## 4.2 Linux基本使用：

### 一、进入Linux系统和退出Linux系统

通常情况下，要使用Linux系统，必须先登录进入系统；使用结束后，应该退出登录状态。

**一般的步骤为：**

1. 开机，启动时选择Linux操作系统，进入linux的GNOME或 KDE图形界面登录界面；或者进入命令行登录界面；
2. 在登录界面的“用户名：”（命令行界面为“login:”）后输入自己账户的用户名（命令行界面；
3. 在“口令” （命令行界面为“Passsword:”）后输入回车账户的密码，开始登录，便可进入Linux系统。
4. 注销：

（1）GNOME图像界面：点击上方任务栏的“系统”→“注销” →点击弹出窗口中的“注销”。

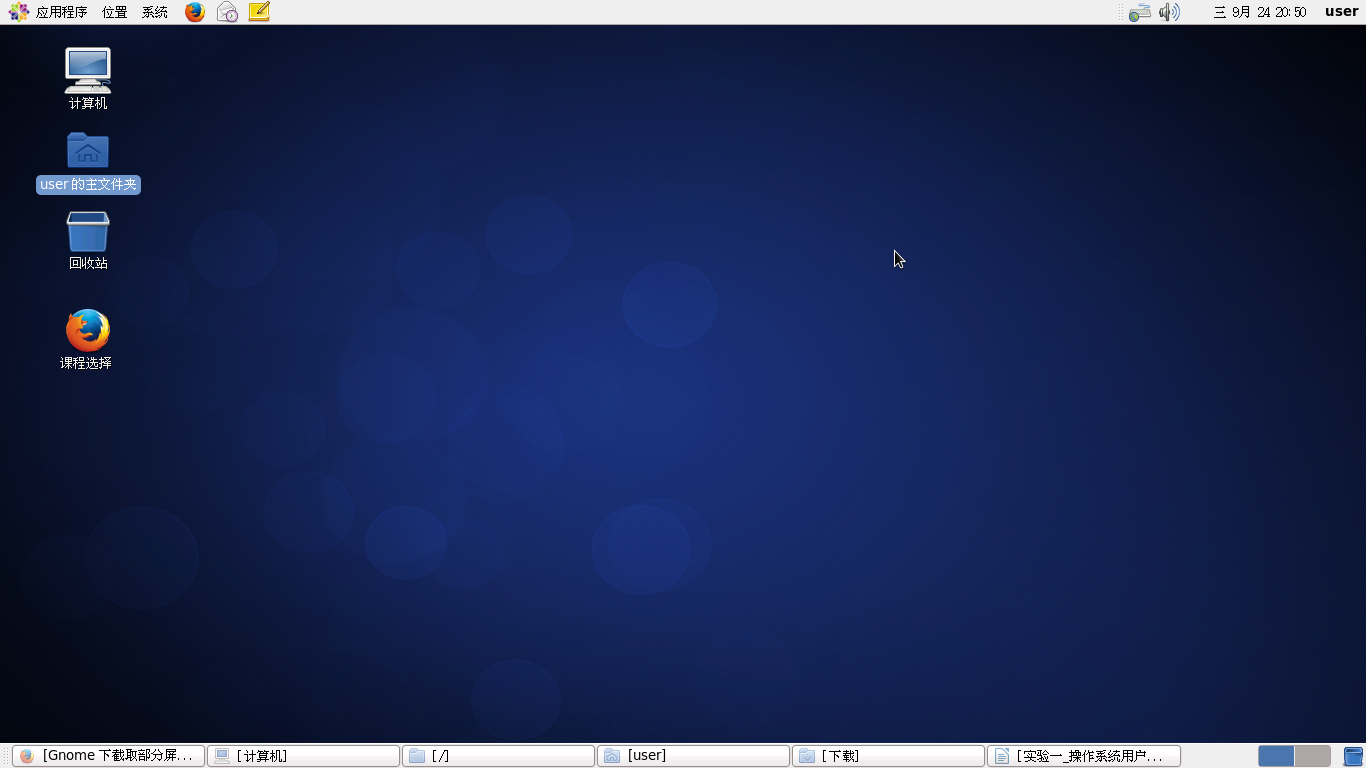
（2）KDE图像界面：点击左下角类似于windows的开始菜单的K菜单图标→点击“注销” →点击弹出窗口中的“结束当前会话”。

（3）命令行界面：在命令提示符后输入“exit”，回车。

注意：注销只是让已登录的账号离开系统，而不是关机。在计算机系统中，随着安全级别的不同，用户可能需要具备一定的权限才可以重启或关闭计算机，在命令行界面下，使用reboot命令可以重启系统，使用shutdown命令可以关闭计算机系统。

**我们的步骤：**

1. 开显示器；
2. 点击桌面上的“课程-实验环境列表”图标；
3. 选择“Linux”，再点击“进入实验环境”，不用输入用户名和密码，便可直接进入linux系统。你将看到如图2-1所示的Linux的GNOME的窗口界面。
4. **使用结束后，请同学们点击图2-1中的“课程选择”图标；**
5. **选择“XP”，再点击“进入实验环境”，将系统切换回Windows XP。**



**图1-1 GNOME的窗口界面**

### 二、图形化界面

Linux提供了类似于Windows的、可视化的图形界面——X Window的图形界面，其中的操作像在Windows操作中一样方便，有窗口、图标和菜单等，所有的管理都可以通过指点设备鼠标来控制。目前，Linux下常见的窗口管理器有KDE和GNOME，我们使用的是其中的GNOME。

**图1-3 GNOME的桌面**



如图2-1所示，整个GNOME的界面大约分为三个部分：

**图1-2 GNOME的控制面板**



**1.控制面板。**如图2-2所示，窗口最上面的部分，是GNOME的控制面板，也可以看成是任务栏，其中有“应用程序”、“位置”、“系统”和一些快捷键。你可以点击上面的“应用程序”，接着可以看到更多的程序出来。譬如，你想浏览网页，那只要选择这些程序中的“Internet”，然后选择“Firefox Web Browser”，系统就会打开火狐浏览器供你浏览网页。

**2.桌面。**界面中央蓝色的部分就是桌面。如放大后的图2-3所示，在桌面上有“计算机”、“User的主文件夹”和“回收站”3个默认的图标，第4个图标就是我们用来切换回XP系统的“课程选择”图标。其中的“计算机”类似于Windows的“我的电脑”， 是一个资源管理器，通过它你可以看到Linux的整个文件树。 “User的主文件夹”是当前登录账号对应的用户的个人工作目录，我们的系统以用户名“User”进行登录的，所以显示的是“User的主文件夹”。通常，用户的个人主文件夹位于/home目录下，以我们的账号User为例，对应的主文件夹为/home/user。另外，当执行某个程序时，程序的显示也都是在桌面的位置。

**3.下方任务栏。**如图2-4所示，界面最下面的一行就是下方任务栏，用户目前所做的所有工作会显示在这里，通过它，用户可以从这些工作中进行快的选择。最左边的火狐图标可将所有工作最小化。

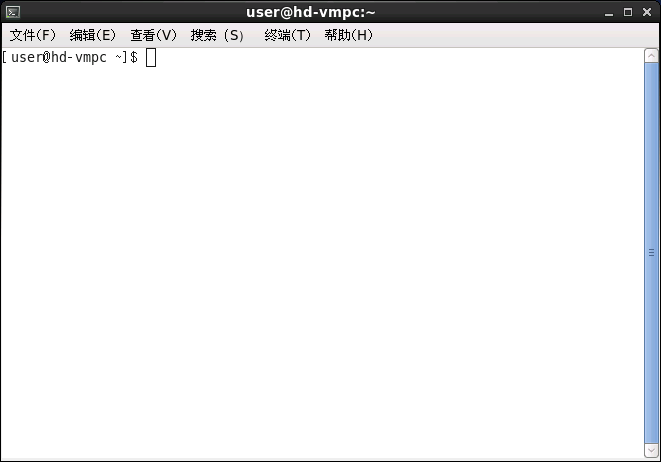
**图1-4 GNOME的下方任务栏**



**4.资源管理器。**请尝试点击“User的主文件夹”来使用资源管理器，分别点击菜单中的“文件”、“编辑”、“查看”等菜单项，看看能进行哪些操作。使用资源管理器时，注意窗口左下角的按钮，点击它，你可以方便地切换到你所需要的目录。

### 三、常用命令

点击控制面板中的“系统工具”，选择“终端”打开一个如图3.1所示的终端窗口，在这个终端窗口中，练习Linux的常用命令和vi编辑器。



**图1-5 终端窗口**

1. **pwd命令**

Linux中用“pwd”命令来查看当前工作目录的完整路径。 简单地说，每当你在终端进行操作时，你都会有一个当前工作目录。在不太确定当前位置时，就可使用pwd来判定当前目录在文件系统内的确切位置。

（1）命令格式： pwd [选项]

（2）命令功能： 查看”当前工作目录“的完整路径

（3） 常用范例：

实例1：用 pwd 命令查看当前工作目录的完整路径。如图3-2所示，当前工作目录为user用户的主工作目录/home/user。

**图1-6 显示当前工作目录**



1. **passwd命令：**

passwd 作为普通用户和超级权限用户都可以运行，但作为普通用户只能更改自己的用户密码，且前提是没有被root用户锁定；如果root用户运行passwd ，可以设置或修改任何用户的密码。若passwd 命令后面不接任何参数或用户名，则表示修改当前用户的密码。

1. 命令格式：passwd [选项]
2. 命令功能：设置修改用户密码
3. 常用参数：

-d, --delete 删除用户密码，仅能以root权限操作，用户在登录时不必输入密码；

-l, --lock 锁住用户无权更改其密码，仅能通过root权限操作；  
-u, --unlock 解除锁定；  
-x, --maximum=DAYS 两次密码修正的最大天数，后面接数字；仅能root权限操作；  
-n, --minimum=DAYS 两次密码修改的最小天数，后面接数字，仅能root权限操作；  
-w, --warning=DAYS 在距多少天提醒用户修改密码；仅能root权限操作；  
-i, --inactive=DAYS 在密码过期后多少天，用户被禁掉，仅能以root操作；  
 -S, --status 查询用户的密码状态，仅能root用户操作；

1. **who命令：**

显示目前登录系统的用户信息。

1. 命令格式：who[选项]
2. 命令功能：显示目前登录系统的用户信息
3. 常用参数：

-m 它的效果同who am i显示出自己在系统中的用户名，登录终端，登录时间

-q 只显示用户的登录帐号和登录用户的数量

-s 使用简短的格式显示

-r 可以查看当前系统的运行级别：run-level 3 Apr 21 09:37 last=S

-u 不显示使用者的动作/工作

-f 不显示使用者的上线位置

-h 不显示标题列

1. **自动补全：**

在shell中输入命令名、文件或目录名是，只需输入前几个字母，然后按一次【tab】键，系统会自动补全能够识别的部分（若不能识别，则命令名不发生变化）；再按一次【tab】键，系统显示出符合条件的所有命令或文件/目录名供用户选择。

列出shell命令集：

1. 按【tab】键两次，可显示所有shell命令；
2. 用“help”、“man builtin”、“man bash”命令，可以列出所有的内部命令；
3. 用“ls /bin”命令，可列出linux系统最基础、所有用户都能使用的外部命令；
4. 用“ls /sbin”命令，可列出只有root用户才能使用的、管理linux系统的外部命令。
5. **ps命令：**

ps命令是平时比较常用的查看进程的命令，显示瞬间进程的状态，并不动态变化；如果想对进程运行时间监控，需要用 top 工具。ps是Process Status的缩写，用来列出当前时刻（即执行ps命令的那一刻），系统中运行的那些进程的快照，根据显示的信息可以确定哪个进程正在运行、哪个进程被挂起、进程已运行了多久、进程正在使用的资源、进程的相对优先级，以及进程的标志号(PID)。

（1）命令格式：ps [选项]

（2）命令功能：显示系统所有进程的状态信息。

（3）常用参数：  
 -e 显示所有进程,环境变量  
 -f 全格式  
 -h 不显示标题  
 -l 长格式  
 -w 宽输出  
-a 显示终端上的所有进程,包括其他用户的进程

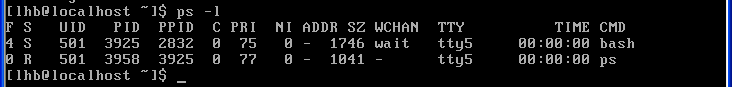
-A 显示所有进程  
-r 只显示正在运行的进程  
-x 显示没有控制终端的进程

-aux 显示所有用户（包括其他用户）的进程

（4）常用范例：

**实例1：以长格式列出当前用户的进程**

**命令：**ps -l



**图1-7 ps –l运行结果**

**说明：**

上述例子中两个进程均是用户lhb(标识数为501)的进程，其中ps正在执行，而ps进程的父进程bash进程正在等待子进程ps结束，因此处于阻塞状态。显示界面中信息的含义如下：

**F 代表这个程序的标志**， 4 代表此进程的权限为 super user，1代表这个进程被创建了但没执行过exec。

**S 代表这个程序的状态** (STAT)：

Linux系统中，进程有5种状态:

①运行(正在运行或在运行队列中等待的就绪进程)，PS将其显示为“R”

②可中断睡眠(阻塞状态, 所等的事件完成，硬件中断，或收到信号均唤醒该进程) ，PS将其显示为“S”

③不可中断睡眠(阻塞状态，但收到信号不会唤醒该进程，通常该进程可能在等I/O中断)， PS将其显示为“D”

④僵死(进程已终止, 但进程描述符即PCB仍然存在, 直到父进程调用wait4()系统调用后释放)， PS将其显示为“Z”

⑤停止(进程收到SIGSTOP, SIGSTP, SIGTIN, SIGTOU信号后停止运行运行)， PS将其显示为“T”

**UID** 进程所有者的用户标识数

**PID** 进程的内部标识数

**PPID** 父进程标识数

**C** CPU使用的资源百分比

**PRI** Priority 进程调度的优先数，数值越小优先权越高

**NI** Nice 值，用来调整进程调度的优先级

**ADDR** 是 kernel function，指出该程序在内存的哪个部分。如果是个 running的程序，一般就是 "-"

**SZ** 使用掉的内存大小

**WCHAN** 目前这个程序是否正在运行中，若为运行则显示为-

**TTY** 登录者的终端机位置

**TIME** 使用掉的 CPU 时间。

**CMD** 进程对应的命令

**实例2：显示所有进程信息**

**命令：**ps -A



**……省略部分结果……**

**图1-8 ps –A运行结果**

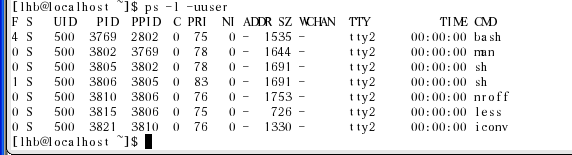
可见，ps –A不仅显示当前用户的进程，还显示其他用户和系统的进程信息。

**实例3：显示用户名为user的指定用户的进程信息**

**命令：**ps -u user

**输出：**

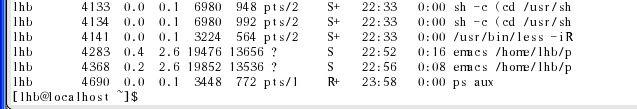
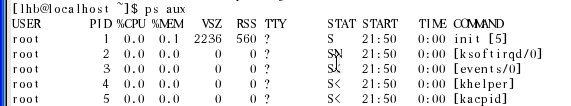
**图1-9 ps –u user运行结果**



**说明：**user用户正在使用“man ps”命令，man进程在执行过程中还创建了子进程，子进程又创建了自己的子进程，……，它们一起完成“man ps”的工作。

**实例4：查看系统所有进程**

**命令：**ps aux



**……（中间省略）……**

**图1-10 ps aux运行结果**

**说明：**ps aux的显示格式与ps –l有所不同

**USER**： 该进程所有者的用户名（而不是用户标识数）

**PID** ：与前一种格式相同，为进程标识数

**%CPU**：该进程使用掉的 CPU 资源百分比

**%MEM**：该进程所占用的物理内存百分比

**VSZ** ：该进程使用掉的虚拟内存量 (Kbytes)

**RSS** ：该进程占用的固定的内存量 (Kbytes)

**TTY** ：该进程是在tty1-tty6中的哪个终端机上面运行，若与终端机无关，则显示 ?，若为 pts/0 等，则表示是通过虚拟终端连接进主机的。

**STAT**：进程状态，同前一种格式

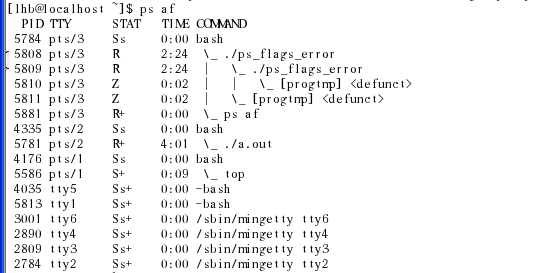
**START**：该进程被触发启动的时间

**TIME** ：该进程实际使用 CPU运行的时间

**COMMAND**：该进程运行的实际命令

**实例5：以类似进程树的方式显示**

**命令：** ps af



**图1-11 ps af运行结果**

**说明：**上述例子中以类似进程树的方式显示当前所有用户的进程，从中可以清楚地看出5784,5808,5809,5810,5811这几个进程之间的家族关系。

1. **pstree命令：**

使用ps命令得到的数据精确，但数据庞大，这一点对掌握系统整体概况来说是不容易的。pstree命令正好可以弥补这个缺憾。它能将当前的执行程序以树状结构显示。pstree命令支持指定特定进程(PID)或用户(USER)作为显示的起始。

（1）命令格式：pstree [选项] [进程标识数|用户]

（2）命令功能：pstree命令显示进程状态树，可以列出当前的进程，以及它们的树状结构。

（3）常用参数：

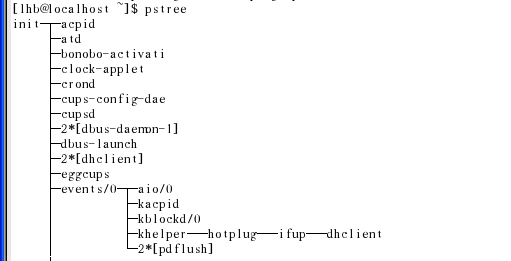
-a：显示执行程序的命令与完整参数。  
-c：取消同名程序，合并显示。  
-n：以PID大小排序。  
-p：显示PID。  
-u：显示UID信息。

-h 　列出树状图时，特别标明现在正在执行的进程和它的祖先进程。

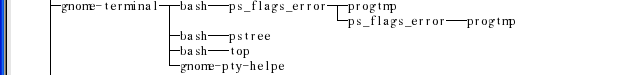
-H <进程标识数 > 此参数的效果和指定”-h”参数类似，但特别标明指定的进程及其祖先。  
（4）命令范例：  
**实例1：显示进程的树状图**

**命令：**pstree

**图1-12 pstree运行结果**



**……（中间省略）……**



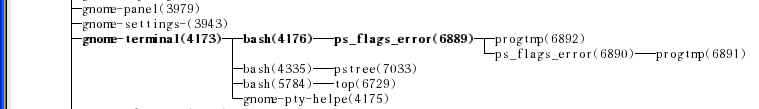
本例中通过gnome打开了3个虚拟终端，其中一个虚拟终端下正在运行pstree命令，另一个正在运行top命令，还有一个在运行用户编写的一个ps\_flags\_error程序，可清楚地看出这些进程之间的相关性。从上述树状图中，我们还可以看出init是上述所有进程的共同祖先。

**实例2：在树状图中显示出进程标识数，并特别标出标识数为6889的指定进程**

**命令：**pstree –H 6889 –p

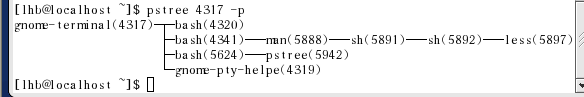
**结果：**

**图1-13 pstree –H 6889 –p运行结果**



**实例3：从标识数为4317的进程开始树状显示，同时显示进程标识数**

**命令：**pstree 4317 –p



**图1-14 pstree 4317 –p运行结果**

1. **top命令：**

top是一个动态显示过程,即可以通过用户按键来不断刷新当前状态.如果在前台执行该命令,它将独占前台,直到用户终止该程序为止.比较准确的说,top 命令提供了实时的对系统处理器的状态监视.它将显示系统中CPU最“敏感”的任务列表.该命令可以按CPU使用.内存使用和执行时间对任务进行排序；而且该命令的很多特性都可以通过交互式命令或者在个人定制文件中进行设定。

（1）命令格式：top [参数]

（2）命令功能：用来显示系统当前的进程状况，提供系统整体性能。  
（3）常用参数：

-d 指定每两次屏幕信息刷新之间的时间间隔。当然用户可以使用s交互命令来改变之。  
-p 通过指定监控进程ID来仅仅监控某个进程的状态。  
-q该选项将使top没有任何延迟的进行刷新。如果调用程序有超级用户权限，那么top将以尽可能高的优先级运行。  
-s 使top命令在安全模式中运行。这将去除交互命令所带来的潜在危险。  
-i 使top不显示任何闲置或者僵死进程。  
-c 显示整个命令行而不只是显示命令名

-b 打开/关闭当前运行进程的加亮效果

（4）top交互命令

在top 命令执行过程中可以使用的一些交互命令。这些命令都是单字母的，如果在命令行中使用了s 选项， 其中一些命令可能会被屏蔽。

h 显示帮助画面，给出一些简短的命令总结说明

k 终止一个进程。

i 忽略闲置和僵死进程。这是一个开关式命令。

q 退出top程序

r 修改进程的优先级别

S 切换到累计模式

s 改变两次刷新之间的延迟时间（单位为s），如果有小数，就换算成m s。输入0值则系统将不断刷新，默认值是3 s

f或者F 从当前显示中添加或者删除项目

o或者O 改变显示项目的顺序

l 切换显示平均负载和启动时间信息

1 数字1，在多CPU系统中，可显示每个CPU的状况，再次按1回原状态

m 切换显示内存信息

t 切换显示进程和CPU状态信息

c 切换显示命令名称和完整命令行

N 显示内容根据进程标识数进行排序

M 根据驻留内存大小进行排序

P 根据CPU使用百分比大小进行排序

T 根据时间/累计时间进行排序

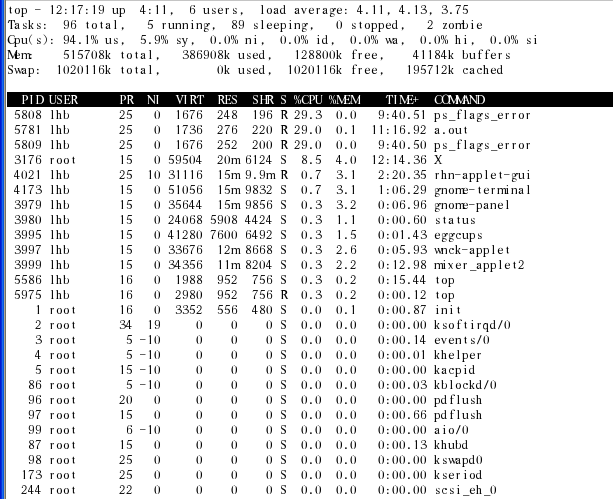
W 将当前设置写入~/.toprc文件中

（5）使用实例：

**实例1：显示进程信息**

**命令：**top

**图1-15 top运行结果**



**说明：**

**统计信息区：**前五行是当前系统情况整体的统计信息区。

**第一行**：当前的时间（12:17:19）；从开机开始到目前为止系统已累计运行的时间（如上例的4小时11分钟）；当前登录的用户数（本例为6）； 1分钟、5分钟、15分钟的平均负载情况。

**第二行：**当前系统中进程总数（96个），处于运行态的进程数（5个），睡眠状态的进程数（89个），stoped状态的进程数（0个），僵死状态的（2个）。

**第三行：**

94.1%us — 用户空间占用CPU的百分比。

5.9% sy — 内核空间占用CPU的百分比。

0.0% ni — 改变过优先级的进程占用CPU的百分比

0.0% id — 空闲CPU百分比

0.0% wa — IO等待占用CPU的百分比

0.0% hi — 硬中断（Hardware IRQ）占用CPU的百分比

0.2% si — 软中断（Software Interrupts）占用CPU的百分比

**第四行:** 物理内存总量,已使用的内存总量，空闲内存总量和缓存的内存量

**第五行：**swap交换分区信息，交换区总量，已使用的交换区总量，空闲交换区总量，缓冲的交换区总量

**第七行开始是各进程（任务）的状态监控，项目列信息说明如下：**

PID — 进程id

USER — 进程所有者

PR — 进程优先级

NI — nice值。负值表示高优先级，正值表示低优先级

VIRT — 进程使用的虚拟内存总量，单位kb。VIRT=SWAP+RES

RES — 进程使用的、未被换出的物理内存大小，单位kb。RES=CODE+DATA

SHR — 共享内存大小，单位kb

S — 进程状态。D=不可中断的睡眠状态 R=运行 S=睡眠 T=跟踪/停止 Z=僵尸进程

%CPU — 上次更新到现在的CPU时间占用百分比

%MEM — 进程使用的物理内存百分比

TIME+ — 进程使用的CPU时间总计，单位1/100秒

COMMAND — 进程名称（命令名/命令行）

上述信息会每隔3秒被动态地刷新一次，直到用户通过q命令或Ctrl+C结束命令的运行为止，刷新的时间间隔可通过命令参数修改，也可在运行过程中通过交互命令修改。

0.0% wa — IO等待占用CPU的百分比

**实例2：按批处理的方式动态显示进程信息，每10秒刷新一次，总共显示5次**

**命令：**top –b –d10–n5

**说明：**上述命令显示进程的信息，总共按批处理的方式显示，共刷新并显示5次,时间间隔为10秒。

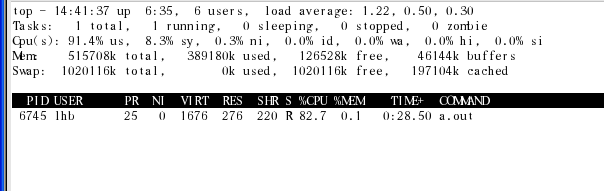
**实例3：显示进程对应的完整命令**

**命令：**top -c

**说明：**上述命令显示进程的信息时，会将进程对应的完整命令显示出来。

**实例4：只显示指定进程的信息**

**命令：**top –p 6745



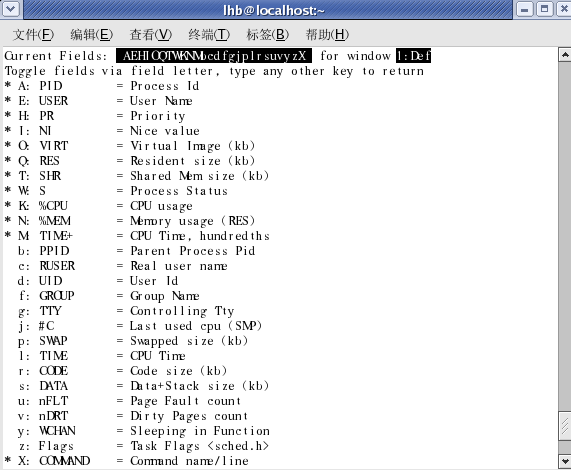
**图1-16 top –p 6745运行结果**

**说明：**上述命令只显示标识数为6745的进程的信息。

**实例5：在第一列显示出PPID信息。**

**操作方法：**

* 先使用top命令，可以看出其中的显示界面中没有显示PPID信息。
* 在top的显示界面中按字符f键，进入图1-8所示的选择显示项目的界面，其中带\*并且对应大写字母的内容是出现当前显示界面中的。找到PPID对应的字母为小写的b，因此它目前没被显示，按字母b,便可将PPID加入显示内容中（或反向将其从显示内容中去除），然后按ESC键退回到原来的显示界面。



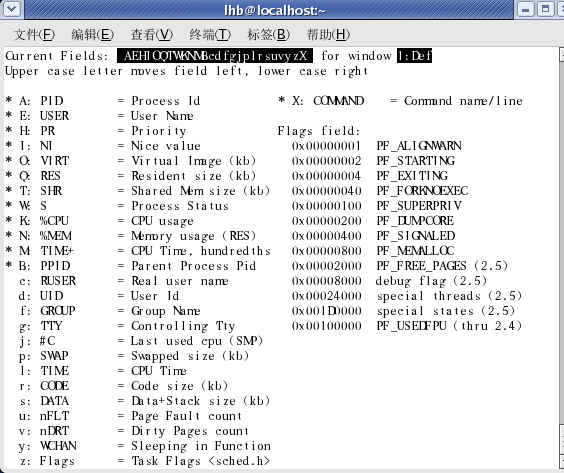
**图1-17 选择显示项目**

如图1-18所示，PPID已出现在显示内容中，但位置并不在第一列。在top显示界面中按字符o键，进入图1-10所示的调整项目顺序的界面，按项目所对应的字母的大写字母可将其往左移，小写字母则右移，因此按大写的B将PPID移到第一列，ESC退出后，可看到如图1-19所示的显示内容。



**图1-18 添加了PPID域**

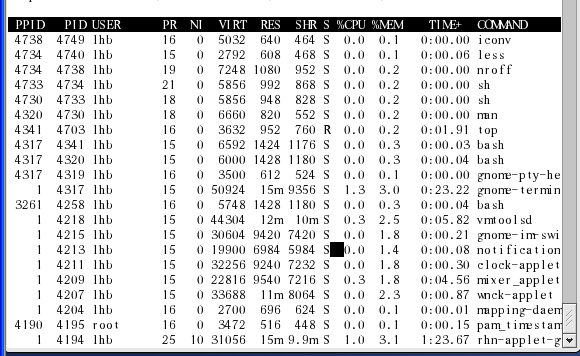
**图1-19 更改项目显示顺序**



**图1-20 调整顺序后的显示界面**

**实例6：在实例5中的基础上，将显示内容按PID的顺序进行排序。**

**操作方法：**在top显示界面下直接按大写字母N。显示结果如图1-21所示。



**图1-21 按PID顺序显示**

1. **kill命令**

Linux中的kill命令用来终止指定的进程（terminate a process），是Linux下进程管理的常用命令。通常，终止一个前台进程可以使用Ctrl+C键；但是，对于一个后台进程就须用kill命令来终止，我们需要先通过ps/pidof/pstree/top等命令获取进程PID，然后使用kill命令来杀掉相应进程。kill命令是通过向进程发送指定的信号来结束相应进程的，其实，前台进程运行时按Ctrl+C键就是向它发送了一个值为2的SIGINT信号，而进程收到该信号后，Linux的缺省动作就是终止该进程。下面给出了Linux在i386上的31个常规信号的取值和说明信息，我们特别用红色标识了终止进程的常用信号。

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **信号名称** | **缺省动作** | **说明** |
| 1 | SIGHUP | 终止 | 终止控制终端或进程 |
| 2 | SIGINT | 终止 | 键盘产生的中断(Ctrl-C) |
| 3 | SIGQUIT | dump | 键盘产生的退出 |
| 4 | SIGILL | dump | 非法指令 |
| 5 | SIGTRAP | dump | debug中断 |
| 6 | SIGABRT／SIGIOT | dump | 异常中止 |
| 7 | SIGBUS／SIGEMT | dump | 总线异常/EMT指令 |
| 8 | SIGFPE | dump | 浮点运算溢出 |
| 9 | SIGKILL | 终止 | 强制进程终止 |
| 10 | SIGUSR1 | 终止 | 用户信号,进程可自定义用途 |
| 11 | SIGSEGV | dump | 非法内存地址引用 |
| 12 | SIGUSR2 | 终止 | 用户信号，进程可自定义用途 |
| 13 | SIGPIPE | 终止 | 向某个没有读取的管道中写入数据 |
| 14 | SIGALRM | 终止 | 时钟中断(闹钟) |
| 15 | SIGTERM | 终止 | 进程终止 |
| 16 | SIGSTKFLT | 终止 | 协处理器栈错误 |
| 17 | SIGCHLD | 忽略 | 子进程退出或中断 |
| 18 | SIGCONT | 继续 | 如进程停止状态则开始运行 |
| 19 | SIGSTOP | 停止 | 停止进程运行 |
| 20 | SIGSTP | 停止 | 键盘产生的停止 |
| 21 | SIGTTIN | 停止 | 后台进程请求输入 |
| 22 | SIGTTOU | 停止 | 后台进程请求输出 |
| 23 | SIGURG | 忽略 | socket发生紧急情况 |
| 24 | SIGXCPU | dump | CPU时间限制被打破 |
| 25 | SIGXFSZ | dump | 文件大小限制被打破 |
| 26 | SIGVTALRM | 终止 | 虚拟定时时钟 |
| 27 | SIGPROF | 终止 | profile timer clock |
| 28 | SIGWINCH | 忽略 | 窗口尺寸调整 |
| 29 | SIGIO/SIGPOLL | 终止 | I/O可用 |
| 30 | SIGPWR | 终止 | 电源异常 |
| 31 | SIGSYS／SYSUNUSED | dump | 系统调用异常 |

1. 命令格式：

kill [-[s] 信号] [参数] 进程号 …

kill –l [信号]

1. 命令功能：

将指定信号发送到相应进程。

1. 常用命令参数  
    -l 参数-l（小写字母L）用于列出全部的信号名称和取值

-a 当处理当前进程时，不限制命令名和进程号的对应关系

-p 指定kill 命令只打印相关进程的进程号，而不发送任何信号

-s 指定发送信号

1. 进程号含义

进程号n>0 表示进程标识数为n的进程

进程号n=0 表示跟当前进程同组的所有进程

进程号n=-1 表示所有进程标识数大于1的进程

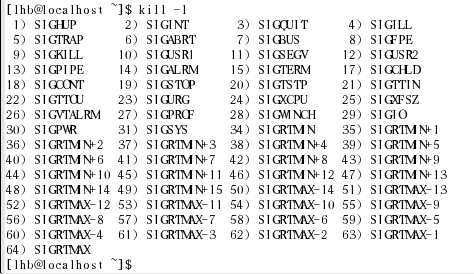
进程号n<-1 表示组标识数为|n|的所有进程

* + 当进程号为小于负1的整数时，必须用‘-- ’做前缀，以与信号的值相区别。

1. 使用实例：

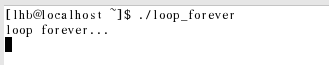
**实例1：**列出系统中所有信号

**命令：** kill –l



**图1-22 kill –l运行结果**

**实例2：**如图1-23所示，当前正在运行的loop\_forever进程是一个在终端屏幕上输出“loop forever…”后便进入无限循环的进程，用kill命令终止该进程。



**图1-23 loop\_forever运行界面**

**命令： top**

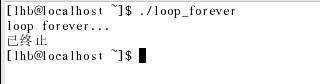
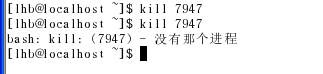
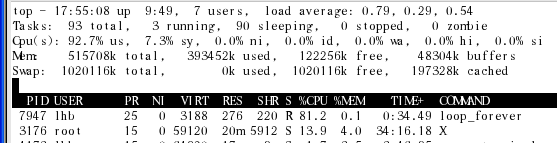
kill –SIGTERM 7947

或者 kill -15 7947

或者 kill 7947

运行结果：

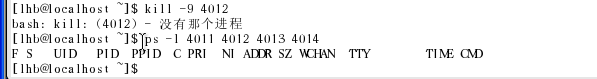
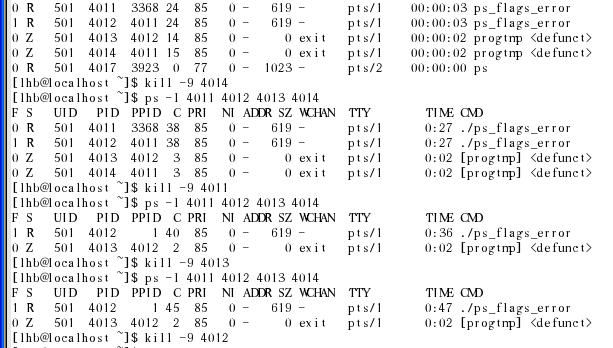
**图1-24 终止loop\_forever进程过程**



我们先用top找到loop\_forever的进程标识数为7947，然后用kill 7947（命令中缺省信号部分，则默认为发送SIGTERM信号）命令向它发信号15终止它，再次执行kill 7947时系统提示：“bash：kill：（7947）- 没有那个进程”，因为进程已经被终止了。进程被终止后，运行loop\_forever的终端会收到系统提示“已终止”，然后显示命令提示符，等待用户输入下一个命令。

**实例3：**通过ps –Al命令看到系统中有如下的的僵尸进程，试用kill -9 pid，看能否将其终止。

**运行结果：**

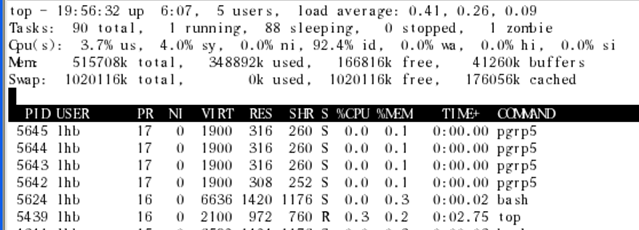


**图1-25 僵尸进程的终止**

上述僵尸进程并没有通过kill -9命令成功终止，通过终止其父进程，可杀掉僵尸进程。

**实例3：终止一组进程。**

如图1-26所示，后台运行的pgrp5程序产生了4个进程，它们的组标识数为5642，请将这4个进程全部终止掉。

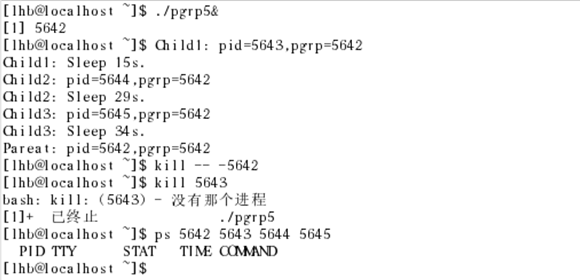


**图1-26 4个pgrp5进程**

**命令**：kill -- -5642

运行结果如图1-27所示：

**图1-27 终止一组进程**



1. **ls命令**

ls是list的缩写，缺省情况下，ls命令用来打印出当前目录的清单，如果ls指定其他目录，那么就会显示指定目录里的文件及文件夹清单。 通过ls 命令不仅可以查看linux文件夹包含的文件，而且可以查看文件权限(包括目录、文件夹、文件权限)、查看目录信息等等。

（1）命令格式：ls [选项] [目录名]

（2）命令功能：列出目标目录中所有的子目录和文件。

（3）常用参数：

-a, –all 列出目录下的所有文件，包括以 **.** 开头的隐含文件

-A 同-a，但不列出“.”(表示当前目录)和“**..**”(表示当前目录的父目录)。

-l 除了文件名之外，还将文件的权限、所有者、文件大小等信息详细列出来。

-d, –directory 将目录象文件一样显示，而不是显示其下的文件。

-i, –inode 印出每个文件的 inode 号

-R, –recursive 同时列出所有子目录层

（4） 常用范例：

例一：ls命令用于列出当前目录下所有的子目录和文件名

命令：ls

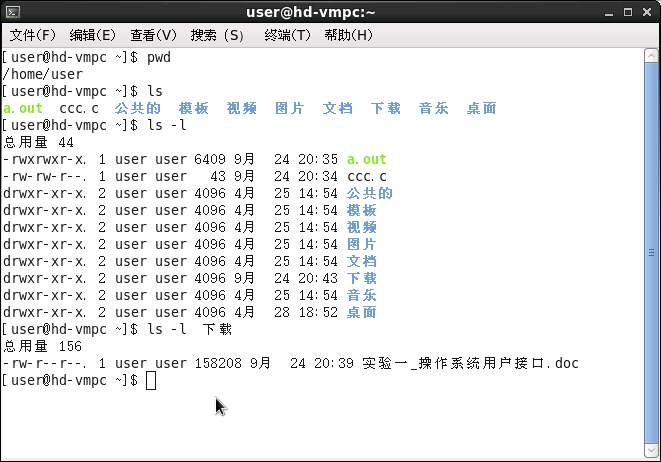
例二：ls命令用于列出当前目录下所有的子目录和文件的详细信息

命令：ls –l

例三：列出当前目录下，“下载”子文件夹下的所有文件和目录的详细资料

命令：ls -l 下载

上述三个例子的执行结果如图3.2所示。



**图3-3 ls命令范例**

“总用量 44” 代表当前目录下文件大小的总和为44K（每个目录的大小都按4K算）

第1列表示文件类型和存取权限：第一个字符有几种情况：“-”表示普通文件，“d”代表目录，“l”代表连接文件，“b”代表设备文件，可见a.out和ccc.c是两个文件，其余还有多个目录。后面的9个字符每3个为一组，分别代表文件所有者、文件所有者所在用户组、其它用户对文件拥有的权限。每组中3个字符分别代表读、写、执行的权限，若没有其中的任何一个权限则用“-”表示。执行的权限有两个字符可选“x”代表可执行，“s”代表套接口文件。

第2列表示文件的链接计数；第3列是文件的所有者。

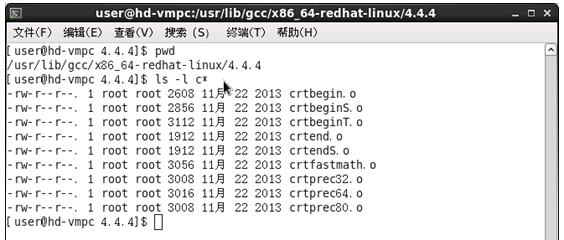
第4列是文件所有者所在用户组；第5列表示文件的大小，单位为字节。

第6、7、8列为文件最后修改的时间；第9列为文件的名字。

例四：列出当前目录中所有以“c”开头的目录和文件的详细内容，可以使用如下命令：

命令：ls -l c\*

**图3-4 列出以“c”开头的文件**



1. **CD命令**

cd命令用于改变当前目录

（1）命令格式：cd [dirName]

（2）命令功能：切换当前目录至dirName。

（3） 常用范例：

例一：进入[系统](http://www.2cto.com/os/#_blank)根目录

命令： cd /

例二：进入user的个人工作目录

命令： cd /home/user

上述例子的执行结果如图3-4所示。



**图3-5 改变当前工作目录**

**图3-4 列出以“c”开头的文件**

（4）linux cd命令使用小技巧：

Cd ~ 进入用户主目录；

cd - 返回进入此目录之前所在的目录；

cd .. 返回上级目录（若当前目录为“/“，则执行完后还在“/"；".."为上级目录的意思）；

cd ../.. 返回上两级目录；

1. **mkdir命令**

mkdir命令用来创建指定名称的目录，要求创建目录的用户在当前目录中具有写权限，并且指定的目录名不能是当前目录中已有的目录。

（1）命令格式：mkdir [选项] 目录...

（2）命令功能：

通过 mkdir 命令可以实现在指定位置创建指定的文件名的文件夹(目录)。要创建文件夹(目录)的用户必须对所创建的文件夹的父文件夹具有写权限。并且，所创建的文件夹(目录)不能与其父目录(即父文件夹)中的文件名重名，即同一个目录下不能有同名的(区分大小写)。

（3）常用参数：

-m, --mode=模式，设定权限<模式> (类似 chmod)，权限格式如777，711等

-p, --parents 可以是一个路径名称。此时若路径中的某些目录尚不存在,加上此选项后,系统将自动建立好那些尚不存在的目录,即一次可以建立多个目录;

（4） 常用范例：

例一：在当前目录下创建一个子目录test1

命令： mkdir test1

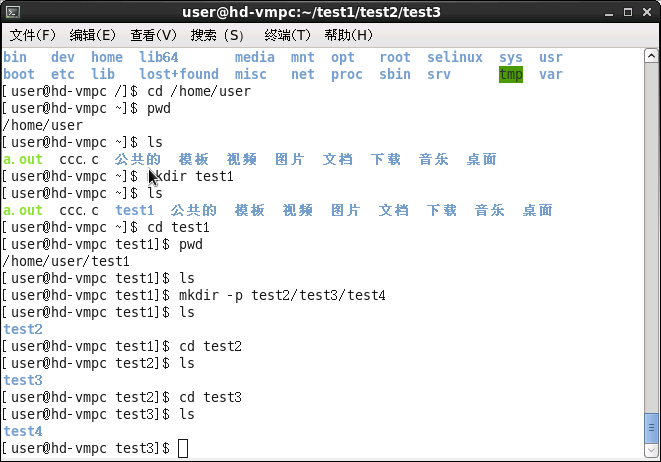
例二：将当前目录改变到test1下，并在当前目录下创建多层子目录：test2/test3/test4

命令： cd test1

mkdir –p test2/test3/test4

例三：在当前目录下创建一个权限为rwx—x—x的子目录test5

命令： mkdir –m 711 test5



**图3-6**

1. **rmdir命令**

rmdir命令的功能是删除空目录，一个目录被删除之前必须是空的。删除某目录时必须具有对父目录的写权限。

（1）命令格式：rmdir [选项]... 目录...

（2）命令功能：

该命令从一个目录中删除一个或多个子目录项，删除某目录时也必须具有对父目录 的写权限。

（3）常用参数：

- p 递归删除目录dirname，当子目录删除后其父目录为空时，也一同被删除。

-v, --verbose 显示指令执行过程

（4） 常用范例：

例一：rmdir 删除空目录test4，如果test4不空则会提示该目录非空，不删除目录，否则删除目录。

命令：rmdir test4

1. **cp命令**

cp命令用来复制文件或者目录。

（1）命令格式：

cp [选项]... [-T] 源文件 目的文件

cp [选项]... 多个源文件... 目的目录

（2）命令功能：

将源文件复制至目标文件，或将多个源文件复制至目标目录。

（3）常用参数：

-f, --force 强行复制文件或目录，不论目的文件或目录是否已经存在

**-i, --interactive**  覆盖前先询问用户

**-n, --no-clobber**  不要覆盖已存在的文件(使前面的 -i 选项失效)

-p —preserve，保留源文件或目录的属性，包括所有者、所属组、权限与时间等。

**-R, -r, --recursive** 递归处理，复制目录及目录内的所有项目

（4） 常用范例：

例一：复制单个文件ccc.c到目标文件newccc.c（newccc.c是原来不存在的文件）

命令：cp ccc.c newccc.c

例二：复制单个文件到目标目录，若目标文件是原来已存在的文件，则提示用户是否覆盖目标文件

命令：cp –i myfile1 mydir

例三：将一个目录下的所有文件复制到另一个目录下

命令：cp -r /dir1/\* /dir2

1. **rm命令**

rm命令的功能为删除一个目录中的一个或多个文件或目录，它也可以将某个目录及其下的所有文件及子目录均删除，使用的时候要特别当心。对于链接文件，只是删除了链接，原有文件均保持不变。

（1）命令格式：

Rm [选项] 文件…

（2）命令功能：

删除一个目录中的一个或多个文件或目录，如果没有使用- r选项，则rm不会删除目录。

（3）常用参数：

-f, --force 忽略不存在的文件，从不给出提示。

-i,--interactive进行交互式删除,在删除时会询问用户是否进行删除操作。

-r, -R, --recursive 指示rm将参数中列出的全部目录和子目录均递归地删除。

-v, --verbose 详细显示进行的步骤

（4） 常用范例：

例一：删除文件file1.txt，系统会先询问是否删除，输入y就会删除文件，输入n不删除。

命令：rm file1.txt

例二：删除目录/home/user/dir2及其下面的所有的内容。

命令：rm –r /home/user//dir2

1. **mv命令**

mv命令是move的缩写，可以用来移动文件或者将文件改名，经常用来备份文件或者目录。

（1）命令格式：

Mv [选项] 源文件或目录 目标文件或目录

（2）命令功能：

视mv命令中第二个参数类型的不同（是目标文件还是目标目录），mv命令将文件重命名或将其移至一个新的目录中。当第二个参数类型是文件时，mv命令完成文件重命名，此时，源文件只能有一个（也可以是源目录名），它将所给的源文件或目录重命名为给定的目标文件名。当第二个参数是已存在的目录名称时，源文件或目录参数可以有多个，mv命令将各参数指定的源文件均移至目标目录中。

（3）常用参数：

-b ： 若需覆盖文件，则覆盖前先行备份。

-f ： 如果目标文件已经存在，不会询问而直接覆盖；

-I ： 若目标文件已经存在，则会先询问是否覆盖；

-u ：若目标文件已经存在，且目标文件比较新，则会更新

（4） 常用范例：

例一：文件改名。

命令：mv yourfile.txt myfile.txt

例二：将将文件log1.txt,log2.txt,log3.txt移动到目录test1中。

命令：mv log1.txt log2.txt log3.txt est1

1. **cat命令**

cat命令的用途是连接文件或标准输入并打印。这个命令常用来显示文件内容，或者将几个文件连接起来显示，或者从标准输入读取内容并显示，它常与重定向符号“>”配合使用。

（1）命令格式：

Cat [选项] [文件]...

（2）命令功能：cat主要有三大功能：

a.一次显示整个文件:cat filename

b.从键盘创建一个文件:cat > filename 只能创建新文件，不能编辑已有文件.

c.将几个文件合并为一个文件:cat file1 file2 > file

（3）常用参数：

-b, --number-nonblank 对非空输出行编号

-n, --number 对输出的所有行编号,由1开始对所有输出的行数编号

（4） 常用范例：

例一：查看 file1.txt 文件的内容

命令：cat file1.txt

例二：查看 file1.txt 文件的内容，要求每行显示行号

命令：cat –n file1.txt

例三：把 file1.txt 的文件内容加上行号后输到 file3.txt 这个文件里。

命令：cat –n file1.txt>file3.txt

1. **more命令**

more命令，功能类似 cat ，cat命令是整个文件的内容从上到下显示在屏幕上。more会一页一页的显示以方便使用者逐页阅读，而最基本的指令就是按空白键【space】就往下一页显示，按【b】键就会往回一页显示，而且还有搜寻字串的功能。

（1）命令格式：

More [-dlfpcsu ] [-num ] [+/ pattern] [+ linenum] [file ... ]

（2）命令功能：

more命令和cat的功能一样都是查看文件里的内容，但有所不同的是more可以按页来查看文件的内容，还支持直接跳转行等功能。

（3）常用参数：

+n 从笫n行开始显示

-n 定义屏幕大小为n行

-c 从顶部清屏，然后显示

-d 提示“Press space to continue，’q’ to quit（按空格键继续，按q键退出）”，禁用响铃功能

-u 把文件内容中的下画线去掉

（4）常用操作命令：

Enter 向下n行，需要定义。默认为1行

Ctrl+F 向下滚动一屏

空格键 向下滚动一屏

Ctrl+B 返回上一屏

= 输出当前行的行号

：f 输出文件名和当前行的行号

V 调用vi编辑器

!命令 调用Shell，并执行命令

q 退出more

（5） 常用范例：

例一：显示 file1.txt 文件从第3行起的内容

命令：more +3 file1.txt

例二：从 file1.txt 文件中查找第一个出现“OK”的行，并从该行前两行开始显示。

命令：more +/OK file1.txt

1. **grep命令**

grep命令是一种强大的文本搜索工具，它能使用正则表达式搜索文本，并把匹配的行打印出来。grep全称是Global Regular Expression Print，表示全局正则表达式版本，它的使用权限是所有用户。

grep的工作方式是这样的，它在一个或多个文件中搜索字符串模板。如果模板包括空格，则必须被引用，模板后的所有字符串被看作文件名。搜索的结果被送到标准输出，不影响原文件内容。

（1）命令格式：grep [option] pattern file

（2）命令功能：

用于过滤/搜索的特定字符。可使用正则表达式能多种命令配合使用，使用上十分灵活。

（3）常用参数：

-a --text #不要忽略二进制的数据。

-i --ignore-case #忽略字符大小写的差别。

-f<规则文件> --file=<规则文件> #指定规则文件，其内容含有一个或多个规则样式，让grep查找

符合规则条件的文件内容，格式为每行一个规则样式。

-n --line-number #在显示符合样式的那一行之前，标示出该行的列数编号。

-v --revert-match #显示不包含匹配文本的所有行。

（4）正则表达式：

^ #锚定行的开始 如：'^grep'匹配所有以grep开头的行。

$ #锚定行的结束 如：'grep$'匹配所有以grep结尾的行。

. #匹配一个非换行符的字符 如：'gr.p'匹配gr后接一个任意字符，然后是p。

\* #匹配零个或多个先前字符 如：'\*grep'匹配所有一个或多个空格后紧跟grep的行。

.\* #一起用代表任意字符。

[] #匹配一个指定范围内的字符，如'[Gg]rep'匹配Grep和grep。

[^] #匹配一个不在指定范围内的字符，如：'[^A-FH-Z]rep'匹配不包含A-R和T-Z的一个字母开头，紧跟rep的行。

\(..\) #标记匹配字符，如'\(love\)'，love被标记为1。

\< #锚定单词的开始，如:'\<grep'匹配包含以grep开头的单词的行。

\> #锚定单词的结束，如'grep\>'匹配包含以grep结尾的单词的行。

x\{m\} #重复字符x，m次，如：'0\{5\}'匹配包含5个o的行。

x\{m,\} #重复字符x,至少m次，如：'o\{5,\}'匹配至少有5个o的行。

x\{m,n\} #重复字符x，至少m次，不多于n次，如：'o\{5,10\}'匹配5--10个o的行。

\w #匹配文字和数字字符，也就是[A-Za-z0-9]，如：'G\w\*p'匹配以G后跟零个或多个文字或数字字符，然后是p。

\W #\w的反置形式，匹配一个或多个非单词字符，如点号句号等。

\b #单词锁定符，如: '\bgrep\b'只匹配grep。

（5） 常用范例：

例一：显示 file1.txt 文件中含有关键字OK的行的行号和内容。

命令：grep –n OK file1.txt

### 四、Linux的在线帮助man

Linux提供了丰富的帮助手册，当你需要查看某个命令的参数时不必到处上网查找，只要man（man 为manual的简写）一下即可。

1. **简单范例**

例如，如果你不清楚pwd命令的用法，你可以在命令提示符下直接输入命令：man pwd，马上就会有pwd的详细资料提供给你：

**PWD（1） User commands Date(1)**

**NAME**

pwd - print name of current/working directory

**SYNOPSIS**

pwd [OPTION]

**DESCRIPTION**

NOTE: your shell may have its own version of pwd which will supercede the version described here. Please refer to your shell's documentation for details about the options it supports.

Print the full filename of the current working directory.

**--help**

display this help and exit

**--version**

output version information and exit

**AUTHOR**

Written by Jim Meyering.

**REPORTING BUGS**

Report bugs to <bug-coreutils@gnu.org>.

**COPYRIGHT**

Copyright © 2004 Free Software Foundation, Inc.   
This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

**SEE ALSO**

The full documentation for pwd is maintained as a Texinfo manual. If the info and pwd programs are properly installed at your site, the command

info coreutils pwd

should give you access to the complete manual.

1. **man page 说明**

前面的范例中，第一行名字后的数字：

"1"表示用户命令

"2"表示系统调用

"3"表示C语言库函数

"4"表示设备或特殊文件

"5"表示文件格式和规则

"6"表示游戏及其他

"7"表示宏、包及其他杂项

"8"表示系统管理员相关的命令

可见，man不仅可以查询命令，还可以查询系统调用、C语言库函数、配置文件的格式、系统管理员可用的管理命令等。值得注意的是man是按照手册的章节号的顺序进行搜索的，比如：

man sleep

只会显示sleep命令的手册，如果想查看库函数sleep，就要输入使用man 3 sleep。

若想知道sleep系统调用需要哪些头文件，则要输入man 2 sleep

通常，man page大致分几个部分：

NAME 简短的命令、数据名称说明

SYNOPSIS 简短的命令语法简介

DESCRIPTION 较为完整的说明，这部分最好仔细看看、

OPTIONS 针对SYNOPSIS部分中，列举说明所有可用的参数

COMMANDS 当这个程序（软件）在执行的时候，可以在此程序（软件）中发出的命令

FILES 这个程序或数据所使用、参考或连接的某些参考说明

SEE ALSO 与这个命令或数据相关的其他参考说明、

EXAMPLE 一些可以参考的范例

BUGS 是否有相关的错误

1. **man page中可以使用的常用按键**

在man中的按键使用：

空格键 向下翻一页

[Page Down] 向下翻一页

[Page Up] 向上翻一页

[Home] 到第一页

[End] 到最后一页

/word 向下搜索word字符串，如果要搜索date的话，就输入/date

?word 向上搜索word字符串

n,N 使用/或?来搜索字符串时，可以用n来继续下一个搜索（不论是/还是?），

可以使用N来进行“反向”搜索。

举例来说，我以/date搜索date字符串，那么可以用n继续往下查询，用N往上查询。若以?date向上查询date字符串，可以用n继续 “向上”查询，用N反向查询

q 结束并退出man page

1. **互联网上的在线Linux man手册**

即使不在Linux下，也可以通过某些网站在线查询某个Linux的命令，如：

<http://www.linuxmanpages.com>[/](http://www.linuxmanpages.com/)

在这里有非常全的Linux的man信息，你可以分1－8来查看相应的manual 。

### 五、vi和vim编辑器

在计算机系统中，编辑文本文件是用户经常要进行的操作。所谓文本文件指的是由ASCII码字符构成的文件。vi编辑器是Unix/Linux系统提供的文本编辑器，用于创建和修改文本文件。vi编辑器与其他字处理软件不同，它不包含任何格式方面的信息，如粗体、居中或者下划线等。

vi编辑器是一个全屏幕编辑器，用户可以在整个文档范围内自由移动光标进行编辑操作。vi编辑器中有100多个命令可供用户使用，提供了丰富的编辑功能，当然对于学习使用者来说也是个挑战。但是不必灰心，因为只有少数一些命令是必须使用，或者使用频繁的，所以只要熟练掌握这些常用命令就可以完成大部分文本文件的编辑任务了。

vim可以当做vi的升级版，vi的命令几乎都可以在vim上使用。Vim会依据文件扩展名或文件的开头信息来判断文件内容，而自动执行该程序的语法判断，再以颜色来显示程序代码和一般信息。因此vim用于程序编辑更加方便。

在系统提示符（$、#）下，输入：**vi <文件名>**，vi可以自动载入所要编辑的文件或创建一个新文件（若该文件不存在）。

1. **vi的三种工作模式**

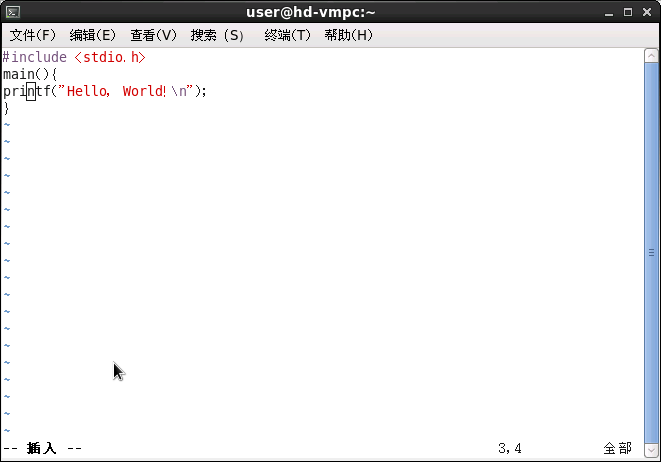
vi编辑器有三种工作模式：一般模式、编辑模式和命令模式：

**（1）一般模式。**以 vi 打开一个文件就直接进入一般模式了(这是默认的模式)。在这个模式中， 你可以使用『上下左右』按键来移动光标，你可以使用『删除字符』戒『删除整行』来处理文件内容， 也可以使用『复制、贴贴』来处理你的文件数据。

即在一般模式中可以进行删除、复制、粘贴等动作，但却无法编辑文件的内容。

**（2）编辑模式**。在一般模式下，按下『i, I, o, O, a, A, r, R』等任何一个字母后就会进入编辑模式。通常，在按下这些字母后，在画面的左下方会出现『 INSERT 或 REPLACE 』的字样，如图5-1所示，此时才可以进行编辑。而要回到一般模式，则必须按下『 ESC 』按键，即可退出编辑模式，返回一般模式。

**（3）命令模式**：在一般模式当中，输入『：、/ 、？』三个字符中的任一个，就可以将光标移动到最底下的那一行，进入命令模式。在命令模式中，可以提供查找字符串的操纵，而读取、保存、大量字符替换、离开vi、显示行号等操作也是在这模式下完成的。



**图5-1 vi编辑器**

1. **使用范例**

* **范例1**：

（1）在Linux命令行界面下输入命令： vi ccc.c 后便进入vi的一般模式。如图5-1所示，vi的界面分为上下两部分，上半部分显示的是文件的实际内容，而下半部，即最下面的一行则显示一些状态信息。如果ccc.c是一个原来不存在的文件，状态行中会显示『“ccc.c” [New File] 』表示它是一个新文件，否则，会显示出被编辑文件的行数、字符数等信息。

（2）按 “i”进入编辑模式，此时，状态行中会出现『 INSERT』的字样，便可以开始编辑文字了。此时，你输入的除了“ESC”以外的所有信息都被视为文件的内容，比如，如图5.1那样，你可以输入：

#include <stdio.h>

main(){

printf(“Hello,World!\n”);

}

（3）按“ESC”回到一般模式，此时，状态行中的『 INSERT』不见了。

（4）在一般模式中输入“：wq”保存文件的内容并离开vi。

* **范例2**

（1）在Linux命令行界面下输入命令： vi file1.txt

（2）按 “i”进入编辑模式，输入下列字符，并保存文件。

You raise me up, so I can stand on mountains;

You raise me up, to walk on stormy seas;

I am strong, when I am on your shoulders;

You raise me up: To more than I can be.

* **范例3**

通过vi编辑器编辑一个systemcall.c文件，其内容如下：

#include <fcntl.h>

#include <stdio.h>

int main(){

int fd=0，i;

char buf[10];

fd=open("file1.txt",O\_RDONLY);

if(fd == -1) printf("Cannot open file!\n");

while((i=read(fd,buf,sizeof(buf)-1))>0){

buf[i]='\0';

printf("%s",buf);

}

}

（3）按“ESC”回到一般模式。

（4）在一般模式中输入“：wq”保存文件的内容并离开vi。

1. **一般模式下的常用按键**

**（1）光标移动**

vi可以直接用键盘上的光标键来上下左右移动，但正规的vi是用小写英文字母 。

h、j、k、l，分别控制光标左、下、上、右移一格。

按Ctrl+B：屏幕往后移动一页。[常用]

按Ctrl+F：屏幕往前移动一页。[常用]

按Ctrl+U：屏幕往后移动半页。

按Ctrl+D：屏幕往前移动半页。

按 0 (数字零)：移动文章的开头。[常用]

按 G：移动到文章的最后。[常用]

按 w：光标跳到下个单词的开头。[常用]

按 e：光标跳到下个单词的字尾。

按 b：光标回到上个单词的开头。

按 $：移到光标所在行的行尾。[常用]

按 ^：移到该行第一个非空白的字符。

按 0：移到该行的开头位置。[常用]

按 #：移到该行的第#个位置，例：51、121。[常用]

**（2）删除**

x：每按一次删除光标所在位置的后面一个字符。[超常用]

#x：例如，6x 表删除光标所在位置的后面6个字符。[常用]

X：大字的X，每按一次删除光标所在位置的前面一个字符。

#X：例如，20X 表删除光标所在位置的前面20个字符。

dd：删除光标所在行。[超常用]

#dd：例如，6dd表删除从光标所在的该行往下数6行之文字。[常用]

**(3)复制 与粘贴**

yw：将光标所在处到字尾的字符复制到缓冲区中。

yy：复制光标所在行。[超常用]

#yy：如：6yy表示拷贝从光标所在的该行往下数6行之文字。[常用]

p：将已复制的内容粘贴在光标后的位置

P：将已复制的内容粘贴在光标前的位置

**(4)替换**

r： 取代光标所在处的字符：[常用]

R：取代字符直到按Esc为止。

**(5)撤销和重做**

u：假如您误操作一个指令，可以马上按u，可撤销前一个操作。[超常用]

U：撤销当前行上最近的所有操作。

ctrl+r：重做上一个操作。

.： 点号可以重复执行上一次的指令。

**(6)更改**

cw：更改光标所在处的字到字尾$处。

c#w：例如，c3w代表更改3个字。

3. 退出vi编辑器

**(7)切换到编辑模式**

i ：进入插入模式，在当前光标前插入

I ：进入插入模式，在当前行首插入

a：进入插入模式，在当前光标后插入

A ：进入插入模式，在当前行尾插入

o：进入插入模式，在当前行之下新开一行

O：进入插入模式，在当前行之上新开一行

r：进入替换模式，替换当前字符

R：进入替换模式，替换当前字符及其后的字符，直至按ESC键

1. **命令模式下的常用命令**

在一般模式下，按“:”便可切换到命令模式，然后可以使用下述命令模式下的命令：

**q命令：**在没有任何修改操作发生的情况下，该命令可以退出vi编辑器。

**q!命令：**不保存文件，强制退出vi编辑器。

**w命令：**保存文件。

**wq命令：**保存文件，然后退出vi编辑器

**w[文件名] 命令：**将编辑后的文件另存到指定文件中。

**r[文件名] 命令：**将指定文件的内容读入，并添加到当前文件光标所在的位置后面。

### 六、Linux环境下C编程

1. **GCC概述**

GCC是linux下最常用的编译器，也能运行在unix、solaris、windows等下，支持多种语言的编译，如C、C++、Object C等语言编写的程序。

gcc对文件的处理需要经过预处理->编译->汇编->链接的步骤，从而产生一个可执行文件。预处理阶段主要是在库中寻找头文件，包含到待编译的文件中；编译阶段检查程序的语法；汇编阶段将源代码翻译成机器语言；链接阶段将所有的目标代码连接成一个可执行程序。各阶段对应不同的文件类型，具体如下：  
 file.c c程序源文件  
 file.i c程序预处理后文件  
 file.cxx c++程序源文件，也可以是file.cc / file.cpp / file.c++  
 file.ii c++程序预处理后文件  
 file.h c/c++头文件  
 file.s 汇编程序文件  
 file.o 目标代码文件

1. **GCC使用格式：**

gcc [参数选项] [文件名]

其中文件名是要编译的文件名称。

1. **GCC遵循的部分后缀越大规则：**

当调用gcc时，gcc根据待编译文件的扩展名（后缀）自动识别文件的类别，并调用对应的编译器。Gcc遵循的部分后缀约定规则如表所示：

|  |  |
| --- | --- |
| **后缀** | **约定规则** |
| .c | C语言源代码文件 |
| .a | 由目标文件构成的档案库文件 |
| .C .cc .cxx | C++源代码文件 |
| .h | 程序包含的头文件 |
| .i | 已经预处理过的C源代码文件 |
| .ii | 已经预处理过的C++源代码文件 |
| .m | Objective-C源代码文件 |
| .o | 编译后的目标文件 |
| .s | 汇编语言源代码文件 |
| .S | 经过预编译的汇编语言源代码文件 |

1. GCC的参数说明：

GCC参数很多，最常用的参数如下：

-c 仅编译或汇编，生成目标代码文件，将.c、.i、.s等文件生成.o文件，其余文件被忽略  
-S 仅编译，不进行汇编和链接，将.c、.i等文件生成.s文件，其余文件被忽略  
-E 仅预处理，并发送预处理后的.i文件到标准输出，其余文件被忽略  
-o file 创建可执行文件并保存在file中，而不是默认文件a.out  
-g 产生用于调试和排错的扩展符号表，用于GDB调试，切记-g和-O通常不能一起使用  
-w 取消所有警告  
-W 给出更详细的警告  
-O [num] 优化，可以指定0-3作为优化级别，级别0表示没有优化  
-x language 默认为-x none，即依靠后缀名确定文件类型，加上-x lan确定后面所有文件类型，直到下一个-x出现为止  
-I dir 将dir目录加到搜寻头文件的目录中去，并优先于gcc中缺省的搜索目录，有多个-I选项时，按照出现顺序搜索  
-L dir 将dir目录加到搜索-lname选项指定的函数库文件的目录列表中去，并优先于gcc缺省的搜索目录，有多个-L选项时，按照出现顺序搜索  
-lname 在链接时使用函数库libname.a，链接程序在-L dir指定的目录和/lib、/usr/lib目录下寻找该库文件，在没有使用-static选项时，如果发现共享函数库libname.so，则使用libname.so进行动态链接   
-fPIC 产生位置无关的目标代码，可用于构造共享函数库  
-static 禁止与共享函数库链接  
-shared 尽量与共享函数库链接（默认）



1. **简单范例1**

（1）按“**<五>vi编辑器**”的简单范例1，先通过vi编辑器编辑好文件ccc.c。

（2）使用Linux命令行界面，在命令提示符后，输入命令：

gcc ccc.c

功能：对源程序ccc.c进行编译链接，产生对应的可执行文件，文件名缺省为：a.out。也可以通过选项-o在编译命令中指定可执行文件名，如：

gcc -o ccc ccc.c

将产生可执行文件ccc，而不是默认的a.out。

（3）在命令提示符后，输入命令：./a.out 或 ./ccc 运行当前目录下可执行文件a.out（或ccc）。

上述范例的运行结果如图6-1所示。

**图6-1 ccc.c的编译和运行**



1. **简单范例2**

（1）先按“**<五>vi编辑器**”的简单范例2和简单范例3通过vi编辑器编辑好文件file1.txt，和systemcall.c；然后用Gcc编译器对systemcall.c进行编译，生成可执行文件，运行之。

（2）阅读上述C源程序，分析它到底是完成什么功能的；再观察程序的运行结果来验证你的理解是否正确。

（3）思考上述程序中文件打开操作和读文件操作到底是谁具体完成的，并用man查阅open，read，printf，了解它们的具体功能、函数参数和返回值等信息，并注意它们属于系统调用，还是库函数，以加速对系统调用概念的理解。

1. **简单范例3**

编译多个源文件。

1. 使用vi/vim编辑器编写两个文件：message.c，main.c：

vi message.c

vi main.c

1. 使用gcc编译：

gcc -c message.c //输出message.o文件，是一个已编译的目标代码文件

gcc -c main.c //输出main.o文件

gcc -o all main.o message.o //执行连接阶段的工作，然后生成可执行文件all

1. 执行可执行文件：

./all

注意：：gcc对如何将多个源文件编译成一个可执行文件有内置的规则，所以前面的多个单独步骤可以简化为一个命令：

gcc -o all message.c main.c

### 七、gdb调试工具

gdb是一个GNU调试工具，可以调试C和C++程序，其主要功能有：1）监视程序中变量的值；2）设置断点；3）单步执行程序。

为了能够使用gdb调试程序，必须在编译时包含调试信息，即使用gcc编译时需要加上“-g”选修，如gcc –g –o test test.c。

gdb命令很多，下面列出一些常用的调试命令：

**1、gdb启动及退出：**

**（1）启动gdb命令：**

gdb exefilename

其中exefilename是可执行文件名，如果没有指定运行程序，也可进入gdb后再用file命令装入文件。

gdb启动成功后，提示符为：（gdb），随后可以输入gdb命令对程序进行调试。

（2）退出gdb命令：(gdb)quit

**2、断点管理命令：**

**（1）设置断点：**

break命令（可简写为b）可以用来在调试的程序中设置断点，该命令有如下四种形式：   
(gdb)break line-number 使程序在执行给定行之前停止。   
(gdb)break function-name 使程序在进入指定的函数之前停止。   
(gdb)break line-or-function if condition 如果condition（条件）是真，程序到达指定行或函数时停止。   
(gdb) break routine-name 在指定例程的入口处设置断点   
如果该程序是由很多原文件构成的，你可以在各个原文件中设置断点，而不是在当前的原文件中设置断点，其方法如下：   
(gdb) break filename:line-number   
(gdb) break filename:function-name

**（2）从断点处继续执行：**

(gdb) countinue

**（3）显示当前gdb的断点信息：**   
(gdb) info break

**（4）删除指定的某个断点：**   
(gdb) delete breakpoint 1   
该命令将会删除编号为1的断点，如果不带编号参数，将删除所有的断点：   
(gdb) delete breakpoint

**（5）禁止使用某个断点**   
(gdb) disable breakpoint 1   
该命令将禁止断点 1

**（4）允许使用某个断点**   
(gdb) enable breakpoint 1   
该命令将允许断点 1

**（5）清除原文件中某一代码行上的所有断点**   
(gdb)clean number   
注：number 为原文件的某个代码行的行号

**3、显示信息命令：**

**（1）print命令：**  
利用print 命令可以检查各个变量的值。   
(gdb) print p //(p为变量名或表达式)

(gdb) print 开始表达式@连续内存空间大小

打印内存中某一连续内存空间的值，主要用于打印数组类型的表达式的值。

(gdb) print 程序中的某一函数调用

如：(gdb) print find\_entry(1,0)

**（2）whatis 命令**：可以显示某个变量的类型   
(gdb) whatis p //(p为变量)  
type = int \*

**（3）display命令：设置要显示的表达式**

(gdb)display 表达式

当程序运行到断点时，显示该表达式的值。

(gdb)info display：显示所有要显示表达式的值。

(gdb)undisplay 表达式：结束已设置的表达式。

（4）awatch命令：设置要监视的表达式

(gdb)awatch 表达式

当表达式的值变化或被读取时，程序暂停，显示表达式的值。

**4、程序运行控制命令：**

（1）run：运行程序

（2）kill：结束程序的调试运行

（3）cont：继续执行程序

（4）next：单步运行，不进入子程序

（5）step：单步运行，进入子程序

**5、文件命令：**

（1）file命令：加载调试文件

file 文件名

（2）list命令：列出文件内容

list [参数]

参数说明：

参数为空：从上次显示的最后一行或附近开始，显示10行

<行号>：从当前文件的该行开始显示

<文件名> <行号>：从指定文件的指定行开始显示

<函数名>：显示指定的函数

<文件名> <函数名>：指定文件的指定函数

<行号1><行号2>：从行号1显示到行号2

**6、堆栈相关命令：**

（1）backtrace命令：可简写为bt，显示栈中内容。

命令：bt: 显示当前函数调用栈的所有信息。

命令：bt <n>: n是一个正整数，表示只打印栈顶上n层的栈信息。

n是一个负整数，表示只打印栈底下n层的栈信息。

（2）frame命令：可简写为f，显示当前栈层的信息：

显示的内容有：栈的层编号，当前的函数名，函数参数值，函数所在文件及行号，函数执行到的语句。

info frame（或info f）：出更为详细的当前栈层的信息，只不过，大多数都是运行时的内内地址。比如：函数地址，调用函数的地址，被调用函数的地址，目前的函数是由什么样的程序语言写成的、函数参数地址及值、局部变量的地址等等。