



Linux 操作系统环境下

# C 语言程序设计

● 王继业 / 主编

● 耿照新 张灵倩 / 副主编



中央民族大学出版社  
China Minzu University Press

# Linux 操作系统环境下 C 语言程序设计

王继业 主编

耿照新 张灵倩 副主编

中央民族大学出版社

## 图书在版编目 (CIP) 数据

Linux 操作系统环境下 C 语言程序设计/ 王继业主编. —北京: 中央民族大学出版社, 2009.7

ISBN 978-7-81108-702-4

I. 系… II. 王… III. C 语言—程序设计 IV.TP312

中国版本图书馆 CIP 数据核字 (2009) 第 103595 号

## Linux 操作系统环境下 C 语言程序设计

---

主 编 王继业  
责任编辑 戴苏芽  
封面设计 李志彬  
出 版 者 中央民族大学出版社  
北京市海淀区中关村南大街 27 号 邮编: 100081  
电话: 68472815 (发行部) 传真: 68932751 (发行部)  
68932218 (总编室) 68932447 (办公室)

发 行 者 全国各地新华书店  
印 刷 厂 北京九州迅驰传媒文化有限公司  
开 本 787×1092 (毫米) 1/16 印张: 17.5  
字 数 385 千字  
版 次 2009 年 7 月第 1 版 2009 年 7 月第 1 次印刷  
书 号 ISBN 978-7-81108-702-4  
定 价 45.00 元

---

版权所有 翻印必究

# 前 言

本书为编者在中央民族大学为电子工程、通信工程、应用物理和光信息科学与技术等专业开设的“Linux 程序设计”选修课的教材。从 2004 年开始，编者对上述专业的本科生开设 Linux 程序设计课程，首要的原因是，上述专业中，计算机的爱好者众多，而 Linux 在计算机爱好者中逐步流行起来。另外的原因包括，Linux 操作系统在上述专业中，具有巨大的专业价值。对于物理类专业来说，Linux 优秀的性能和与 UNIX 的兼容，使得其对于计算和数值模拟是一个难得的平台环境；而对于电子类专业来说，由于 Linux 的开源，使得 Linux 成为有价值的嵌入式操作系统的待选者，从而在电子信息行业中得到较广泛应用。

正是基于上面的理由，编者在教学过程中，主要讲述在这个 Linux 操作系统环境下如何编写应用程序，而不顾及内核层次的问题。因为内核层次的问题，往往是更专业一些的计算机工程师应该考虑的问题，作为 Linux 操作系统的用户，电子和物理类专业的人，只是一个系统平台的使用者和应用程序的编制者。尤其是电子专业的学生，将来很可能成为嵌入式系统环境下的一名应用程序设计人员，而多数嵌入式系统环境，一般更有一些像 Linux 系统而非 Windows。当然 Wince 是一个例外，因为它是 Windows。

有的学生可能会进入更软一些的行业，直接和 Linux 内核或其他系统内核打交道。这也没关系，我们的课程为应用程序设计打下了良好的基础，这对内核的理解也是有好处的，很难设想一个应用程序都设计不好的人，可以从事操作系统内核的相关设计工作，比如设计驱动程序。因为这些程序更关键，一旦出现问题，往往导致系统的不稳定，甚至死机等故障，是比应用程序更重要的代码。

作为一本本科教材，本书的作用首先就是教学，因此编者考虑教学应用的方便，在不同的章节中配置了大量代码供参考，并有相当数量的思考题，编制了上机实践课程题目。这些都是为本科教学方便而设计，同时对一般读者也具有相当参考价值。

本书除了用于本科教学外，同样可以为该领域工作的工程技术人员、其他想转入该领域工作的工程师以及 Linux 程序设计的爱好者提供帮助，也是一本非常有益的参考读物。

本书由三位编者编写，分工如下：Linux 操作部分，主要包括第 1-4 章，由耿照新编写；图形界面设计和附录等由张灵倩编写；其余由王继业编写，并完成统稿和总体调整。

编者首先感谢使用过本书（那时还只是一本讲义）的中央民族大学的学生，他们通过实践验证了上机题目的可行性。同时感谢自然科学基金“TC-1 和 Cluster 卫星的联合观测：近地磁尾尾向流对亚暴膨胀项触发的影响”，正是为该项目的工作使编者对 Linux 系统有了更多的应用和更深入的认识。

本书大部分示例均为编者自行编写和取自于编者的工程实践，极少的部分来自网络，掠美之处，不便一一指出，敬请谅解。

# 目 录

第一章 什么是 LINUX.....	1
1.1 LINUX 的历史.....	1
1.1.1 Linux 的诞生.....	1
1.1.2 Linux 的吉祥物.....	1
1.1.3 Linux 的发行版.....	2
1.1.4 自由软件基金会和 GNU 版权.....	3
1.2 LINUX 特点.....	3
1.2.1 多用户系统.....	4
1.2.2 登录.....	5
1.2.3 文件系统的层次结构.....	6
1.2.4 一般 linux 系统的文件树结构.....	7
思考和练习.....	8
第二章 LINUX 系统的安装.....	9
2.1 FEDORA CORE 6.0 的安装过程.....	9
2.1.1 安装前的准备工作.....	9
2.1.2 开始安装 FC6.....	10
2.1.3 硬盘分区.....	12
2.1.4 接下来的安装.....	14
2.1.5 安装后的配置.....	17
2.2 UBUNTU 8.04 的安装.....	20
2.2.1 安装前的准备.....	20
2.2.2 开始安装.....	21
2.2.3 安装后的设置.....	25
思考和练习.....	26
第三章 LINUX 的命令行操作.....	27
3.1 初识 LINUX.....	27
3.1.1 登录 Linux.....	27
3.1.2 几个有趣的命令.....	28
3.1.3 在文件系统中遨游.....	32
3.1.4 显示目录内容.....	33
3.2 常用 LINUX 命令.....	34
3.2.1 Linux 系统中目录的层次结构.....	35
3.2.2 文件系统.....	36

3.2.3	处理文件.....	38
3.2.4	目录和文件的属性.....	40
3.2.5	其他命令.....	41
	思考和练习.....	43
第四章	VI 的使用.....	44
4.1	编辑方式.....	44
4.1.1	光标定位.....	45
4.1.2	搜索字符串.....	45
4.1.3	替换、删除.....	45
4.1.4	剪切和粘贴.....	46
4.1.5	撤消和重复.....	46
4.2	插入方式.....	46
4.3	命令方式.....	47
4.3.1	退出命令.....	47
4.3.2	文件.....	47
4.3.3	行号使用.....	47
4.3.4	字符串搜索.....	47
4.4	规则表达式.....	48
4.4.1	正文替换.....	48
4.4.2	删除正文.....	48
4.5	编辑程序的选项和运行系统命令.....	48
	思考和练习.....	49
第五章	SHELL 环境和程序设计.....	50
5.1	BASH.....	50
5.1.1	bash shell 的基本特点.....	50
5.1.2	bash 高级属性.....	52
5.2	管道中的过滤器.....	54
5.2.1	基本过滤命令.....	54
5.2.2	awk 编程.....	59
5.3	SHELL 程序设计.....	64
5.3.1	建立和运行 shell 程序.....	65
5.3.2	shell 程序变量.....	65
5.3.3	语句和表达式.....	68
5.3.4	子函数及其他.....	74
	思考和练习.....	78
第六章	C 语言编程环境简介.....	79

6.1	第一个 C 程序 .....	79
6.2	GNU C 编译和连接 .....	80
6.2.1	gcc 命令行选项 .....	80
6.2.2	函数库和头文件 .....	81
6.2.3	GNU C 扩展 .....	82
6.3	GNU MAKE 项目管理 .....	84
6.3.1	编写 make 文件 .....	84
6.3.2	伪目标 .....	85
6.3.3	变量 .....	85
6.3.4	隐含规则和规则模式 .....	87
6.3.5	make 命令行参量 .....	87
	思考和练习 .....	87
第七章 文件系统操作 .....		89
7.1	文件操作 .....	89
7.1.1	文件的打开和关闭 .....	89
7.1.2	文件的输入输出 .....	90
7.1.3	设置打开文件的位置 .....	91
7.1.4	文件描述符和文件指针 .....	92
7.1.5	文件控制 .....	92
7.1.6	一个例子 .....	93
7.2	目录操作 .....	94
7.2.1	工作目录 .....	94
7.2.2	操作目录结构 .....	95
7.2.3	目录、文件的属性 .....	97
7.2.4	文件的其他操作 .....	98
7.2.5	一个例子 .....	98
7.3	设备文件 .....	100
7.3.1	设备文件控制函数 .....	100
7.3.2	串行口的编程 .....	101
7.3.3	声卡的编程 .....	104
	思考和练习 .....	110
第八章 进程管理 .....		112
8.1	进程执行环境 .....	112
8.1.1	程序的参数 .....	112
8.1.2	环境变量 .....	117
8.2	进程 .....	118
8.2.1	获得进程号 .....	118

8.2.2	创建进程 .....	118
8.2.3	运行程序 .....	119
8.2.4	进程的终止 .....	120
8.2.5	进程的完成状态 .....	121
8.2.6	进程创建的完整例子——执行外部命令 .....	122
	思考和练习 .....	124
第九章 信号 .....		125
9.1	信号的基本概念 .....	125
9.1.1	信号的种类 .....	125
9.1.2	信号的发生 .....	125
9.1.3	信号的传递与响应 .....	126
9.2	一些标准的信号 .....	127
9.2.1	程序出错信号 .....	127
9.2.2	程序终止信号 .....	128
9.2.3	闹钟信号 .....	129
9.2.4	异步 I/O 信号 .....	130
9.2.5	作业控制信号 .....	130
9.2.6	操作错误信号 .....	131
9.2.7	外围信号 .....	131
9.2.8	信号消息 .....	132
9.3	特定信号的反应 .....	132
9.3.1	信号的控制的基本方法 .....	132
9.3.2	信号的控制的高级方法 .....	134
9.3.3	signal()函数和 sigaction()函数的关系 .....	134
9.3.4	sigaction 函数举例 .....	135
9.3.5	sigaction 函数的标志 .....	136
9.3.6	初始化信号回调 .....	136
9.4	定义信号句柄 .....	137
9.4.1	能够返回的信号句柄 .....	137
9.4.2	结束进程的信号句柄 .....	138
9.4.3	信号函数中的非局域转移 .....	139
9.4.4	信号函数执行时到达的信号 .....	140
9.4.5	时间相近信号的合并 .....	141
9.4.6	信号句柄和非重入函数 .....	143
9.4.7	数据的原子操作和信号 .....	145
9.4.8	非原子操作带来的问题 .....	145
9.4.9	原子类型 .....	146
9.4.10	原子类型应用范式 .....	146



9.5	被信号中断的原始操作.....	146
9.6	信号的产生.....	147
9.6.1	进程自己产生.....	147
9.6.2	其他进程产生信号.....	148
9.6.3	使用 kill 的权限.....	149
9.6.4	利用 kill 函数进行进程通讯.....	149
9.7	信号的阻塞.....	150
9.7.1	阻塞信号的作用.....	151
9.7.2	信号集.....	151
9.7.3	进程的信号掩码.....	152
9.7.4	举例：禁止关键代码时信号到达.....	152
9.7.5	在信号句柄中阻塞信号.....	153
9.7.6	查找阻塞的信号.....	154
9.7.7	信号阻塞的代替方法.....	154
9.8	等待信号.....	155
9.8.1	用 pause()函数.....	155
9.8.2	pause()函数产生的问题.....	156
9.8.3	用 sigsuspend()函数.....	156
	思考和练习.....	157
第十章	进程间通讯.....	158
10.1	管道和命名管道.....	158
10.1.1	管道.....	158
10.1.2	命名管道.....	162
10.2	系统 V IPC 机制.....	162
10.2.1	一般概念.....	163
10.2.2	消息队列.....	164
10.2.3	信号量.....	167
10.2.4	共享内存.....	173
	思考和练习.....	177
第十一章	SOCKET 通讯.....	178
11.1	SOCKET 的基本概念.....	178
11.1.1	什么是 Socket.....	178
11.1.2	网络协议.....	179
11.1.3	数据结构.....	180
11.2	IP 地址和域名.....	181
11.2.1	IP 地址.....	181
11.2.2	域名系统.....	183

11.3	SOCKET 相关系统调用 .....	186
11.3.1	socket() .....	186
11.3.2	bind() .....	186
11.3.3	connect() .....	188
11.3.4	accept()和 listen() .....	189
11.3.5	send()和 recv() .....	190
11.3.6	sendto()和 recvfrom() .....	191
11.3.7	close()和 shutdown() .....	192
11.3.8	getpeername()函数 .....	192
11.3.9	gethostname()函数 .....	192
11.3.10	原始格式通讯的一个例子 .....	193
	思考和练习 .....	199
<b>第十二章</b>	<b>非连接通讯—UDP .....</b>	<b>200</b>
12.1	UDP 服务器 .....	200
12.1.1	建立 UDP 监听套接口 .....	201
12.1.2	UDP 应用协议举例 .....	203
12.2	接收 UDP .....	206
12.2.1	UDP 客户机 .....	206
12.2.2	UDP 客户举例 .....	207
	思考和练习 .....	211
<b>第十三章</b>	<b>面向连接的通讯—TCP .....</b>	<b>212</b>
13.1	服务器程序 .....	212
13.1.1	守护进程 .....	212
13.1.2	使用 TCP 连接的服务器 .....	214
13.1.3	一些额外的处理 .....	216
13.1.4	使用 TCP 的服务器程序样例 .....	217
13.2	客户程序 .....	219
	思考和练习 .....	223
<b>第十四章</b>	<b>使用 GTK 进行图形界面设计 .....</b>	<b>224</b>
14.1	GTK 的基本概念和机制 .....	224
14.1.1	Linux 下的图形系统 .....	224
14.1.2	Gtk 和 gnome .....	225
14.1.3	gtk 基本实现机制 .....	225
14.2	使用 GTK 进行基本的图形界面设计 .....	226
14.2.1	最简单的 gtk 程序 .....	226
14.2.2	添加按钮 .....	227

14.2.3	更多信号相关的操作.....	229
14.2.4	部件的布局.....	229
14.2.5	常用的窗口部件.....	234
14.3	使用 GLADE 进行界面设计.....	235
14.3.1	glade 简介.....	235
14.3.2	简单的实例.....	235
14.3.3	进一步改进.....	238
14.3.4	简单的计算器.....	240
	思考和练习.....	243
附录 A	GNU 通用公共许可证(GPL)中文版.....	244
附录 B	GNU 通用公共授权 (第三版).....	249
附录 C	LINUX 课程上机实验.....	258
	插图、表格和示例程序目录.....	261
	参考书目.....	264

# 第一章 什么是 Linux

## 1.1 Linux 的历史

Linux 的历史虽然不长，但是它的思想源头 UNIX 却有着悠久的历史，而 Linux 秉承了 UNIX 的所有思想和优点，某种程度上也继承了 UNIX 的历史。下面详细介绍 Linux 的历史。

### 1.1.1 Linux 的诞生

Linux 是一种 UNIX 操作系统的克隆，它（的内核）由 Linus Torvalds 以及网络上组织松散的黑客队伍一起从零开始编写而成。

在上个世纪 90 年代，由于计算机硬件工业的发展，以 intel 的 x86 系统架构的个人计算机大行其道，80386 及以上的处理器的性能不仅具有优良的性能，而且具备运行现代操作系统所具备的硬件条件。但在此时，人们使用的大多还是旧而且不能发挥更大硬件功效的 DOS 等操作系统。此时，有个芬兰赫尔辛基大学 (Helsinki) 的学生 Linus Torvalds 做了件不寻常的事情，Linus 手边有个 Minix 操作系统 (Unix 为了教学目的的分支)，他对这个操作系统相当的有兴趣。他开始为自己的新硬件写了一些 Minix 上的驱动程序，并希望能够被 Minix 的作者允许加入到发行中去。但他很失望地被告知，原作者希望保持 Minix 的纯洁性，因此他想，何不自己写一个类 UNIX 的操作系统，使它支持新的硬件呢？他就很有心地研读 Unix 的核心，并且去除较为繁复的核心程序，将它改写成适用于一般个人计算机的 x86 系统上面。到了 1991 年，他终于将 0.02 版的 hobby 放到网络上供大家下载，并且由于 hobby 受到大家的肯定，相当多的朋友一起投入这个工作中，终于到了 1994 年将第一个完整的核心 Version 1.0 发布到互联网上。由于 Linux kernel 的发展是由网络上的“虚拟团队”所完成的，大家都是通过网络取得 Linux 的核心原始码，经由自己精心改造后再回传给 Linux 社群，进而一步一步地发展完成完整的 Linux 系统。Torvalds 先生是这个集团中的发起者。

### 1.1.2 Linux 的吉祥物

Linux 的吉祥物是一只可爱的小企鹅。1994 年发表 Linux 正式核心 1.0 的时候，大家要 Linus Torvalds 想一只吉祥物，Torvalds 突然想到小时候去动物园被一只企鹅追着满地打滚，企鹅的力量和速度给他留下了很深刻的印象，就把 Linux 的吉祥物定为了企鹅。

现在的吉祥物企鹅是用自由软件世界的图像设计工具 GIMP 设计的，其完整的设

计过程可以参看 <http://www.isc.tamu.edu/~lewing/linux/>，上面有详细的描述。图 3-1 就是企鹅的图像（原图为彩色）。



图 3-1 Linux 吉祥物

### 1.1.3 Linux 的发行版

由于 Linux 的稳定性良好，并且可以在便宜的 x86 架构下的计算机平台运作，所以吸引了很多软件商与自由软件的开发团队在这个 Linux 的核心上面开发相关的软件，例如有名的 sendmail, wu-ftp, apache 等等。此外，亦有一些商业公司发现这个商机，因此，这些商业公司或者是非营利性的工作团队，便将 Linux 核心、核心工具与相关的软件集合起来，并加入自己公司或团队创意开发的系统管理模块与工具，而释出一套可以完整安装的操作系统，这个完整的 Linux 操作系统，我们就称呼它为发行版。当然，由于是基于 GNU 的架构下，因此各家公司所发行的光盘套件是可以在网络上面自由下载的。不过，如果想要有必要的技术支持，就需要购买发行版的光盘。

不过，由于发展的 Linux 公司实在太多了，例如有名的 Red Hat, Debian, Mandrake, SuSE, Ubuntu 等等，所以很多人都很担心，每个发行版是否都不相同呢？这大可不必，由于各个发行版都是架构在 Linux 内核下来开发属于自己公司风格的发行版，因此大家都遵守 Linux Standard Base (LSB) 规范，也就是说，各个发行版其实都是差不多的。大家可以按照自己的喜好来选择 Linux 的发行版光盘。下面列出几个主要的 Linux 发行版情况：

Red Hat, 红帽子，网址为 <http://www.redhat.com>，其发行的 RH Linux 曾风行一时，后来改为 Fedora Core Linux，现在版本为 Fedora Linux 9.0，是当前最流行的发行版之一。另外，该公司还致力于开发类 Linux 的嵌入式操作系统 eCos。

Mandrake, 网址为 <http://www.linux-mandrake.com/en/>，曾经很流行，现在仍然有不少支持者。

Slackware, 最古老的发行版之一，网址为 <http://www.slackware.com/>，由于其一直保持简单和纯粹的风格，因此拥有稳定的用户群。

SuSE, 网址为 [http://www.suse.com/index\\_us.html](http://www.suse.com/index_us.html)，曾经是欧洲最流行的发行版，现在仍然保持活跃。

Debian, 网址为: <http://www.debian.org/>, 是当前流行的发行版之一。

Ubuntu, 是众多的发行版中的后起之秀, 当前用户非常多, 是最具风格的一个发行版。

#### 1.1.4 自由软件基金会和 GNU 版权

当前流行的软件按其提供方式可以划分为三种模式: 商业软件 (Commercial software)、共享软件 (Share ware) 和自由软件 (Free ware 或 Free software)。

商业软件由开发者出售拷贝并提供技术服务, 用户只有使用权, 但不得进行非法拷贝、扩散、修改或添加新功能; 共享软件由开发者提供软件试用程序拷贝授权, 用户在试用该程序拷贝一段时间之后, 必须向开发者交纳使用费用, 开发者则提供相应的升级和技术服务; 而自由软件则由开发者提供软件全部源代码, 任何用户都有权使用、拷贝、扩散、修改该软件, 同时用户也有义务将自己修改过的程序源代码公开。

1984 年, 曾和 Bill Gates 同为哈佛大学学生的 Richard Stallman 组织开发了一个完全基于自由软件的软件体系计划——GNU, 并拟定了一份普遍公共许可 (General Public License, 简称 GPL)。GNU 计划的宗旨是: 消除对于计算机程序拷贝、分发、理解和修改的限制。也就是说, 每一个人都可以在前人工作的基础上加以利用、修改或添加新内容, 但必须公开源代码, 允许其他人在此基础上继续工作。Linux 从产生到发展一直遵循的是“自由软件”的思想。正因为如此, Linux 才发展得如此迅速和健康。1994 年 3 月 14 日, Linus 发布 Linux 的第一个“产品”版 Linux 1.0 的时候, 是按完全自由发布版权进行发布的。它要求所有的源代码必须公开, 而且任何人均不得从 Linux 交易中获利。然而, 半年以后, 他开始意识到这种纯粹的自由软件的方式对于 Linux 的发布和发展来说实际上是一种障碍, 因为它限制了 Linux 以磁盘拷贝或者 CD-ROM 等媒体形式进行发布的可能, 也限制了一些商业公司参与 Linux 的进一步开发并提供技术支持的良好愿望。于是 Linus 决定转向 GPL 版权, 这一版权除了规定有自由软件的各项许可权之外, 还允许用户出售自己的程序拷贝, 并从中赢利。这一版权上的转变后来证明对于 Linux 的进一步发展确实至关重要。从此以后, 便有多家技术力量雄厚又善于市场运作的商业软件公司加入了原先完全由业余爱好者和网络黑客所参与的这场自由软件运动, 开发出了多种 Linux 的发布版本, 增加了更易于用户使用的图形界面和众多的软件开发工具, 极大地拓展了 Linux 的全球用户基础。并有多家著名的商业软件开发公司开发了基于 Linux 的商业软件, 如 IBM、SUN software inc.、ORACLE、INFORMIX 等。

所以, Linux 严格说起来, 应该是 GNU/Linux。

## 1.2 Linux 特点

Linux 首先是一种操作系统。对于一般的个人电脑而言, 当我们打开计算机, 首

先执行的程序是固化在计算机内部只读存储器的系统自检程序 (post)，自检程序如没发现问题，就把控制权交给自举程序。自举程序查找硬盘或软盘的第一个扇区(叫引导扇区)，这个扇区如果包含了合法的代码，就把系统的控制权交给它。这段代码有时是操作系统的一部分，有时不是。引导程序取得控制权后，紧接着读入相应的操作系统，可以是 windows，也可以是 Linux，或其他别的什么系统。如果是 windows，你就看到了它的启动画面，进入 windows 环境。如果是 Linux，那么 Linux 系统的内核被读入内存，并取得控制权。这时 Linux 就控制了系统的所有资源，包括 CPU、内存、外存、网络端口等。同时它给其他的程序提供服务。从而完成一个操作系统的所有应该具有的功能。

Linux 系统是一种 UNIX，它被称为多用户系统，因为多个用户可以同时使用一台 Linux 机器；又由于每个用户可以同时运行多个程序，所以 Linux 又是多任务的。同时 Linux 又具有 UNIX 系统所有特点：有 250 多种单独的命令，既包括列出文件、文件复制这种简单的命令，也包含网络管理、系统修正等复杂的命令。Linux 又是一种多选择的系统，同一个任务，可以有多个命令、多种方法实现，同一个命令，又能完成多种功能。这也许是 Linux 最不同寻常的地方。比如，简单 cp 命令能够拷贝文件、建立文件的连接、显示文件内容、播放音乐等，也许还有许多功能。正因为这样，Linux 提供解决几乎所有问题的方法，能完成所有任务，并且用各种不同的方法。这和其他操作系统有点不一样。比如，我们想把另一块硬盘按照已有的一块硬盘的结构建立分区和文件系统，并复制内容，这被称为“克隆”。在 windows 系统下，我们得使用专门的第三方软件实现，操作系统没有这个功能。在 Linux 系统下，我们只要一个命令：`dd if=/dev/hda of=/dev/hdb` 即可。

正因为 Linux 的丰富多彩，使得学习 Linux 不是一个容易的过程。

### 1.2.1 多用户系统

多用户系统，顾名思义，应当是使多个用户能同时（或不同时）使用一台计算机，而每个用户都像拥有独立的计算机一样。那么，多个用户怎么使用一台计算机呢？这有许多方法。一种是 Linux 系统提供虚拟控制台，即在一个主控制台上通过切换提供多个界面供不同的用户使用，这种方法通常能够给一个人需要以不同身份登录系统提供方便。另一种方法是通过串行线连接不同的终端，不同的用户通过终端登录系统，运行程序。这种方法可以在银行的储蓄所的操作终端中见到。还有一种方法更为常用，就是通过网络程序，比如 telnet，实现登录和使用你的 Linux 机器。比如在我的 Linux（或 Windows）机器上，运行命令 `telnet 192.168.0.1`，在我的网络环境设置下，就可以看到如下界面：

```
Red Hat Linux release 7.1 (Seawolf)
```

```
Kernel 2.4.2-2 on an i586
```

```
login:
```

只需键入用户名和密码，就可以使用远端的 Linux 机器。你可以通过网络让远端

的计算机运行应用程序，而输入和显示通过网络由你提供。

## 1.2.2 登录

由于 Linux 系统允许许多用户同时使用计算机，因此当你想进入系统时，必须告诉计算机你是谁，这就是登录。当 Linux 内核启动后，最终要运行一类叫“Shell”的程序提供用户登录界面，shell 程序有文本界面和图形界面两类，不论是哪种 shell，用户登录后 shell 都是作为用户的“代理”在系统中执行用户的输入命令，帮助用户完成各种操作。

文本界面下常用的 shell 是 bash，如果是文本界面，登录过程大致如下：

```
Red Hat Linux release 7.1 (Seawolf)
Kernel 2.4.2-2 on an i586
login: wjy
Password:
Last login: Wed Sep  4 16:31:39 from 192.168.0.11
[wjy@linux wjy]$
```

其中 wjy 是用户名，紧接着提示输入密码，输入密码是并不显示出来，是因为保密的原因。如果用户名和密码都正确了，登录成功，系统显示一句欢迎词，这里显示的是上次登录的时间，然后给出提示符\$，我们就可以输入命令了。

需要注意的是，Linux 系统是区分大小写的，大小写不同的字母认为是不同的字符，这和 windows 系统有时候不一样，要特别注意。

登录成功后，shell 程序继续负责接受命令和完成相应的处理，上面说的提示符就是 shell 提供的，它表明 shell 程序就绪，准备接受命令。

上面的登录过程是基于字符截面的，这也是一个不寻常的特征，尤其对 windows 的用户来说。这很矛盾：一方面基于字符界面的 Linux 很难学习，另一方面，它有时（经常是）确比纯粹图形界面强大得多。

实际上，Linux 系统也有非常强大的图形界面系统（可以看成一种 shell），而且有不只一种，其中较广泛应用的一种叫 Xwindows。在一般个人电脑上，习惯于使用图形方式登录，和上面的类似，只是更漂亮一些。但是，人们认为，即使使用的 X Windows system 是最好的产品，字符界面也更胜一筹。而且，依据经验，想不经过程序就想使用 Linux 提供的所有功能是不可能的。

上面提到，进入系统需要你的用户名（user name）和密码（password），这些信息都记录在系统中一个叫 passwd 的密码文件里。在这个文件中，每个用户占一行。我的密码文件像下面这样：

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
```



```
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/var/spool/news:
ftp:x:14:50:FTP User:/var/ftp:
nobody:x:99:99:Nobody:/:
named:x:25:25:Named:/var/named:/bin/false
wjq:x:500:500:./home/wjq:/bin/bash
zlj:x:501:501:./home/zlj:/bin/bash
mysql:x:100:101:MySQL server:/var/lib/mysql:/bin/bash
```

其中每一行包含用冒号隔开的 7 部分，意义如下：

**name:** 用户名，用户登录时必须键入的名字。

**password:** 有不同的含义，如果为 **x**，表示用户密码存储在一个叫影子文件的单独文件里，如果不是 **x**，它的意义就是加密后的密码。

**Uid:** 这个数字用来识别每个用户。

**Gid:** 是用户组的识别数字。

**Comment:** 注释，说明一些关于用户的细节。

**Home:** 是用户的工作目录，又称主目录。

**Shell:** 是用户使用的 shell 程序。

其中 **root** 用户称为超级用户，或根用户，它对系统具有所有权限。

退出系统登录的方法是键入命令 **logout**，或在提示符后面键入文件结束符，这可以通过按 **ctl-D** 实现。

### 1.2.3 文件系统的层次结构

Linux 文件系统包含三类文件：

- **普通文件：**存放的是数据和程序，也就是二进制流。文件中不包含任何特定的结构。
- **目录文件：**目录是一种结构，它允许不同的文件和目录放在一起，像 windows 系统中的文件夹。其中包含的下级目录叫子目录。
- **特殊文件：**包含多种类型，一般说，它和不同进程间通讯、计算机和外部设备通讯有关系。

所有这些文件都放在一个大的树型结构中。树的根是一个单独的目录，称为根 (**root**) 目录，并且用斜杠 “/” 表示。在根目录下有一些标准的子目录和文件，所谓标准，是一种传统。在这些子目录下又包含下级子目录和文件，依此类推。

任何文件，只要不在相同的子目录下，即使文件名相同，也不会混淆。这是显而易