Linux 操作系统的联机帮助对每个命令(包括主要系统配置文件)的准确语法都做了说明，可以使用 man、info 等命令来获取相应命令的联机说明，如“***man ls***”和“***info ls***”。一般按字符 q 即可退出 man 和 info 命令。

思考与练习题 1.6　查找联机帮助，了解 wc 命令的功能。

思考与练习题 1.7　查找联机帮助，给出文件/etc/fstab、命令 pwd 的用途。

4) 本书命令输入说明

若命令提示符为“#”，则假定是以管理用户 root 身份登录；若命令提示符为“$”，则假定是以普通用户 can 身份登录。为方便读者练习，书中的用户输入命令和信息以斜体文本行显示，系统显示内容以常规字体文本行显示，用户输入的命令行后面以“#”开头的文字是对命令功能的解释，如图 1-4 所示。

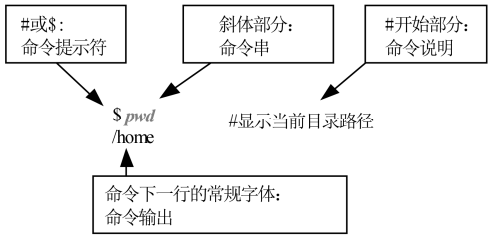

图 1-4　本书命令输入说明

1.4 Linux 文件、目录操作及文件属性、权限

通常，普通用户的主要工作是处理文件，通过命令从文件中读取输入数据，经过处理后，保存到另一文件。Linux 系统为每个普通用户在/home 目录下创建了一个以用户名命名的“家”目录，如用户 can 的“家”目录是/home/can，用户 guest 的“家”目录是/home/guest，但根用户 root 的“家”目录是/root。按照 Linux 系统权限管理规范，普通用户仅能在其“家”目录(用户主目录)下创建、修改、删除文件，而不能增删“家”目录之外其他目录中的文件，从而使系统有较好的文件保护能力。与文件操作相关的 Linux 命令主要包括：文件与目录操作命令、文件内容查阅命令、文件目录权限设置命令、文件查找命令等。

1.4.1 目录路径与目录操作

1. 绝对路径、工作目录和相对路径

Linux中的目录是“树状结构”，一般每个叶节点为一个普通文件，每个分支节点为一个目录文件。要操作或访问某个文件，应通过路径方式给出文件所在位置。

- 绝对路径：要对某个文件(或目录)进行操作，可在操作命令中给出从根目录“/”开始，直到文件名、上下级目录以“/”隔开的完整路径，称为绝对路径，如显示文件内容的命令 ***cat /etc/passwd*** 和 ***more /home/can/NachOS-4.1/code/test/***。Linux 系统中文件路径的目录分隔符是“/”而不是“\”，这是与 Windows 系统的另一个区别。
- 工作目录：为缩短文件路径字符串的长度，Linux 系统为每个命令窗口(Terminal，终端)和应用进程设置了一个工作目录(初始设置为用户的“家”目录，可用命令 cd 改变)，当用户操作工作目录中的文件时，仅需要在命令中给出文件名，不需要给出完整路径，以简化命令输入。若当前工作目录为“/home/can”，要在该目录下创建新文件 f1，只需要执行命令 ***touch f1***。
- 相对路径：从绝对路径中删除工作目录部分后得到的路径为相对路径。若当前工作目录为“/home/can”，则文件/home/can/NachOS-4.1/code/test/add.c 可用相对路径表示为 NachOS-4.1/

code/test/add.c，相应的文件内容显示命令也简化为 ***cat NachOS-4.1/code/test/add.c***。

2. 几个特殊目录名(“.” “..” “-” “~”)

为了便于命令的输入，Linux 系统定义了几个符号来表示一些常用的特殊目录。

- “.”代表当前工作目录，若工作目录为/home/can，则在文件路径中，“.”等同于/home/can，每个目录下都有一个文件名为“.”的目录。
- “..”代表上一层目录，若当前目录为/home/can，则“..” 表示目录/home，每个目录下也有一个文件名为“..”的目录。
- “-”代表前一个工作目录，若当前工作目录为/home/can，则执行 cd /etc 命令后，“.”表示/etc，而“-” 表示/home/can。
- “~”代表当前用户所在的家目录，若当前用户为 can，则其家目录/home/can 可表示为“~”；非当前用户 guest 的家目录则表示为~guest。

思考与练习题 1.8 用 ***ls -a*** 命令显示目录列表，可以看到每个目录下都有文件名为“.”和“..”的两个目录文件，这是为什么？

思考与练习题 1.9 当前用户为 can，当前工作目录为/home/can/work 时，文件/home/can/work/lib/wrapper.h 的相对路径是什么？文件~/a.out 的绝对路径是什么？

3. Linux 目录操作命令(cd、pwd、mkdir、rmdir、rm、ls)

Linux 下文件目录的操作包括创建目录、删除目录、切换工作目录、显示当前工作目录路径等。

1) cd(切换当前工作目录)、pwd(显示当前工作目录)

Linux 系统使用 cd(change directory)命令改变当前工作目录，使用 pwd(print work directory)命令显示当前工作目录的绝对路径。通常人们喜欢将这两个命令联合使用，用 cd 命令切换到目标目录，用 pwd 命令验证切换到哪里了。这两个命令的格式为：

```
$ cd  目录名
$ cd
$ pwd
```

以下是使用范例：

```
$ pwd                  #假设当前登录用户为 can，其“家”目录为最初的当前工作目录
 /home/can
$ cd  ~guest           #将当前工作目录切换到用户 guest 的家目录/home/guest，没有报告出错信息
                       #立即出现命令提示符，未显示任何出错信息，表明命令成功执行
$ pwd                  #显示当前工作目录，表明确实切换到用户 guest 的家目录/home/guest
 /home/guest
$ cd  ~                #表示回到用户 can 的家目录/home/can
$ pwd                  #显示当前工作目录，结果为/home/can，表明确实切换到了用户 can 的家目录
/home/can
$ cd                   #cd 命令不带任何参数和命令选项，表示回到自己的家目录/home/can
$ cd  ..               #表示切换到上级目录，即/home/can 的上层目录/home
$ pwd                  #显示当前目录路径，结果是“/home”，表明确实进入了希望的目录
 /home
$ cd  -                #表示回到前一条 cd 命令执行前的目录，即/home/can
                       #读者随后可用命令 pwd 测试 cd -是否执行成功
```

```
$ cd   /var/spool/mail      #用绝对路径直接转到目录/var/spool/mail
$ cd   ../mqueue            #用相对路径转到目录/var/spool/mqueue
```

思考与练习题 1.10　假设当前登录用户是 root，不执行命令，分析下列两个命令序列中 pwd 命令的输出。

命令序列 1：

```
# cd   ~
# pwd
# cd   ~can
# pwd
```

命令序列 2：

```
# cd   ..
# pwd
# cd   -
# pwd
```

2) mkdir(创建目录)、rmdir(删除空目录)、rm(删除非空目录)

Linux 提供的 mkdir(make directory)、rmdir(remove directory)两个命令分别用于创建新的目录、删除空目录，但删除非空目录要用 rm(remove)命令。通常会在某个 mkdir、rmdir、rm 命令后跟 ls(list)命令，列出文件目录，以验证目录创建、目录删除操作是否成功。

```
$ mkdir 目录名              //创建目录
$ ls   目录名               //显示目录列表
$ rmdir 空目录名            //删除空目录
$ rm   -rf   目录名         //删除任何目录，选项-rf 表示强制删除包括子目录在内的文件
```

以下是使用范例(假设先以 can 用户身份登录并打开终端窗口)：

```
$ cd   /tmp              #/tmp 是可读写公共临时目录
$ pwd                    #显示当前工作目录，为/tmp
/tmp
$ rm    -rf    *         #删除当前目录下的所有文件与目录，清空，一般慎用
$ ls                     #显示当前目录列表，已经为空
$ mkdir   test           #在当前目录/tmp 下建立名为 test 的新目录
$ ls                     #用 ls 测试执行情况，看到了 test 目录名，表示创建成功
test
$ mkdir   test1   test/sub   test2   #创建 test1、test/sub、test2 三个目录
$ ls   .    test         #列出当前工作目录和 test 目录列表，看到了三个新创建的目录
test   test1   test2
test:
sub
$ rmdir   test1          #删除空目录 test1，成功
$ rmdir   test           #试图删除非空目录 test，报告失败及出错原因
rmdir: failed to remove 'test1': Directory not empty
$ rm   -rf   test        #改用带-rf 选项的 rm 命令删除非空目录 test，执行成功
$ ls                     #再检视目录内容
test2                    #仅剩目录 test2，test 与 test1 都被删除
```

3) ls(文件目录检视命令)

ls 命令用于检视指定目录下的文件列表与文件属性，其应用十分广泛，常用格式为：

```
$ ls   [-aAdfFhilRS]  目录名
```

其中方括号[]表示其中的命令选项可有可无，以下是对常用命令选项的说明。

- *-a*：列出全部文件(或称文档)，包括文件名以“.”开头的隐藏文件。
- *-A*：列出全部文件，包括隐藏文件(但不包括“.”与“..”这两个目录)，这个选项用得较多。

- *-F*：根据文件、目录等信息类型，给出类型标记符号。例如，*代表可执行文档；/代表目录；=代表 socket 文件；|代表 FIFO 文件；无标记符号者为普通无执行权限文件。
- *-i*：列出索引节点(inode)编号。
- *-l*：以长格式列出目录内容，包含大多数文件属性，这个选项用得较多。
- *-R*：连同子目录内容一起列出。

以下是一些命令执行范例(先以 root 用户身份登录并打开终端窗口)：

```
$ cd                    #回到用户 can 的“家”目录
$ ls                    #显示当前目录的文件列表
Desktop        Nachos-3.4-for-ubuntu.tar.gz    Public
…
 $ ls    -A             #显示当前目录的文件列表，包括文件名以“.”开头的隐藏文件
                        #文件名以“.”开头的隐藏文件，在文件管理器中以及使用
                        #不带-A 和-a 选项的命令时都不会显示
.bash_history      .lesshst     Pictures
…
$ ls    /etc            #给出绝对路径，列出目录/etc 下的文件名
…
$ ls    -F              #列出当前目录的文件，给出每个文件的类型标记
Desktop/     nachos-3.4/    Pictures/
fifo1|       a.out*        test/       fl
$ ls   -l   ~           #将家目录(可用符号“~”表示)下的所有文件
                        #及详细属性列出来，每行一个文件
total 24708
drwxr-xr-x 2 root root       4096 2012-08-21 17:31 Desktop
drwxr-xr-x 2 root root       4096 2012-08-18 23:27 Documents
drwxr-xr-x 2 root root       4096 2012-08-18 23:27 Downloads
-rw-r--r-- 1 root root          0 2015-02-01 11:41 fl
prw-r--r-- 1 root root          0 2015-02-01 11:38 fifo1

$ ls   -i               #显示当前目录(省略目录名时为当前目录)下所有文件的
                        #文件名及其 i 节点号(显示于文件名之前)
686757 Desktop         686812   nachos-4.0.tar
807026 Documents       807159   NachOS-4.1.bak

$ ls   -ial
或
$ ls   -i   -a   -l     #显示当前目录下的所有文件
                        #(包括隐藏文件)的名称、i 节点及详细属性
                        #(i 节点的概念在下面介绍)
                        #多个命令选项可写在一起，也可分开写
683678 -rw-------   1 root root       7428 2014-04-05 15:44 .bash_history
686917 -rw-r--r--   1 root root       3135 2012-08-19 15:07 .bashrc
925835 drwx------   5 root root       4096 2015-02-01 08:07 .cache
678320 drwx------  9 root root       4096 2012-10-24 17:55 .config
……
```

思考与练习题 1.11　使用 ls 命令查看 can 主目录下有哪些隐藏文件，并猜测其用途。

1.4.2　文件属性与权限

1. 文件属性

前面的 ls -l 命令以长格式显示目录下的文件列表，每行显示一个文件的属性，每个文件或目录常用的属性有 9 种，如下所示：

```
root@ubuntu:~# ls -ild com1 fifo1 f1 work
695133  crw-r--r--  1  root  root  54, 1  2015-02-01 12:11  com1
695132  -rw-r--r--  1  root  root      0  2015-02-01 11:41  f1
694990  prw-r--r--  1  root  root      0  2015-02-0111:38   fifo1
689855  drwxr-xr-x  4  root  root   4096  2012-10-24 23:26  work
索引节点号  文件类型  访问权限  链接计数  所属用户  所属用户组  文件大小（以字节计）  最后修改时间  文件名
```

其中，所属用户是指该文件归属哪个用户，所属用户组是指归属哪个用户组。文件大小以字节为单位，但由于管道 fifo1 的数据完全存在于内存中，因此不占用磁盘空间，其文件大小显示为 0；字符设备文件 com1 也不是真正的文件，只是以文件名的形式来表示设备，文件大小字段中的第 1 个数 54 表示设备类型，称为主设备号，文件大小字段中的第 2 个数 1 表示该设备在同类设备中的编号为 1。文件属性中其他字段的含义稍后介绍。

Linux 系统在文件目录列表中用字符-、d、c、b、p、l 分别表示常规文件、目录文件、字符设备文件、块设备文件、管道文件、符号链接文件。Linux 文件系统中用 16 位二进制数对文件类型和权限进行编码，图 1-5 描述了 Linux 文件类型和访问权限位的结构(类型名为 st_mode)。其中，12~15 位是文件类型的编码，符号 S_IFxxx 表示相应文件类型的宏，若文件具有相应的类型，则其宏所对应的位为 1。例如目录文件类型，st_mode 的位 14 为 1，则表示这种文件类型的宏的八进制数值为 S_IFDIR=0040 000；而对于符号链接类型，st_mode 的位 13、位 15 均为 1，因此 S_IFLNK=0120 000。

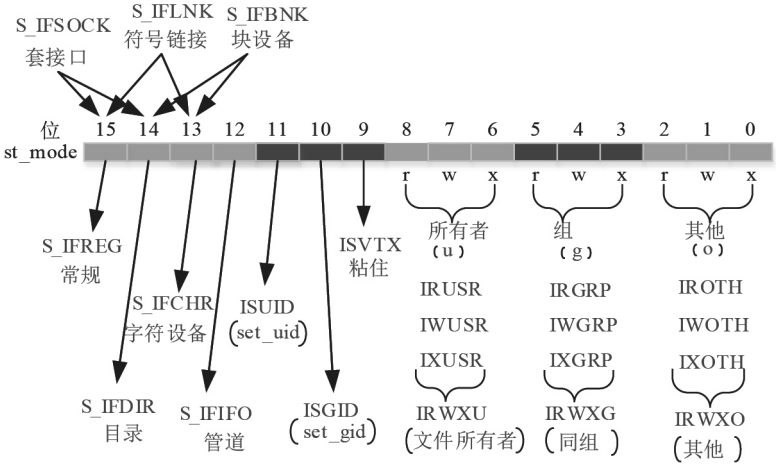

图 1-5　Linux 文件类型和访问权限位的结构

2. 文件访问权限

Linux对每个文件(包括文件目录、管道、设备等)都设置了访问权限，对所属用户(又称文件所有者，owner)、所属用户组(group)和其他用户(other)，都可设置读(r, read)、写(w, write)、执行(x, execute)三种访问权限。

st_mode 用位 0~11 表示文件访问权限，其中位 6~8 分别对应文件所有者的执行(x)、写(w)、读(r)权限，当 owner 拥有某种权限时，对应位为 1，否则为 0。位 3~5 为所属用户组访问权限，位 0~2 为其他用户访问权限。当使用命令 ls -l 显示文件权限时，若某位置有 r、w 或 x，则表示某类用户对文件有相应权限；若某位置为-，则表示无对应权限。例如，rw-r--r--表示文件所有者有读、写权限，无执行权限，所属用户组的用户和其他所有用户仅有读权限，无写、执行权限。st_mode 的位 9~11 仅在特殊情况下使用，一般为 0。

对于普通文件、管道和设备等文件来说，某用户对一个文件有 r 权限，是指该用户能读这个文件的内容；有 w 权限，是指能更改文件的内容；有 x 权限，是指能执行这个文件代表的程序或命令，当然，此时该文件应该是可执行的命令文件或脚本文件。

对于目录文件来说，权限解释有所不同。某用户对目录有 r 权限，表示能列出该目录的内容；有 w 权限，表示能在该目录中增加或删除文件；有 x 权限，表示能用 cd 命令进入该目录。

以下面的文件为例进行说明：

```
-rwxr-xr-x    1    can      users      1234567      2015-02-01 11:41   hello
drwxr-xr--    2    alice    users      4096         2015-02-01 12:41   sub
```

文件 hello 访问权限的左边三位 rwx 表示所属用户 can 对文件 hello 有读、写、执行三种权限；中间三位 r-x 表示属于用户组 users 的用户对文件 hello 有读、执行权限，本来应该出现“w”的位置换成了符号“-”，表示没有写权限；右边三位 r-x 表示既不是用户 can 又不在 users 用户组中的用户对该文件有读和执行权限。目录 sub 访问权限的左边三位 rwx 表示所属用户 alice 能列出目录内容、在该目录中创建和删除文件、能进入该目录；中间三位 r-x 表示用户组 users 中的用户能列出目录内容、进入该目录；右边三位 r--表示所有其他用户仅能列出目录内容，既不能进入该目录，又不能在该目录下增删文件。

1.4.3 Linux 文件操作命令

1. 复制、移动与删除文件(cp、mv、rm、ln)

在 Linux 系统中，复制文件使用 cp(copy)命令，cp 命令的用途有很多，除单纯的复制功能，还可以建立符号链接文件(相当于 Windows 系统的快捷方式，其中保存的是被链接文件的路径)、比对两个文件的新旧，从而予以更新，以及复制整个目录，等等。

ln(link)命令用于创建硬链接(hard link)与符号链接(symbolic link)，硬链接为同一索引节点的另一文件名，符号链接仅为某文件的一条路径。

rm(remove)命令用于移除文件，不但可删除文件，还可删除目录。

mv(move)命令用于移动文件或目录到一个新的目录位置，也可以用于重命名(rename)文件。

1) cp(复制文件或目录)

命令格式：

① 创建一个文件的副本

```
cp [-adfilprsu]  源文件(source)  目的文件(destination)
```

② 将多个选定文件复制到某目录下

```
cp [options] source1 source2 source3 .... directory
```

常用选项如下。

-f：为强制(force)的意思，若目的文件存在或有其他疑问，不会询问用户，而是强制复制。

-i：若目的文件(destination)已经存在，在覆盖时会先询问用户。

-l：创建文件的硬链接，而非创建新文件。

-r：递归持续复制，用于复制目录。

-s：创建一个符号链接，符号链接相当于 Windows 环境下的快捷方式。

示例 1-1(复制单个文件)：将家目录下的.bashrc 文件复制到/tmp 目录下，将文件名改为 bashrc。

```
$ cd   /tmp                    #进入/tmp 目录
                               #可以用 pwd 来确认是否已进入希望的目录
$ cp   ~/.bashrc   bashrc      #复制成当前目录下的文件 bashrc.bak
$                              #立即显示提示符$，无错误报告，命令执行成功
```

示例 1-2(复制单个文件)：将/var/log/wtmp 复制到/tmp 目录下，文件名不变。

```
$ cd   /tmp
$ cp   /var/log/wtmp   .       #复制到当前目录“.”下
$ ls   -l   /var/log/wtmp wtmp
-rw-rw-r--   1 root utmp 71808 Jul 18 12:46 /var/log/wtmp
-rw-r--r--   1 root root 71808 Jul 18 21:58 wtmp
```

示例 1-3(复制整个目录)：复制/etc/目录中的所有内容到/tmp 目录下。

```
$ cd   /tmp
$ cp   /etc/   /tmp            #由于复制的内容是目录，因此若还使用通常的复制方式就会导致出错
cp: omitting directory `/etc`
$ cp   -r   /etc/ /tmp         #增加-r 选项，复制成功
```

示例 1-4(创建硬链接、符号链接)：为示例 1-1 复制的 bashrc 文件创建硬链接和符号链接。

```
$ ls   -l   bashrc
-rw-r--r--   1 root root 395 Jul 18 22:08 bashrc
$ cp   -s bashrc bashrc_slink  或 ln -s bashrc bashrc_slink      #创建符号链接(softlink)
$ cp   -l   bashrc bashrc_hlink 或 ln   bashrc bashrc_hlink     #创建硬链接
$ ls   -l   bashrc*                                             #显示文件列表，以验证是否创建成功
-rw-r--r--   2 root root 395 Jul 18 22:08 bashrc                #这是原来的文件
-rw-r--r--   2 root root 395 Jul 18 22:08 bashrc_hlink           #这是新建的硬链接
                                                                #两个文件的链接计数都变为 2
lrwxrwxrwx   1 root root    6 Jul 18 22:31 bashrc_slink -> bashrc
                                                                #新建的符号链接
```

示例 1-5(同时复制多个文件)：将家目录中的.bashrc 及.bash_history 文件复制到/tmp 目录下。

```
$ cp   ~/.bashrc   ~/.bash_history   /tmp
$
```

2) rm(移除文件或目录)

命令格式：rm [-fir] 文件或目录

常用选项如下。

-f：意指强制移除。

-i：互动模式，在删除前会询问用户是否动作。

-r：递归删除，见到文件删文件，见到目录删目录。

示例 1-6：复制一个文件，然后删除。

```
$ cd   /tmp
$ cp   ~/.bashrc   bashrc
$ rm      bashrc                    #删除当前目录下的文件 bashrc
```

示例 1-7：删除一个不为空的目录。

```
$ mkdir   test
$ cp   ~/.bashrc     test/          #将文件复制到 test 目录中，test 就不是空目录了
$ rmdir   test                      #试图删除 test 目录
rmdir: `test`: Directory not empty  #删不掉，因为 test 不是空目录
$ rm   -rf   test                   #添加-rf 选项后就删除成功了
```

3) mv(移动文件与目录，或者更名)。

常用格式：

```
mv [-fiu] source destination                          (文件或目录更名)
mv [options] source1 source2 source3 .... directory   (文件或目录移动)
```

常用选项如下。

-f：强制直接移动而不询问。

-i：若目标文件(destination)已经存在，就会询问是否覆盖。

-u：若目标文件已经存在，且源文件(source)比较新，则更新(update)。

示例 1-8(移动单个文件)：复制一个文件，创建一个目录，将复制的文件移到该目录中。

```
$ cd   /tmp
$ cp   ~/.bashrc   bashrc
$ mkdir   mvtest                    #保证文件要移去的地方作为目录已经存在
$ mv   bashrc   mvtest              #将文件 bashrc 移到目录 mvtest 中
$
```

示例 1-9(目录更名)：将刚刚创建的目录 mvtest 更名为 mvtest2。

```
$ mv   mvtest   mvtest2             #执行该命令前 mvtest2 不是目录，否则就是移动目录
```

示例 1-10(移动多个文件)：再创建两个文件，全部移到/tmp/mvtest2 目录中。

```
$ cp   ~/.bashrc   bashrc1
$ cp   ~/.bashrc   bashrc2
$ mv   bashrc1   bashrc2   mvtest2
```

思考与练习题 1.12

① 当前登录用户为 can，在其主目录下创建工作目录 work，并将/home/NachOS-4.1 整个目录复制到 work 目录下，写出会话过程(即命令序列)。

② 写出删除/home/can/work/NachOS-4.1/整个目录的命令。

2. 查阅文件内容(cat、tac、head、tail、more、less、od)

1) 直接检视文本文件内容：cat、tac、head、tail

检视文本文件内容的最常用命令是 cat(catenate)和 tac，cat 是按正常顺序显示内容，tac 则逆序显示文件内容。这两个命令仅适合查看较小文件的内容，因为它们会一次性地将所有内容以刷屏方式显示在终端窗口中，实际上最后展示在用户面前的只是最后一屏，要查看前面的文本行，需要回滚终端窗口中的滚动条。有时只需要查看文件的前若干行和后若干行，这时可分别用 head 和 tail 命令来实现。

常用格式：

```
# cat [-AEnTv] [文件名]        # tac [文件名]
# head  文件名                 # tail  文件名
```

cat 命令常用于显示文件内容，下面仅给出该命令的使用范例。

示例 1-11：检视文件/etc/passwd 的内容。

```
$ cat   /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
…
```

示例 1-12：承接上例，顺便打印行号。

```
$ cat   -n   /etc/passwd
1   root:x:0:0:root:/root:/bin/bash
2   daemon:x:1:1:daemon:/usr/sbin:/bin/sh
…
```

2) 翻页检视文本文件内容：more、less

more 和 less 命令按翻页方式在屏幕上打印文本文件内容，用得也非常多。不同的是：more 命令按翻页方式向下显示文件内容，less 命令按翻页方式向下或向上显示文件内容，因此使用 more 命令不能查看已看过的页，而使用 less 命令还可以回看已经看过的页。常用格式为：

```
more  文件名
less   文件名
```

对于多页文件，more 和 less 命令先显示第一页，翻页的方法如下。

- 空格键(space)：向下翻一页。
- Enter 键：向下翻一行。
- [page down]：向下翻一页。
- [page up]：向上翻一页，仅用于 less 命令。

以下两个命令还提供了可迅速找到所需内容页面的方法。

- “/字符串”：向下查找字符串。

- “?字符串”：向上查找字符串，仅用于 less 命令。

在文件内容尚未展示完毕的情况下，要退出命令，只需要输入字母“q”。

示例 1-13：用 more 或 less 命令翻页检视文件/etc/passwd 的内容。

```
$ more   /etc/passwd
……
avahi-autoipd:x:103:108:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:104:109:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
--More--(51%)
$ less    /etc/passwd
……
avahi-autoipd:x:103:108:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:104:109:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
```

思考与练习题 1.13 查看/proc/mounts、/proc/cpuinfo、/proc/meminfo、/proc/version、/proc/uptime、/proc/devices、/proc/modules、/etc/passwd、/etc/shadow、/etc/fstab、/etc/group 文件的内容，猜测它们的用途。

非文本文件(二进制文件)的内容一般通过特定应用程序查看，如数据库文件的内容需要通过数据库管理工具查看。如果只需要知道每字节的值，可使用 od 命令。假设文件 test.txt 的内容是“计算机与网络安全学院”，用 od 命令按十六进制显示的各字节内容为：

```
$ od   -x   test.txt
0000000 e83b a1ae aee7 e697 ba9c b8e4 e78e 91bd
0000020 bbe7 e59c 89ae 85e5 e5a8 a6ad 99e9 3ba2
0000040 000a
0000041
```

其中，每行的第 1 列是本列第 1 字节离文件起始处的八进制偏移量。包括换行符在内，该文件大小为 33 字节，每个汉字用 3 字节编码。

3. 创建与编辑文件(gedit、touch、dd)

在 Linux 环境下，若未启动图形界面，可用 vi 等工具创建、编辑源程序等文本文件。vi 的启动命令为“vi 文件名”，详细使用方法见相关参考资料，在此不做介绍。若已启动图形用户界面，一般用 gedit 等 GUI 编辑器。

1) 用 gedit 创建或编辑文本文件

常用格式：

```
$ gedit 文件名
```

示例 1-14：用 gedit 打开源程序 h1.c 并进行编辑，命令为$ ***gedit   h1.c &***。

命令后加“&”符号表示在后台执行命令，并立即显示命令提示符，图 1-6 显示了 gedit 文件编辑器的界面。

2) 用 touch 命令创建空文件

touch 命令一般用于创建空文件，常用格式为：

```
$ touch   文件名
```

示例 1-15：在/tmp 目录下新建一个空文件 testtouch。

```
$ cd   /tmp
$ touch   testtouch
$ ls   -l   testtouch
-rw-r--r--   1 root root       0 Jul 19 20:49 testtouch
```

图 1-6　gedit 文件编辑器的界面

3) 用 dd 命令创建指定大小且初始化为 0 的文件

dd 命令可用于创建指定大小、内容不做要求的文件。

命令格式为：

```
dd   if=/dev/zero   of=文件名 count=块数 bs=块大小
```

含义是：从设备/dev/zero 创建大小为 bs*count 字节的文件，总共从设备读取 count 块，每块大小为 bs，因此文件大小为 bs*count。由于从设备/dev/zero 读出的数据全为 0，因此新建的文件被初始化为 0。

示例 1-16：在/tmp 目录下创建一个大小为 10 MB 的文件 testdd。

```
$ cd   /tmp
$ dd   if=/dev/zero   of=testdd   count=10240   bs=1024
$ ls   -l   testdd
-rw-r--r--   1 root root       0 Jul 19 20:49 testtouch
```

1.4.4　修改文件属性

文件通常有文件名、链接计数、文件大小、修改时间、文件类型、访问权限、文件所有者、文件用户组等属性，这些属性保存在索引节点(又称 i 节点)中，i 节点在创建文件时由系统分配。所有文件的索引节点位于磁盘或分区的特定区域，每个 i 节点在表中有一个编号，就是索引节点号。在文件的诸多属性中，文件名可用 mv 命令修改；i 节点号在文件创建时分配，不可改变；链接计数表示有几个名称指向同一个 i 节点(索引节点)，由系统自动维护；文件大小以字节数为单位，在用户向

文件写入内容或调整文件大小时由系统自动修改；修改时间由系统更新为最后写入文件的时间；文件类型是文件固有的属性，不能更改。能够通过专门命令修改的属性主要是访问权限、文件所有者、文件用户组三种属性，为方便某种需要，Linux 系统也允许通过命令直接更新文件的最后修改时间。比如将“家”目录的 Desktop 子目录的修改时间更新为当前时间。

```
$ cd
$ ls  -d  -l  Desktop
drwxr-xr-x 2 root root 4096 2012-08-21 17:31 Desktop
$ touch  Desktop
$ ls  -d  -l  Desktop
drwxr-xr-x 2 root root 4096 2015-02-01 17:20 Desktop
```

1. 文件权限更改：chmod

由前面的文件访问权限说明可知，Linux 文件的访问权限位有 12 位。其中位 9~11 不常使用，通常取值为 0。经常使用的是位 0~8，这 9 个二进制位分成三组，从高到低分别为文件所有者访问权限、所属用户组访问权限、其他用户访问权限。例如，对于 rwxr-xr--，位 6~8 为 111(显示为 rwx)，表示文件所有者有完整的读、写、执行权限；位 3~5 为 101(显示为 r-x)，表示同组用户只有读、执行权限；位 0~2 为 100(显示为 r--)，表示其他用户仅有读权限。这种 9 位的权限正好方便用三位八进制数 754 来表示，因此文件权限修改命令 chmod 通常有两种指明文件权限的格式：三位八进制数格式和 rwxrwxrwx 格式。

chmod 命令的基本格式为：

```
chmod  [-R]  三位的八进制数 文件或目录名
chmod  [-R]  [u,g,o] [+,-]  [r,w,x]  文件或目录
```

命令说明：

(1) 第一种格式直接将文件的权限设置成三位八进制数表示的权限，-R 命令选项表示递归设置(recursive)，将整个目录及其所有文件设置成指定权限。

(2) 第二种格式用于给 user(文件所有者)、group(用户组)、other(其他用户)增加或减少某种权限，如参数 u+x 表示给文件所有者增加执行权限，g+w 表示给同组用户增加写权限，o-r 表示取消其他用户的读权限，ugo+r 表示给文件所有者、同组用户、其他用户都增加读权限。

示例 1-17：在/tmp 目录下创建文件 f52、f521、f522，将文件 f52 的文件权限更改为 777，为所有用户添加对 f521 文件的读写权限，去掉所有用户对 f522 文件的写权限。

```
$ cd  /tmp
$ touch  f52  f521  f522                    #创建三个空文件
$ ls  -l  f52  f521  f522
-rw-rw-r-- 1 xuqg xuqg 0 Mar  5 18:31 f52
-rw-rw-r-- 1 xuqg xuqg 0 Mar  5 18:31 f521
-rw-rw-r-- 1 xuqg xuqg 0 Mar  5 18:31 f522
$ chmod  777  f52                           #将文件权限设置为 777，即 rwxrwxrwx
$ chmod  ugo+rw  f521  或 chmod ug+rw,o+r,o+w f521
                                            #为文件所有者(user)、同组用户、
                                            #其他用户(other)增加 rw 权限
$ chmod  ugo-w  f522                        #对三类用户(u、g、o)取消 w 权限
$ ls  f52  f521  f522  -l                   #验证前面三条命令执行结果的正确性
```

```
-rwxrwxrwx 1 xuqg xuqg 0 Mar    5 18:34 f52
-rw-rw-rw- 1 xuqg xuqg 0 Mar    5 18:34 f521
-r--r--r-- 1 xuqg xuqg 0 Mar    5 18:34 f522
```

2. 文件归属更改：chown、chgrp

chown、chgrp 命令分别用于文件所有者、文件所属用户组，一般需要 root 权限。因为随意改变文件所属用户或用户组可能会带来安全问题，所以这种操作仅允许 root 用户执行。基本命令格式为：

```
chown   用户名 文件名              #更改文件所属用户名
chgrp    组名 文件名               #更改文件所属用户组
```

示例 1-18：以 root 身份登录，在/tmp 目录下创建文件 f53，将其文件所有者、所属用户组分别更改为 can、bin。

```
# cd   /tmp
# touch   f53
# ls   -l   f53
-rw-r--r-- 1 root root 0 2015-02-01 23:10 f53        #f53 原来属于 root 用户、root 用户组
# chown   can   f52
# chgrp   bin   f52
$ ls   -l   f52
-rw-r--r-- 1 can bin 0 2015-02-01 22:10 f53          #f53 现在属于 can 用户、bin 用户组
```

思考与练习题 1.14

(1) 一个 Linux 文件的八进制数访问权限为 755，用 ls -l 命令显示的文件权限是什么？假设用 ls -l 命令显示的文件权限是 rw-r--r--，用八进制数表示的权限值是多少？

(2) 写出命令，在当前目录下创建文件 f54，将其访问权限设置为 664。

(3) 当前目录下文件 test.sh 的权限是 rw-r--r--，成功执行命令 chmod +x test.sh 后，test.sh 文件的权限变成_____，用八进制数表示为_____。

1.4.5　使用通配符(“*”和“？”)匹配文件名

前面的文件操作命令仅对一个文件或目录进行操作，但 cp、mv、rm 等命令也可以分别对多个文件或目录进行复制、移动、删除操作。选择多个文件或目录有两种方法。一种是直接列出待访问的多个文件名或目录名，例如：

```
rm   ff1   ff2              #同时删除两个文件 ff1 和 ff2
cp   ff1   ff2   personal   #同时将两个文件复制到目录 personal 中
```

另一种方法是使用通配符指出文件名或目录名，常用的通配符有以下两个。

- *：匹配任何字符串。
- ?：匹配任何一个字符。

示例 1-19：在/tmp 目录下创建两个文件 ff1 和 ff2，将所有文件名以 ff 开头、长度为 3 个字符的文件复制到目录 personal 中。

```
$ cd   /tmp
$ mkdir   personal
$ touch   ff1   ff2
```

```
$ mv   ff?   personal
$ ls   personal
```

示例 1-20：删除 personal 目录下所有文件名以 ff 开头的文件。

```
$ cd   /tmp
$ rm    personal/ff*
$ ls   personal
```

示例 1-21：删除 personal 目录下的所有文件、目录，包括子目录。

```
$ cd    /tmp
$ rm   personal /*   -rf
$ ls   personal
```

思考与练习题 1.15

(1) 写出命令，删除当前目录下所有文件名以“f”开头的文件和以“.o”结尾的文件。

(2) 写出命令，显示所有文件名仅包含两个字符的文件。

1.4.6 文件的压缩与打包

在系统管理和编程开发中，经常需要对一批文件(一个目录或满足某种条件的一些文件)进行打包、压缩，以便发布、传播等，也需要对压缩文件、打包文件执行解压缩、解包操作。Linux 系统提供了 compress、gzip、bzip2、tar、bzcat、dd、cpio 等一系列工具，能满足不同场景下的打包、解包之需。

1. 文件的压缩与打包命令(compress、gzip、bzip2、tar、bzcat、dd、cpio)

compress、gzip、bzip2 命令采用不同的压缩算法，实现单个文件的压缩、解压缩；tar 命令用于多文件的打包、解包，该命令还可通过命令选项来调用压缩命令，对打包后的文件进行压缩，或先对文件解压缩，再解包。各命令的主要功能如下。

- compress：压缩文件，压缩后缀为.z。
- gzip：压缩文件，压缩后缀为.gz。
- bzip2：压缩文件，压缩后缀为.bz2。
- tar：打包一批文件，包文件后缀为.tar。

通常将 tar 与 gzip 或 bzip2 命令结合起来执行打包与压缩操作。

常用命令格式：$ tar <选项> [压缩文件] <文件列表>

常用命令选项：

-cvf 打包	-xvf 解包
-zcvf 打包并压缩成.gz 格式文件	-zxvf 先对.gz 文件解压缩，再解包
-cjvf 打包并压缩成.bz2 格式文件	-xjvf 先对.bz2 文件解压缩，再解包

示例 1-22：在当前目录下创建目录 dir5，在其中创建 4 个文件 f1、f2、f3、f4，对该目录打包并压缩成文件 dir5.tar.gz，删除该目录，然后解包 dir5.tar.gz。

```
$ mkdir   dir5
$ cd   dir5
```

```
$ touch   f1   f2   f3   f4          #创建 4 个空文件
$ cd   ..                           #进入父目录
$ ls dir5                           #列出目录 dir5 下的文件
f1   f2   f3   f4
$ tar   -zcvf   dir5.tar.gz   dir5  #将整个目录 dir5 打包后压缩成 dir5.tar.gz，
                                    #也就是对目录 dir5 进行备份
$ rm   -rf   dir5                   #删除目录 dir5
$ ls   dir5                         #列出目录 dir5 下的文件，该目录已不存在
ls: cannot access dir5: No such file or directory
$ tar   -zxvf   dir5.tar.gz         #解压缩并解包文件 dir5.tar.gz
$ ls   dir5                         #验证目录 dir5 是否被成功恢复
f1   f2   f3   f4
```

2. 在 Windows 主机与 Linux 虚拟机之间互传文件

在本课程实验中，经常需要将在 Linux 虚拟机上创建的文档或源程序导出到 Windows 主机上，或进行反向传输。可通过简单的拖放操作在 Windows 主机与 VMware Linux 虚拟机之间互传文件。

(1) Linux 虚拟机到 Windows 主机的文件传输方法。

将 Linux 虚拟机上的文件用 tar 命令打包成一个压缩文件，之后打开 Linux 的文件管理器，将压缩文件拖到 Windows 主机上的某个文件夹(桌面)中。

(2) Windows 主机到 Linux 虚拟机的文件传输方法。

打开 Linux 文件管理器，将打包并压缩好的文件从 Windows 系统拖到 Linux 系统中的指定位置，用双击操作和 tar 命令对压缩文件解压缩和解包即可。

(3) 请参阅“Linux 文件目录压缩解压缩及与 Windows 系统间文件互传(演示).exe”视频演示文件。

思考与练习题 1.16　写出命令，将目录 work 下的所有.c 源代码文件打包并压缩成 prog.tar.bz2，并将其传送到 Windows 主机。

思考与练习题 1.17　写出命令，将 Windows 主机上的 Web 服务器源代码包 boa-0.94.13.tar.gz 传送到 Linux 虚拟机，将其解压缩并解包，展开其目录结构。

1.5 输入/输出重定向和管道

Linux 环境有很多有用的特性可以给命令操作带来极大便利。除前面介绍的相对路径、特殊目录名、通配符，还有输入重定向、输出重定向和管道。输入重定向是指将从终端读取输入数据改为通过符号“<”从文件读取。输出重定向是指将命令的正常输出改送到文件而非终端，重定向的方法是在命令串后用“>”或“>>”指明输出文件名。输出重定向可将比较长的输出先保存起来，以后再查看和分析。管道是指用一个“|”符号将两个命令连起来，将前一命令的输出直接作为后一命令的输入。

下面是一个范例：

```
$ man   passwd > a                  #将前一命令给出的 passwd 联机帮助重定向到文件 a
                                    #覆盖文件 a 的所有内容
$ date >> a                         #将命令 date 给出的日期时间信息追加到文件 a
```

```
$ cat  <  /etc/passwd              #不带参数的 cat 命令本来是从终端读取输入，
                                   #通过输入重定向改从文件读取
$ more  /etc/passwd | sort         #将文件/etc/passwd 的内容送往命令 sort 排序输出
$ find  ~  -name "*.c"| more       #find 命令在当前用户的家目录树中查找所有后缀为.c 的文件名，
                                   #文件信息交由 more 分页显示
$ grep  -r  main() | more          #grep 命令在当前目录树文件中搜索包含"main()"的
                                   #文本，交由命令 more 分页显示
```

1.6 本章小结

本章主要讲述 Linux 系统的基本知识、目录结构、文件属性、访问权限，重点介绍 Linux 系统常用的文件操作命令，为日后在 Linux 环境下进行编程打下基础。希望深入学习 Linux 操作命令用法的读者，可参考专门的书籍。表 1-2 汇总了 Linux 系统常用的用户操作命令。

表 1-2　Linux 系统常用的用户操作命令列表

序号	功能	命令名	常用格式举例
1	列出文件列表	ls、dir	①ls　②ls -l　③ls -al ④ls /etc　⑤ls -il ~ ⑥ls ~root　⑦ls -F /tmp
2	显示当前工作目录路径	pwd	pwd
3	切换工作目录	cd	①cd /proc　②cd ..　③cd ~ ④cd ~root
4	创建文件目录	mkdir	①mkdir dir1 ②mkdir -p dir2/dir3
5	删除空目录	rmdir	rmdir dir2/dir3
6	复制文件	cp	①cp /etc/passwd ②cp /tmp/a* work ③cp ~/.bashrc bashrc.bak ④cp -rf /home/NachOS-4.1 work ⑤cp -s ~/.bashrc bashrc_slink
7	创建硬链接和符号链接	cp、ln	①cp -l ~/.bashrc bashrc_hlink 或 ln ~/.bashrc bashrc_slink ②cp -s ~/.bashrc bashrc_slink 或 ln -s ~/.bashrc bashrc_slink
8	删除文件或目录(包括非空目录)	rm	①rm bashrc　②rm -rf test/* ③rm /tmp/a?　④rm /tmp/b*
9	移动、更名文件、目录	mv	①mv file1 file2　②mv file1 dir1 ③mv dir1 dir2　④mv /tmp/c* ./dir1
10	显示文本文件内容	cat、tac、more、less	①cat /etc/passwd ②tac /etc/passw ③more /etc/passwd
11	创建文本文件	gedit、vi	略
12	创建空文件，更新文件修改时间	touch	①touch f521 ②touch ~/Desktop

(续表)

序号	功能	命令名	常用格式举例
13	修改文件权限	chmod	①chmod　ugo+rw　f521 ②chmod　777 f52
14	更改用户、用户组	chown chgrp	①chgrp　bin　f52 ②chown　can　f52
15	文件/目录的打包、压缩与解压缩、解包	tar	tar　-zcvf　dir5.tar.gz　dir5 tar　-zxvf　dir5.tar.gz

课后作业

思考与练习题 1.18　不考虑操作权限因素，下面哪些 Linux 命令是正确的？哪些不正确，存在什么问题？

(1) ls　-ial (2) ls　-i-l (3) ls/home/can (4) ls-l　/etc	(5) ls　-l/etc (6) cp　/etc/passwd　/tmp (7) cp　/etc/passwd　. (8) cp　/etc/passwd　/tmp	(9) ls - l　/home (10) Ls　/ (11) ls　\etc

思考与练习题 1.19　写出显示根目录“/”下各文件(包括隐藏文件)所有属性(包括 i 节点号)的命令，说出根目录的 i 节点号是什么？为何“.”与“..”这两个文件具有相同的 i 节点号？目录“/”的链接计数是多少？为何是这个值？

思考与练习题 1.20　以下列出了命令执行后的出错信息，请判断是缺少什么权限所致。

命令及输出	当前用户对目录/root 缺少何种权限所致？
$ cd　/root bash: cd: /root: Permission denied	
$ ls　/root ls: cannot open directory /root: Permission denied	
$ mkdir　/root/work mkdir: cannot create directory ‘/root/work’: Permission denied	

思考与练习题 1.21　图 1-7 为 Linux 系统目录树结构的一部分。

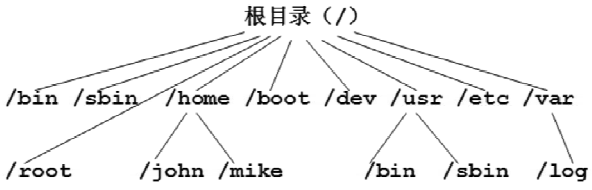

图 1-7　Linux 系统目录树结构的一部分

请完成以下练习题：

(1) 用户 john 与用户 root 的家目录在哪里？

(2) 若当前工作目录是/home/john，请给出目录 mike、usr 的相对路径与绝对路径。

(3) 若当前用户为 john，请用符号“~”给出目录 mike、home 的相对路径。

第 2 章

Linux Shell 编程

Linux 中的 Shell 作为用户与操作系统的接口，是用户使用操作系统的窗口。Shell 既是命令解释器，又是一种编程语言。作为命令解释器，Shell 是一个终端窗口，接收用户输入的命令，识别、解释、执行该命令，并向用户返回结果，Shell 的功能类似于 Windows 系统中的 cmd.exe 程序。作为编程语言，Shell 提供了变量、流程控制结构、引用、函数、数组等功能，可将公共程序、系统工具、用户程序“黏合”在一起，创建 Shell 脚本(又称 Shell 程序)，实现更加复杂的应用功能。Linux 系统的很多管理任务是通过 Shell 脚本实现的，例如，Linux 系统的启动过程就是通过运行/etc/rc.d 目录中的脚本来执行系统配置和创建服务的。Shell 脚本还用于用户工作环境的定制，如 Java 开发环境、Android 开发环境、大数据应用开发环境等，都是通过 Shell 脚本来设置的。掌握一些基本的 Shell 脚本编程知识对操作、使用 Linux 非常有帮助。每个 Linux 系统发行版本中都包含多种 Shell，一般有 Bash、Bourne Shell、TC Shell、C Shell 和 Korn Shell 等。其中，Bash 是 Bourne-Again Shell 的英文缩写，它吸收和继承了其他 Shell 的优点，成为当前应用最广泛的 Shell，是 Linux Shell 的事实标准。

本章学习目标：

- 掌握 Shell 脚本、变量、表达式、数学运算、字符串处理、输入/输出的语法结构
- 掌握使用 Shell 条件和条件、选择、循环三大控制结构的基本编程方法
- 理解全局变量、局部变量、环境变量、命令行参数的基本概念与用途
- 掌握文件 I/O 和 I/O 重定向的基本编程方法
- 理解 Shell 函数

2.1 Shell 编程基本概念

Shell 脚本就是由很多 Linux 命令通过 Shell 控制结构粘合起来构成的文本文件，可将一个 Shell 脚本当作一条 Linux 命令来执行，以高效方式完成较为复杂的管理控制功能，Shell 脚本又称 Shell 程序。

2.1.1 Shell 脚本的结构

组成 Shell 脚本的语句可包括 Linux 命令、赋值语句、输入/输出语句和流程控制结构。下面的 shscri.sh 是一个 Shell 脚本实例：

```
1    #!/bin/bash
2    list=`ls ./temp`
3    for  f  in  $list
```

```
4    do
5       mv   ./temp/$f   ./temp/$f.txt
6    done
7    echo   finished!
```

第 1 行是一个特殊的注释行(所有以字符#开头的脚本行都是注释行)，它用符号“#!”指明本脚本应该用 Shell 命令解释器/bin/bash 来解释执行脚本。第 2 行将当前目录下的文件名列表赋给变量 list，赋值表达式`ls ./temp`是一个用反引号(``)括起的命令，表示该命令的输出为表达式值，即./temp 目录下的文件列表。第 3~6 行是一个 for 循环，第 3 行表示变量 f 依次取变量 list 中的每一项，执行 for 循环体，其中变量名前加美元符号$(即$list)表示引用变量 list 的值；do 与 done 之间为循环体，仅包含一条语句，它对./temp 目录下文件名为 f 变量值的文件添加后缀“.txt”。第 7 行的 echo 命令输出信息“finished!”，表示脚本执行完毕。

2.1.2 Shell 脚本的创建与执行方法

Shell 脚本是文本文件，可以用任何文本编辑器来创建，如 gedit、vi、kate 等，甚至可以在 Windows 环境下创建好后再复制到 Linux 系统中。

为了执行 2.1.1 节中的 Shell 脚本 shscri.sh，我们先创建目录./temp 及该目录下的一些文件：

```
$ mkdir   ./temp
$ touch   f1   f2   f3   f4
```

Shell 脚本本身并不包含 CPU 直接执行的机器指令，其中的指令或语句由第 1 行指定的 Shell 命令解释器解释执行。在前面的例子中，Shell 命令解释器是/bin/bash。运行 Shell 脚本实际上是在执行/bin/bash，bash 从脚本文件中逐条读入 Shell 指令，解释执行，Shell 脚本实际上只是 bash 的一个数据文件。因此，前面 Shell 脚本的运行方法和结果是：

```
$ bash   shscri.sh
finished!
```

可通过查看/temp 目录下的文件列表来检查脚本 shscri.sh 是否执行成功。

既然脚本 shscri.sh 是 Linux Shell 脚本命令，就应允许我们直接输入文件路径和文件名以执行它。Linux bash 接收用户输入的命令时，首先会对用户权限进行检查，仅当用户有执行权限时，才会执行命令或脚本。如果输入的文件名代表可执行的二进制文件，Linux 就直接加载执行；如果输入的文件名代表文本文件，Linux 就会根据脚本第 1 行的“#!...”去查找相关的解释程序，然后启动相应的解释程序来执行脚本命令。因此，如果给 shscri.sh 添加了执行权限：

```
$ chmod   +x   shscri.sh
$ ls   -l   shscri.sh
-rw-rw-r-- 1 xuqg xuqg   104 Jul 15 16:44 shscri.sh
```

就可以输入完整路径直接执行它：

```
$ ./shscri.sh
finished!
```

上述命令串中，“./”是脚本文件 shscri.sh 所在的目录，因为“.”代表当前工作目录。

对于未添加执行权限的任何文件(包括 Linux 命令文件)，如果直接输入文件路径或文件名并试图运行它，Linux 会显示权限不允许的错误信息，例如：

```
$ ls   -l   f1
-rw-rw-r-- 1 xuqg xuqg       0 Jul 15 16:58 f1
$ ./f1
bash: ./f1: Permission denied
```

2.1.3　Shell 变量与赋值表达式

Shell 程序中一些命令产生的数据常常会被传给其他命令以做进一步处理，这可通过变量来完成。变量允许临时性地存储信息，供脚本中的其他命令使用。

用户变量可以是任何不超过 20 个字符(字母、数字或下画线)的文本字符串，用户变量区分大小写，所以变量 Varl 和变量 varl 是不同的。Shell 变量的使用非常灵活，不必事先定义变量，在给变量赋值时，变量会自动获得定义。Shell 变量值的类型都是字符串，可以将任何字符串赋值给变量。值通过等号直接赋给用户变量，在变量、等号和值之间不能出现空格。如果字符串值的中间有空格，应该用引号括起。以下是一些给用户变量赋值的例子：

```
var1=10
var2=57
var4=testing
var4="still more testing"
```

Shell 变量的赋值表达式可以由字符串常量、Shell 变量引用、Linux 命令输出直接拼接而成，但要注意以下几点。

(1) 为区分字符串常量和变量引用，Shell 要求通过美元符号($)来引用变量。

(2) 若被引用的 Shell 变量名后紧跟字母、数字、下画线等字符，则应将变量名用花括号({})括起，否则 bash 无法从中正确提取变量名。

(3) 为区分赋值表达式中的 Linux 命令和字符串常量，Linux 命令需要用反引号(``)括起。

(4) 未经定义的 Shell 变量也可引用，只是变量值为空值。

当然，赋值表达式的值也可用 echo 命令直接显示输出，在 echo 命令中，字符串表达式的中间允许存在空格而两边不用加引号。

下面的 exvar.sh 是一个使用变量的 Shell 脚本示例：

```
1    #!/bin/bash
2    #testing variables
3    num=3
4    guest1=Alice
5    guest2="Bill Gates"
6    msg="$guest1 logs out before the ${num}th day."
7    echo $msg
8    echo "current directory of ${guest2} is `pwd`. "
```

在第 5 行中，变量赋值语句右边的字符串中间有空格，故要用引号将整个字符串括起；第 6 行用$符号引用变量 num 的值，因为变量名后紧跟字符串常量 th，所以将变量名用花括号{}括起；第 8 行要获得命令 pwd 的输出，因此在 pwd 两边加反引号。下面是 Shell 脚本的输出结果：

```
$ chmod   +x   exvar.sh
$ ./exvar.sh
Alice logs out before the 3th day.
current directory of Bill Gates is /home/xuqg/temp.
```

思考与练习题 2.1　下面的 Shell 脚本存在多处错误，每行都有错误，请指出来并予以更正。

```
#!bi n/bash
var1 = 5
var2=hello world
var3=$var2abcd
echo 我的当前目录是：'pwd'
```

2.1.4　Shell 输入/输出语句

Shell 脚本用 echo 命令将包含变量值、字符串常量、命令输出的表达式值显示出来，如前面示例所示。Shell 脚本用 read 命令让用户从键盘终端输入信息，存入 Shell 变量。read 命令的格式如下：

```
read [-s] [-p prompt] variable1 variable2 …
```

上述语法表示将用户输入的多个字符串依次存入 Shell 变量 variable1、variable2、…使用 bash 命令提取输入时，以空格为字符串分隔符。bash 命令有两个命令选项，这里用方括号括起，表示命令选项根据需要可选。

- -p prompt：表示在提示用户输入前显示提示串 prompt。
- -s：使输入不可见，适用于输入密码等敏感信息。

以下脚本 io.sh 是使用案例：

```
1    #!/bin/bash
2    #testing read and echo commands
3    read  -p  "Enter your name: " first last
4    echo "Checking data for $last,$first"
5    read  -s  -p "请输入您的密码: " pass
6    echo  "Is your password really $pass?"
```

脚本的执行结果如下：

```
$ chmod +x io.sh
$ ./io.sh
Enter your name: Bill Gates
Checking data for Gates, Bill
请输入您的密码:123456  #输入的密码不显示
Is your password really 123456?
```

2.1.5　终止脚本的执行和终止状态

使用 bash 启动的命令或脚本运行完毕后，我们经常需要了解该命令或脚本的执行情况，是成功、失败，还是根本就没有执行。如果失败，是什么原因所致。后续命令需要根据终止状态进行不同的处理。因此，在 Linux 系统中，任何命令、脚本执行完毕后，都有终止状态，即使命令根本没有执行或不存在，也会有终止状态。

按照 Linux 系统规范，命令或脚本的终止状态码是一个介于 0 和 255 的整数，一般 0 代表执行成功，>0 代表执行失败。命令或脚本的终止状态分为用户设置和系统设置两种情况。

1) 用户设置终止状态码

高级编程语言和 bash 都提供了系统函数或命令来结束程序或脚本的执行，并设置终止状态码。在 C 语言程序中，用系统函数 exit()来终止程序，设置终止状态。在 bash 中，用命令 exit 终止脚本，

设置终止状态，其格式为：

```
exit 状态码
```

状态码为 0 表示脚本执行成功，为非 0 表示执行失败，非 0 编码与失败原因之间的对应关系由编程人员自行定义。

2) 系统设置终止状态码

如果 C 程序没有执行 exit()函数调用，则其终止状态码为程序最后一个函数调用的返回值；如果 Shell 脚本没有执行 exit 命令，则其终止状态码为最后执行的命令的终止状态码。

如果用户输入的命令或脚本根本就不存在或根本没有执行，或者 Linux 命令、Shell 脚本非正常终止，则其状态码由 Linux 系统根据失败原因自动设定，这时每个编码都有特定含义。表 2-1 是 Linux 命令、脚本的常见终止状态码及描述。

表 2-1　Linux 命令、脚本的常见终止状态码及描述

终止状态码	描述	终止状态码	描述
0	命令成功执行	128	无效的终止参数
126	命令无法执行	128+x	使用 Linux 信号的致命错误
127	没有找到命令	130	使用 Ctrl+C 组合键终止进程

Linux 将进程(命令、脚本的执行)的终止状态保存在一个特殊的 Shell 变量(?)中，可在进程结束时，立即读取该变量的值以取得前一个命令或脚本的终止状态。因为读取$?变量值的命令也是一条命令，所以 bash 会根据这条命令的终止状态更新变量$?的值。表 2-2 中是一些示例。

表 2-2　使用环境变量$?获取命令返回值的一些示例

命令类别	命令或程序示例	说明
Linux 命令	$ *date* Sat Sep 29 10 : 01: 30 EDT 2007 $ *echo　$?* 0	该命令执行成功，其终止状态码为 0
	$ *asdfg* -bash: asdfg: comrnand not found $ *echo　$?* 127 $ *echo　$?* 0	该命令不存在，其终止状态码为 127，因此第 1 个 echo $?的输出为 127；由于第 1 个 echo $?命令执行成功，它把$?设置成 0，这样第 2 个 echo $?的输出为 0
	$ *./myprog.c* -bash: ./myprog.c: permissi on denied $ *echo　$?* 126	myproc.c 虽然存在，但无执行权限，其终止状态码为 126
Shell 脚本	status1.sh #/bin/bash pwd $ *bash　./status1.sh* $ *echo　$?* 0	该脚本的最后一条命令执行成功，其终止状态码为 0

(续表)

命令类别	命令或程序示例	说明
Shell 脚本	status2.sh #/bin/bash exit 0 $ *bash ./status2.sh* $ *echo $?* 0	该脚本的最后一条命令是 exit 0，返回终止状态码 0，因而显示的终止状态码是 0
	status3.sh #/bin/bash exit 10 $ *bash ./status3.sh* $ *echo $?* 10	该脚本的最后一条命令是 exit 10，返回终止状态码 10，因而显示的终止状态码是 10
Linux C 程序	status4.c int main() { exit(5); } $ *gcc -o status4 status4.c* $ *./status4* $ *echo $?* 5	gcc 命令将 status4.c 编译成可执行程序 status4，命令./status4 执行该程序。由于 C 语言的库函数 exit()返回的终止状态码是 5，因此 echo $?命令显示的终止状态码是 5
	status5.c int main() { atoi("123"); } $ *gcc -o status5 status6.c* $ *./status5* $ *echo $?* 123	由于最后的函数调用atoi()的返回值是整数123(字符串“123”的整数值)，因此其终止状态码是 123

思考与练习题 2.2 在当前目录下输入命令“ls -l”并执行的输出结果如下:

```
-rw-rw-r--  1 xuqg xuqg      114 Jul 15 19:07 test.sh
-rw-rw-r--  1 xuqg xuqg      110 Jul 15 18:46 semlib.c
drwxr-xr-x  4 root root     4096 Feb  1 00:18 dir1
```

请问下列命令序列中 echo $?的输出结果是什么？(注意：Linux Shell 允许在一行中输入多个命令，命令间以分号隔开。)

(1) ./dir1; echo $?; echo $?

(2) cd; echo $?

(3) ./semlib.c; echo $?; echo $?

(4) ./abcd; echo $?