## - Xác định thư mục hiện thời: pwd

## - Xem nội dung thư mục: ls [đường dẫn]

## - Tạo thư mục: mkdir đường dẫn/tên thư mục

## - Tạo tệp tin rỗng: touch tên tệp

## - Hiển thị nội dung tệp:

+ cat tên tệp: xem toàn bộ nội dung

+ more tên tệp: xem từng trang, phím space để chuyển trang

+ head -n số tên tệp: xem n dòng đầu tiên

+ tail -n số tên tệp: xem n dòng cuối cùng

## - Ghi dữ liệu vào tệp: ls > tên tệp

## - Trộn nội dung nhiều tệp tin vào một tệp tin: cat tên các tệp đầu vào > tên tệp đầu ra

### - Tạo tệp tin:

+ echo [nội dung] > filename

+ cat > filename sau đó nhập nội dung rồi Ctrl + d

### - Đếm các thông số trong tệp:

Dùng lệnh wc (word count) để đếm số dòng, số từ, số kí tự trong tệp tin.

Cú pháp: wc [Option] filename

Trong đó tham số Option:

• -l: Đếm số dòng

• -w: Đếm số từ

• -c: Đếm số kí tự

• -L: In ra chiều dài của dòng dài nhất trong tệp

## **Sao chép tệp tin hoặc thư mục** Để sao chép tệp tin hoặc thư muc từ thư mục nguồn đến thư mục đích, dùng lệnh **cp** Cú pháp: *cp [Option] source destination* Trong đó: *•* source: là đường dẫn đến tệp tin hoặc thư mục cần sao chép *•* Nếu là thư mục, lựa chọn Option là –r

## **Đổi tên thư mục/tệp tin** Để đổi tên thư mục/tệp tin, dùng lệnh **mv** (*move*) Cú pháp: *mv filename newfilename* Trong đó: *•* filename: là đường dẫn đến tệp tin/thư mục muốn đổi tên *•* newfilename: là tên mới muốn đổi

## **Di chuyển thư mục/tệp tin** Để di chuyển tệp tin hoặc thư muc từ thư mục nguồn đến thư mục đích, ta dùng lệnh **mv** (*move*) Cú pháp: *mv path1 path2 ... pathN destination* Trong đó: *•* path1 path2 ... pathN: là danh sách đường dẫn đến các thư mục hoặc tệp tin cần di chuyển. Nếu muốn sao chép tất cả các tệp tin/thư mục dùng kí hiệu "\*" *•* destination: là đường dẫn của tệp tin hay thư mục đích

## **Xóa thư mục** Để xóa một thư mục, ta có thể dùng các lệnh sau: *•* **rmdir** (*remove directory*) Cú pháp: *rmdir [Option] directory* Trong đó: **–** directory: là tên thư mục cần xóa. **–** Option: là lựa chọn có thể có hoặc không. Nếu không có tham số thì <directory> là thư mục rỗng. Ví dụ: *student@linux ∼ $ rmdir temp*

## *•* **rm** (*remove*) Cú pháp: *rm [Option] file1*. Trong đó: **–** file1 là tên tệp tin hoặc thư mục cần xóa. **–** Option là lựa chọn có thể có hoặc không. Nếu tham số là -d thì file1 là thư mục rỗng, nếu tham số là -r thì xóa được tất cả các thư mục con. **Tạo *link* (liên kết) với tệp tin** Trong Linux, mỗi tên file thực chất là một *link* ới một vùng dữ liệu trên bộ nhớ. Ví dụ: echo “This is a file” > file1.txt cat file1.txt Kết quả: This is a file

Có thể có nhiều link tới cùng một vùng dữ liệu. Để tạo 1 *link* tới vùng dữ liệu tương  
ứng với file đã có dùng lệnh **link** hoặc **ln**:  
Cú pháp: *link filename linkName*  
*ln fileName linkName*  
Ví dụ: link file1.txt file2.txt  
cat file2.txt  
Kết quả: This is a file

Liên kết trên gọi là liên kết cứng (hard link). Ngoài ra còn có liên kết mềm (soft  
link) tương ứng với tên tắt của một tệp tin hay thư mụ. Để tạo liên kết mềm dùng  
lệnh **ln**:  
*ln -s fileName linkName*  
Ví dụ: link -s file1.txt file2.txt  
cat file2.txt Kết quả: This is a file

## **Quyền truy cập tệp tin và thư mục**

|  |  |  |
| --- | --- | --- |
| Quyền | Trên file | Trên thư mục |
| r (đọc) | Đọc nội dung file (cat) | Đọc nội dung của thư mục (ls) |
| w (ghi) | Thay đổi nội dung file | Tạo mới file trong thư mục (touch) |
| x (thực thi) | Sử dụng fileName như một lệnh trong Linux | Thay đổi thư mục (cd) |

Để thay đổi quyền truy cập đối với file và thư mục, dùng lệnh **chmod**  
Cú pháp: chmod [options] files  
**chmod** chấp nhận 2 dạng tùy chọn [options]:  
*•* Các quyền truy cập được xác định dưới dạng chuỗi ký tự: u (user), g(group),  
o (other), a (all), w(write), x(excute), +(thêm quyền), - (xóa quyền), = (gán  
quyền)  
Ví dụ: *student@linux ∼ $ chmod ug=rw,o-rw,a-x \*.txt*  
Câu lệnh trên cài đặt quyền cho tất cả các file kết thúc bằng *\*.txt* thành *rw-rw*  
*- - - -*  
*•* Các quyền truy cập được xác định bởi các số bát phân. Ánh xạ các quyền vào  
các số bát phân được thể hiện như sau:

|  |  |  |
| --- | --- | --- |
| Quyền | Số bát phân | Số nhị phân |
| - - - | 0 | 000 |
| - -x | 1 | 001 |
| -w- | 2 | 010 |
| -wx | 3 | 011 |
| r- - | 4 | 100 |
| r-x | 5 | 101 |
| rw- | 6 | 110 |
| rwx | 7 | 111 |

Linux sử dụng bộ ba số bát phân viết liền nhau lần lượt biểu diễn quyền truy  
cập của người sở hữu, nhóm người sở hữu tệp tin/thư mục, những người dùng  
và nhóm người dùng khác.

## **Kiểm tra kết nối mạng**

Để kiểm tra kết nối mạng trong Linux, dùng lệnh **ping**. Lệnh này kiểm tra máy  
tính có thể kết nối với Internet hay một địa chỉ IP cụ thể nào đó hay không.  
Lệnh **ping** sẽ gửi một gói dữ liệu yêu cầu tới một máy chủ và chờ đợi phản hồi, nó  
sẽ duy trì gửi các gói tin cho đến khi kết thúc.  
Cú pháp: *ping [option] host*  
Trong đó:  
2  
*•* option: Là các tùy chọn.  
**–** -b: ping một địa chỉ broadcast  
**–** -c N: gửi N gói tin  
**–** -q: chế độ im lặng, hiển thị các gói tin đầu và cuối  
*•* host: Địa chỉ máy chủ hoặc địa chỉ IP cụ thế nào đó.  
Ví dụ: *student@linux ∼$ ping -c 4 google.com*

## **Xem cấu hình mạng**

Để các máy có thể giao tiếp được với nhau trong mạng, thiết bị dùng làm phương  
tiện giao tiếp đó là Card giao tiếp mạng (network card). Để xem các thông tin và  
cấu hình mạng của máy hiện tại, dùng lệnh **ifconfig**.  
Cú pháp: *ifconfig <Giao diện> [địa chỉ] [up/down] [arp | -arp][ broadcast*  
*<địa chỉ>] [netmask <mặt nạ mạng>]*  
Trong đó,  
*•* <Giao diện>: là tên của thiết bị giao tiếp mạng, chẳng hạn eth0 cho card  
mạng đầu tiên, eth1 cho card mạng thứ hai.  
*•* <địa chỉ>: địa chỉ mạng sẽ gán cho giao diện này.  
*•* up/down: tùy chọn này sẽ kích hoạt/tắt giao diện được chỉ ra.  
*•* arp | -arp: cho phép cấm giao thức ARP trên giao diện này.  
*•* broadcast <địa chỉ>: xác định địa chỉ quảng bá cho giao diện này  
*•* netmask <mặt nạ mạng>: xác định mặt nạ cho giao diện này.  
Ví dụ, để xem cấu hình mạng của máy hiện tại ta dùng lệnh:  
*student@linux ∼$ ifconfig*  
Ví dụ, để xem các thông tin về một card mạng nào đó, dùng lệnh:  
*student@linux ∼$ ifconfig eth0*

## **Tìm kiếm DNS** Để tìm kiếm DNS, dùng lệnh **host**. Nhập vào tên miền khi muốn xem địa chỉ IP và ngược lại, nhập vào địa chỉ IP khi muốn xem tên miền đi kèm. Cú pháp: *host [tên miền/IP]* Ví dụ: *student@linux ∼$ host google.com*

## **Truy cập từ xa** Trong Linux, để truy cập từ xa có thể dùng các lệnh sau: *• ssh* là một lệnh để đăng nhập từ xa. Với yêu cầu hai máy đều phải cài ssh. Cú pháp: *ssh [machinename@hostname]* Ví dụ: *student@linux ∼ $ ssh test01@192.168.10.94* Sau đó nhập mật khẩu của máy cần đăng nhập từ xa vào và sử dụng *• rlogin* cũng có thể dùng để đăng nhập từ xa. Cú pháp: *rlogin [machinename@hostname]* Ví dụ: *student@linux ∼ $ rlogin test01@192.168.10.94* Để thoát khỏi đăng nhập từ xa, dùng lệnh *exit* hoặc *logout*.

## **Gửi thông báo** Để gửi thông báo cho người dùng khác trong cùng hệ thống, dùng lệnh **write**. Cú pháp: *write <tên user2> [tên trạm cuối]* Lệnh này chuyển thông báo từ người dùng user1 đến người dùng user2. *•* Nếu user2 không làm việc thì màn hình của user1 sẽ hiện ra thông báo: *"user2* *is not logged in"*. *•* Nếu user2 đang làm việc thì máy của user2 sẽ phát ra tiếng chuông và trên màn hình hiện ra thông báo: Message from user1 on tty17 at <giờ, phút> Khi đó, máy của user1 sẽ hiện màn hình để nhập thông báo. Kết thúc kết nối với người dùng user2 dùng tổ hợp phím *Ctrl + d*.

Ngoài ra, còn sử dụng lệnh **wall** để gửi thông báo đến tất cả những người dùng  
trong hệ thống.  
Cú pháp: *wall [message]*  
4  
Ví dụ: *student@linux ∼ $ wall*  
*Hello World*

## **Giao tiếp giữa các máy** Để giao tiếp giữa các máy, dùng lệnh **nc** (*netcat*). Cú pháp: *nc -l [port]* Computer1 muốn giao tiếp với computer2: *nc -l 1234* Computer2 muốn chấp nhận giao tiếp với computer1: *nc Computer1\_IP [port]* Ví dụ: *student1@linux ∼ $ nc -l 1234* *student1@linux ∼ $ nc 192.168.10.94 1234*

## **Biểu thức chính quy** 1. **Biểu thức chính quy** (*regular expression* - viết tắt *regex*) là một cú pháp đặc biệt được sử dụng để mô tả các mẫu ký tự. Trong Linux, biểu thức chính quy thường được sử dụng cùng với các chương trình: *grep, sed, vi, ...* để hỗ trợ các thao tác tìm kiếm và thay thế trong văn bản. Biểu thức chính quy là tập hợp các ký tự điều khiển (*metacharacter*) với những ý nghĩa đặc biệt và các ký tự thông thường. **Các ký tự điều khiển trong biểu thức chính quy**

|  |  |
| --- | --- |
| **Ký tự** | **Mô tả** |
| ˆ | Đánh dấu đầu dòng |
| $ | Đánh dấu cuối dòng |
| . | Biểu diễn ký tự bất kỳ |
| [] | Biểu diễn một ký tự bất kỳ trong cặp dấu [] |
| [ˆ] | Biểu diễn phép phủ định của nội dung trong [] |
| [*−*] | Biểu diễn một ký tự bất kỳ bên trong vùng được chỉ rõ bởi [] |
| () | Biểu diễn các ký tự trong () như một nhóm ký tự |
| *\ <* | Đánh dấu đầu từ |
| *\ >* | Đánh dấu cuối từ |
| *\b* | Đánh dấu biên từ |
| *\B* | Đánh dấu xâu rỗng không ở biên từ |
| *\n* | Biểu diễn ký tự/nhóm ký tự nằm giữa cặp () thứ n trước đó |
| ? | Phép lặp: Ký tự/nhóm ký tự đi trước có thể xuất hiện hoặc không |
| + | Phép lặp: Ký tự/nhóm ký tự đi trước xuất hiện *≥* 1 lần |
| *∗* | Phép lặp: Ký tự/nhóm ký tự đi trước xuất hiện *≥* 0 lần |
| *{n}* | Phép lặp: Ký tự/nhóm ký tự đi trước xuất hiện đúng n lần |
| *{n, }* | Phép lặp: Ký tự/nhóm ký tự đi trước xuất hiện *≥* n lần |
| *{n, m}* | Phép lặp: Ký tự/nhóm ký tự đi trước có thể xuất hiện từ n đến m lần |
| *|* | Phép hoặc |

Dùng *\* trước các ký tự điều khiển để sử dụng chúng như những ký tự thông  
thường.

## **Pipe** (giao tiếp giữa các tiến trình) Đôi khi các tiến trình cần trao đổi thông tin cho nhau để xử lý. Một cơ chế được sử dụng khá phổ biến trong Linux là *pipe*, sử dụng chỉ thị *|* cho phép đầu ra của tiến trình bên trái ký hiệu *|* là đầu vào của tiến trình bên phải. Ví dụ 1: *student@linux ∼ $ ls -R /etc | more* Kết quả: Nội dung của thư mục */etc* được hiển thị ra màn hình theo từng trang Ví dụ 2:*student@linux ∼ $ echo "hoa phong lan" | cat > hoa.txt* *student@linux ∼ $ cat hoa.txt* Kết quả: hoa phong lan

## **grep** là một công cụ hữu hiệu trong Linux để tìm kiếm trong nội dung tệp tin các dòng phù hợp với mẫu tìm kiếm. Cú pháp: *grep [options] ‘rexExp’ fileName* *grep [options] ‘rexExp’ fileName > ouputFile* (ghi nội dung tìm kiếm ra tệp *ouputFile*) Trong đó [options]: *• ‘rexExp’*: là biểu thức chính quy biểu diễn mẫu tìm kiếm. *•* [*options*]: **–** -i: Tìm kiếm không phân biệt hoa thường

**–** -v: Tìm các kết quả không chứa các xâu biểu diễn bởi ‘*rexExp*’  
**–** -w: Đưa ra các dòng có chứa xâu biểu diễn bởi ‘*rexExp*’ là một từ  
**–** -c: Đếm số dòng chứa xâu ‘*rexExp*’  
**–** -e: Đếm số lần xuất hiện của xâu ‘*rexExp*’ trong tệp tin  
**–** -A5: Đưa ra 5 dòng sau kết quả  
**–** -B5: In ra 5 dòng trước kết quả  
**–** -C5: In ra 5 dòng trước và sau kết quả  
Lệnh *grep* có thể sử dụng kết hợp với các lệnh khác: *ls, cat, echo, ...*

## **Trình soạn thảo vi**

### ***Các chế độ vi*** Để thực hiện các thao tác trong trình soạn thảo vi, có thể thực hiện bằng hai chế độ chính: *•* Chế độ soạn thảo: Văn bản được đưa vào trong tài liệu, bạn có thể chèn hoặc bổ sung văn bản. *•* Chế độ dòng lệnh: Trong chế độ này, bạn có thể thực hiện các thao tác đơn giản như tìm kiếm, ghi, thoát chương trình hay trộn tài liệu,... ngoại trừ việc nhập văn bản.

### ***Khởi động vi*** Khởi động vi dùng cú pháp: *vi filename* Ví dụ: *student@linux ∼ $ vi bai1.txt*

Chế độ dòng lệnh là chế độ mặc định khi bạn mở hoặc tạo tệp tin mới. Dùng  
phím **i** hoặc **a** để hiển thị chế độ soạn thảo. Từ chế độ soạn thảo, dùng phím  
ESC để hiển thị chế độ dòng lệnh.

### **Lưu và thoát** Để thoát khỏi chế độ của vi, sử dụng lệnh: **:q**. Để lưu tệp tin, sử dụng lệnh: **:w**. Khi muốn lưu và thoát, dùng cả hai lệnh: **:wq**. Khi muốn thoát mà không lưu thay đổi, dùng lệnh: **:q!** **Di chuyển con trỏ** Có thể dùng các phím đơn để di chuyển con trỏ trong vi, ngoài ra có thể dùng các phím mũi tên trên bàn phím. *•* h - sang trái 1 ký tự *•* l - sang phải 1 ký tự *•* k - lên 1 dòng *•* j - xuống dưới một dòng *•* w - sang phải 1 từ *•* b - sang trái 1 từ

### ***Chèn văn bản***

Trong chế độ dòng lệnh, **i** hoặc **a** cho phép bạn chèn thêm  
văn bản vào tài liệu. Một số đặc tính khác của việc chèn văn bản trong **vi**:  
*•* a chèn văn bản vào sau vị trí con trỏ hiện tại  
*•* A chèn văn bản vào cuối dòng  
*•* i chèn văn bản vào trước vị trí con trỏ hiện tại  
*•* o chèn văn bản vào dòng mới phía dưới con trỏ hiện tại  
*•* O chèn văn bản vào dòng mới phía trên con trỏ hiện tại  
*•* s xóa kí tự hiện thời và chèn văn bản  
*•* S xóa dòng hiện thời và chèn văn bản  
**Sao chép/dán**  
Để sao chép một dòng dùng lệnh **yy**. Để sao chép N dòng dùng lệnh **Nyy**  
Ví dụ: Sao chép toàn bộ dòng hiện thời: *yy*  
Để dán ta dùng lệnh **p**.  
**Ghi/mở văn bản**  
Để ghi văn bản, dùng câu lệnh **:w filename**.  
Ví dụ, ghi tài liệu hiện tại ra tệp có tên là newfile: *:w newfile*  
Để mở tệp tin ra đọc, dùng lệnh: *:r filename*  
**Tìm kiếm trong văn bản**  
Để tìm kiếm, ta dùng các lệnh sau:  
*•* ? tìm trở lên  
*•* / tìm trở xuống

Nhấn ‘n’ để di chuyển đến kết quả tìm kiếm tiếp theo.  
Ví dụ:  
*/hu* là tìm các xâu ‘hu’ trong văn bản.  
**Xóa văn bản**  
Sử dụng các lựa chọn sau:  
*•* x - xóa ký tự tại vị trí con trỏ  
*•* X - xóa ký tự bên trái con trỏ  
*•* dd - xóa 1 dòng  
*•* dw - xóa 1 từ  
*•* 3dw - xóa 3 từ  
*•* 5dd - xóa 5 dòng  
**Undo/redo**  
Sử dụng các lựa chọn:  
*•* r - redo  
*•* u - undo

**Quản lý tiến trình**  
**Hiển thị thông tin tiến trình**  
Để liệt kê các tiến trình đang thực thi, dùng lệnh **ps** (*process status*)  
Cú pháp: *ps [option]*  
Trong đó, các option là:  
*•* -a: hiển thị các tiến trình của user được liên kết tới tty  
*•* -e (-A): hiển thị thông tin về mỗi tiến trình  
*•* -f: hiển thị PID của tiến trình cha và thời điểm bắt đầu  
*•* -l: tương tự như -f  
*•* x: hiển thị các tiến trình ngoại trừ các tiến trình là controlling tty (Ví dụ:  
/sbin/mingetty tty\*)  
*•* u: dạng hiển thị hướng đến người dùng  
Lệnh *ps* có thể kết hợp với lệnh *grep* để tìm kiếm một tiến trình đang chạy. Ví dụ:  
Ví dụ: *student@linux ∼ $ ps aux* -> Liệt kê tất cả các tiến trình.  
Khi đó, các thông số của tiến trình sẽ được liệt kê gồm có: Chủ nhân của tiến trình  
(owner), mã số nhận diện tiến trình (PID), thời gian hiện sử dụng CPU (%CPU),  
mức chiếm dụng bộ nhớ của tiến trình (%MEM), trạng thái tiến trình (STAT) và  
các thông tin khác.  
Một số trạng thái của tiến trình thường gặp: R-đang thi hành, S-đang bị đóng,  
Z-ngừng thi hành, W-không đủ bộ nhớ, ...  
Ngoài ra, có thể dùng lệnh **top** để xem các thông số liên quan đến các tiến trình,  
đặc biệt là các thông tin sử dụng tài nguyên của các tiến trình đó.  
Cú pháp: *top [option]*  
Trong đó, các *option* là:  
*•* -u: Xem những tiến trình đang hoạt động dưới một tài khoản nào đó.  
*•* -p <PID>: Xem một tiến trình thông qua PID của tiến trình đó.  
*•* -c: Hiển thị đầy đủ dòng lệnh thay vì hiển thị tên lệnh tạo tiến trình  
*•* -d <time>: Thời gian tải lại các hiển thị liên quan đến lệnh top. Giá trị được  
tính theo giây. Mặc định là 5s.

Có thể dùng thêm lệnh **pgrep** để xem PID của một tiến trình trên hệ thống. Ví  
dụ, khi mở một tệp tin soạn thảo bằng emacs, bạn sẽ tìm kiếm PID của tiến trình  
này như sau:  
Ví dụ: *student@linux ∼ $pgrep emacs*

**Liệt kê các tiến trình**  
Để liệt kê các tiến trình theo dạng cây, dùng lệnh **pstree**  
Cú pháp: *pstree [option]*  
Trong đó, các *option* là:  
*•* -p: hiển thị PID  
*•* -h: tô đậm những tiến trình hiện hành và những tiến trình con cháu của tiến  
trình hiện hành  
*•* -a : chỉ ra tham số dòng lệnh. Nếu dòng lệnh của một quá trình được tráo đổi  
ra bên ngoài, nó được đưa vào trong dấu ngoặc đơn.  
*•* -c : không thể thu gọn các cây con đồng nhất. Mặc định, các cây con sẽ được  
thu gọn khi có thể  
*•* -H : giống như tùy chọn -h, nhưng quá trình con của quá trình hiện thời không  
có màu sáng trắng  
*•* -l : hiển thị dòng dài  
*•* -n : sắp xếp các quá trình cùng một tổ tiên theo chỉ số quá trình thay cho sắp  
xếp theo tên  
**Các chế độ chạy của tiến trình**  
Để quản lí các chế độ chạy của một tiến trình, có thể dùng các lệnh **&** hoặc **Ctrl**  
**C, Ctrl Z, fg, bg**  
*•* &: Cho tiến trình hoạt động ở trạng thái nền (*background*)  
Ví dụ: *student@linux ∼ $ ls -l -R/ > /home/student/list.txt* & -> ứng  
dụng ls sẽ chạy nền bên dưới  
Hoặc:  
Ví dụ: *student@linux ∼ $emacs*& -> ứng dụng emacs sẽ chạy nền, khi  
đó người sử dụng có thể dùng terminal để thực hiện các lệnh khác.  
*• Ctrl C* : Kết thúc tiến trình đang thực thi, sau khi ấn Ctrl C, có thể dùng lệnh  
jobs để hiển thị trạng thái của tiến trình đang chạy  
*• Ctrl Z* : Tạm ngừng tiến trình đang thực thi sau đó, có thể dùng các lệnh fg,  
bg để tiếp tục:  
**–** *bg*: tiếp tục tiến trình vừa tạm ngừng ở trạng thái nền (*background*)

**–** *fg*: tiếp tục tiến trình vừa tạm ngừng ở trạng thái hiện (*foreground*)  
? Thực thi lệnh liệt kê tất cả các tệp tin, thư mục có trong */home/student*? Kết  
quả lưu vào tệp tin list.txt. Sau đó chuyển lệnh trên vào chế độ bg? Tạm ngừng  
lệnh trên và cho phép thực thi lại?  
**Dừng một tiến trình**  
Lệnh **kill** thường được sử dụng để ngừng thi hành một tiến trình.  
Cú pháp: *kill [option] <PID>*  
Trong đó:  
*•* option là một lựa chọn:  
**–** -s : xác định tín hiệu được gửi. Tín hiệu có thể là số hoặc tên của tín hiệu.  
Dưới đây là một số tín hiệu hay dùng:  
*∗* SIGHUP(1): Hangup (gọi lại tiến trình)  
*∗* SIGINT(2): Interrupt (Ngắt từ bàn phím Ctrl  
*∗* SIGKILL(9): Hủy tiến trình ngay lập tức  
*∗* SIGTERM(15): Terminate – Kết thúc tiến trình, nhưng cho phép xóa  
các tệp tin tạm  
**–** -l : hiển thị danh sách các tín hiệu mà lệnh *kill* có thể gửi đến các quá  
trình (các tín hiệu này có trong file /usr/include/Linux/signal.h)  
*•* PID: mã số nhận diện tiến trình muốn dừng  
Lệnh *kill* có thể gởi bất kỳ tín hiệu *signal* nào tới một tiến trình, nhưng theo mặc  
định nó gởi tín hiệu 15, TERM (là tín hiệu kết thúc chương trình). Super-user mới  
có quyền dừng tất cả các tiến trình, còn người sử dụng chỉ được dừng các tiến trình  
của mình.

**Độ ưu tiên của một tiến trình**  
Để chạy chương trình với một độ ưu tiên nào đó, dùng lệnh **nice**  
Cú pháp: *nice –n <độ ưu tiên> <chương trình>*  
Trong đó, độ ưu tiên từ -20 (độ ưu tiên cao nhất) đến 19 (ưu tiên thấp nhất), độ  
ưu tiên mặc định là 0.  
Ví dụ: *student@linux ∼ $ nice -n 12 abcd*  
Để thay đổi độ ưu tiên của một tiến trình dùng lệnh *renice*  
Cú pháp: *renice <độ ưu tiên> [option]*  
Hoặc: *renice <độ ưu tiên> <pid>*  
Trong đó, option là:

*•* -g : thay đổi quyền ưu tiên theo nhóm người dùng  
*•* -p : thay đổi quyền ưu tiên theo chỉ số của quá trình  
*•* -u : thay đổi quyền ưu tiên theo tên người dùng  
Ví dụ: *student@linux ∼ $ renice 1 3456*

**Giới thiệu về Bash Shell Script**  
Shell là một chương trình dùng để tương tác giữa người dùng và máy tính. Thông  
dịch các lệnh của người dùng nhập vào hoặc từ các tệp tin.  
Shell Script: Là các chương trình shell gồm tập hợp các lệnh đặc biệt, từ đó có thể  
tạo nên chương trình.  
*•* **Các loại Shell trên Linux**  
Một số Shell trên Linux:  
**–** BASH (*Bourne-Again Shell*) phát triển bởi Brian Fox và Chet Ramey.  
Đây là Shell thông dụng nhất trên Linux.  
**–** CSH (*C Shell*) phát triển bởi Bill Joy tại University of California (dành  
cho BSD). Sử dụng cấu trúc lệnh giống C, rất thân thiện cho các lập trình  
viên C trên linux.  
**–** KSH (*Korn Shell*) phát triển bởi David Korn tại AT & T Bell Labs.  
**–** TCSH (*TENEX C Shell*) là phiên bản nâng cấp của C Shell.  
Để xem hệ thống đang chạy shell gì, dùng lệnh echo $SHELL  
*•* **Viết Bash Shell Script đơn giản**  
**–** Soạn thảo chương trình  
*∗* Sử dụng mọi trình soạn thảo dạng text: vi, emacs, ...  
*∗* Nội dung:  
+ Gồm các câu lệnh được sử dụng trên dòng lệnh của Linux  
+ Dòng đầu tiên chứa câu lệnh *#!/bin/bash* (Chỉ ra loại shell sử dụng)  
+ Để chú thích một đoạn mã: dùng # trước đoạn mã cần chú thích  
+ Các câu lệnh trên cùng một dòng phải phân tách bằng dấu;  
+ Lưu tệp tin với đuôi .sh  
**–** Thiết lập quyền thực thi cho chương trình shell  
Mặc định các tệp tin tạo ra chưa có quyền thực thi. Cần phải cấp quyền  
thực thi bằng lệnh *chmod*  
Ví dụ: *chmod u+x fileName*  
**–** Thực thi chương trình sử dụng một trong các lệnh:  
*∗* bash fileName  
*∗* sh fileName

*∗* ./fileName