TRƯỜNG ĐH KHOA HỌC TỰ NHIÊN HÀ NỘI
KHOA TOÁN – CƠ – TIN HỌC

BÀI THỰC HÀNH LINUX
Bài 03

——————— oOo ———————

Phần 1: Thực hành

1. **Trình soạn thảo Emacs**

   Trong Linux, có một số trình soạn thảo văn bản thường dùng như: nano, vi, gedit... Tuy nhiên, Emacs (*Editor MACroS*) là trình soạn thảo phổ biến nhất. Emacs chạy trên cả giao diện văn bản lẫn đồ họa.

   Các tính năng của Emacs:

   - Soạn thảo trên nhiều cửa sổ (window) và bộ đệm (buffer)
   - Tìm kiếm, thay thế, tự sửa lỗi
   - Soạn thảo đệ quy (recursive edit): cho phép soạn thảo khi một câu lệnh đang thực hiện giữa chừng
   - Nhiều chế độ soạn thảo: văn bản thường, các tệp tin chương trình (tô màu cú pháp và thực hiện từng đoạn mã lệnh), ngôn ngữ đánh dấu (HTML), LaTeX, vẽ hình bằng các kí tự
   - Các macro bàn phím
   - Sửa đổi theo ý thích cá nhân bằng cách chỉnh sửa các biến của chương trình
   - Lập trình bằng ngôn ngữ Emacs Lisp
   - Nhiều chương trình phụ trợ (danh sách thư mục, đọc và soạn e-mail, trò chơi...)

   Để bắt đầu với trình soạn thảo Emacs. Có hai cách sau:

   - *emacs filename*
     Trong đó, filename là tên của tệp tin mà bạn muốn chỉnh sửa. Nếu tệp tin đó chưa có, câu lệnh này sẽ tạo tệp tin filename cho bạn.
   - Nếu tệp tin đã tồn tại, kích chuột phải vào tệp tin và chọn Open with -> GNU Emacs

   Màn hình

   Màn hình Emacs gồm: Vùng soạn thảo, dòng chế độ, tiểu bộ đệm.

   - Dòng chế độ hiển thị các thông tin:
     - Trạng thái bộ đệm: Đã thay đổi (cặp dấu *), chưa thay đổi (cặp dấu -), hay chỉ đọc (cặp dấu %)
     - Tên tập tin đang biên tập
     - Chế độ chính (trong dấu ngoặc)
     - Lượng của tập tin xem trên màn hình
   - Tiểu bộ đệm: Nằm bên dưới dòng chế độ, dùng để hiển thị thông điệp và dấu nhắc yêu cầu gõ lệnh

Một số phím tắt

Ngoài các menu trên giao diện, Emacs còn có rất nhiều những phím tắt. Một số phím tắt cơ bản sau (trong đó, C là Ctrl, M là Alt):

- C-x/C-c: Thoát khỏi Emacs
- C-x/C-f: Tìm và mở tệp tin
- C-x/C-s: Lưu tệp tin
- C-x/C-w: Lưu tệp tin với tên khác
- C-s: Tìm kiếm
- C-x h: Đánh dấu chọn tất cả (Select all)
- C-x u: Khôi phục lại phần vừa xóa (Undo)
- C-w: Cắt (Cut) vùng được đánh dấu
- M-w: Copy vùng được đánh dấu
- C-y: Dán (Paste)
- M-x lệnh: Thực hiện lệnh (lệnh được gõ vào cửa sổ nhỏ ở phía dưới)
  Ví dụ 1: *M-x replace-string* là lệnh thay thế một xâu trong văn bản bằng xâu nhập vào.
  Ví dụ 2: *M-x replace-regexp*: Thay thế một xâu xác định bởi một biểu thức chính quy bằng một xâu khác.

Các chế độ chính trong emacs

Emacs tùy biến lệnh cho các loại văn bản khác nhau thông qua chế độ. Để thiết lập chế độ cho bộ đệm hiện tại, dùng *M-x majormodename-mode*

Ví dụ: Để vào chế độ Java Mode - biên tập mã nguồn Java, gõ *M-x java-mode*

Các chế độ trong emacs:

- *fundamental mode*: Chế độ văn bản, thích hợp cho soạn thảo mọi loại văn bản, chỉ không cung cấp các tính năng đặc biệt.
- *text mode*: Soạn thảo văn bản, có các lệnh đặc biệt để kiểm tra chính tả, canh giữa dòng, ..
- *lisp mode*: Để biên tập mã nguồn Common Lisp
- *c mode*: Để biên tập mã nguồn C, có canh lề đặc biệt, ..
- Ngoài ra còn các chế độ: TCL mode, HTML mode, TeX mode, ...

Tập tin, bộ đệm và cửa sổ

Emacs có ba cấu trúc dữ liệu liên quan mật thiết với nhau:

(a) Tập tin là tập tin Linux thực sự trên đĩa. Bạn không bao giờ biên tập trực tiếp trên tập tin này. *Emacs* đọc tập tin vào bộ đệm và viết một bộ đệm vào tập tin để lưu nó

(b) Bộ đệm là cấu trúc dữ liệu nội tại giữ văn bản bạn thực sự biên tập. Emacs có thể có con số bộ đệm bất kì tại bất kì thời điểm nào. Bộ đệm có tên; một bộ đệm xuất phát từ một tập tin gần như luôn luôn mang tên của tập tin đó, và chúng ta nói rằng bộ đệm đang viếng thăm tập tin. Điều này có nghĩa là khi bạn lưu bộ đệm, nó sẽ được lưu vào tập tin đúng. Vào

bất cứ lúc nào cũng chỉ có đúng một bộ đệm được chọn: đây là bộ đệm mà con trỏ phần cứng đang hoạt động và là nơi mà lệnh sẽ có tác dụng. Bộ đệm có thể được xoá theo ý muốn

(c) Cửa sổ là nơi bạn xem bộ đệm. Vì giới hạn thực thể của màn hình, có thể bạn không có chỗ để xem tất cả các bộ đệm cùng lúc. Bạn có thể chia nhỏ màn hình, ngang hay dọc, thành nhiều cửa sổ, mỗi cửa sổ xem một bộ đệm khác nhau. Cũng có thể có vài cửa sổ để xem các phần khác nhau của cùng bộ đệm. Có thể tạo và xoá cửa sổ tuỳ thích; xoá cửa sổ không xoá bộ đệm liên hệ với cửa sổ đó.

Lệnh thao tác trên tập tin

- C-x C-f (find-file): Nhắc nhập tên tập tin và đọc tập tin vào bộ đệm để biên tập. Nếu tập tin đang được biên tập trong một bộ đệm nào đó, nó chỉ chuyển sang bộ đệm đó mà không đọc tập tin
- C-x C-s (save-buffer): Lưu tập tin, hay chính xác hơn là viết bộ đệm hiện tại lên đĩa
- C-x s (save-some-buffers): Lưu tất cả các bộ đệm đang viếng thăm tập tin, truy vấn từng cái và đưa ra một số tuỳ chọn

Lệnh thao tác trên bộ đệm

- C-x b (switch-to-buffer): Nhắc nhập tên bộ đệm và chuyển bộ đệm của cửa sổ hiện tại sang bộ đệm đó. Tạo bộ đệm rỗng mới nếu tên mới được nhập vào.
- C-x C-b (list-buffers): Xuất hiện cửa sổ mới liệt kê tất cả bộ đệm với tên, đã thay đổi hay không, kích thước theo byte, chế độ chính và tập tin mà bộ đệm đang viếng thăm.
- C-x k (kill-buffer): Nhắc nhập tên bộ đệm và dỡ bỏ toàn bộ cấu trúc dữ liệu cho bộ đệm đó khỏi Emacs. Nếu bộ đệm đã thay đổi bạn sẽ có cơ hội lưu nó. Lệnh này không xoá tập tin liên hệ, nếu có.
- C-x C-q (vc-toggle-read-only): Đặt thuộc tính chỉ-đọc hoặc đặt thuộc tính đọc-viết nếu nó đang là chỉ-đọc.

Lệnh thao tác trên cửa sổ

- C-v (scroll-up): Cuộn tới (về cuối tập tin) một màn hình. Theo mặc định, Emacs chừa 2 hàng từ màn hình trước.
- M-v (scroll-down): Như C-v, nhưng cuộn ngược.
- C-x o (other-window): Chuyển sang cửa sổ khác. Lặp đi lặp lại lệnh này sẽ di chuyển qua tất cả các cửa sổ, từ trái sang phải và trên xuống dưới.
- C-x 1 (delete-other-windows): Xoá tất cả các cửa sổ khác, trừ cửa sổ hiện tại. Lệnh này không xoá các bộ đệm và tập tin liên hệ với cửa sổ.
- C-x 0 (delete-window): Xoá cửa sổ hiện tại, thay đổi kích thước các cửa sổ khác cho thích hợp.
- C-x 2 (split-window-vertically): Chia cửa sổ hiện tại thành hai, theo chiều dọc. Lệnh này tạo một cửa sổ mới, nhưng không tạo bộ đệm mới: cùng một bộ đệm sẽ được xem trong 2 cửa sổ. Điều này giúp xem các phần khác nhau của bộ đệm đồng thời.

- C-x 3 (split-window-horizontally): Tương tự như C-x 2 nhưng chia cửa sổ theo chiều ngang.
- C-M-v (scroll-other-window): Tương tự như C-v, nhưng cuộn các cửa sổ kế tiếp (là cửa sổ mà lệnh C-o sẽ chuyển sang).

2. **Trình soạn thảo vi**

vi là trình soạn thảo các tệp tin văn bản trong các hệ thống Linux. Có các ưu điểm sau:

- Màn hình được xem như một cửa sổ mở trên tập tin
- Có khả năng di chuyển con trỏ đến bất kì vị trí nào trên màn hình
- Cửa sổ có thể di chuyển tự do trên tập tin

Phần lớn các phím sử dụng độc lập hoặc kết hợp với các phím *Shift* và *Ctrl* để tạo ra các lệnh của **vi**. Các lệnh của **vi** được gọi khi có dấu **:** ở cuối màn hình.

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*

Kết quả là một màn hình soạn thảo sẽ hiện lên. Các dấu ~ trước mỗi dòng cho biết dòng đó còn trống. Dòng dưới cùng cho biết tên tệp tin đang được mở. Nếu như là tệp tin mới thì trạng thái của tệp tin là "[new file]". Nếu tệp tin cũ thì sẽ hiển thị số dòng, số ký tự trong tệp tin.

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:

- Ctrl + r - redo

- u - undo

3. **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: *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, 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 dưới quản lí của tài khoản *student, root*?

? Lưu kết quả của lệnh top vào tệp tin *topOutput.txt*?

? Mở một tệp tin bằng emacs, kiểm tra xem tiến trình này có hoạt động hay không?

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*) 2
  - *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
  - -p: lệnh kill sẽ chỉ đưa ra chỉ số của quá trình mà không gửi một tín hiệu nào.
  - -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.

? Dừng một tiến trình bất kì của student trong số các tiến trình vừa liệt kê, kiểm tra lại xem tiến trình đó đã dừng hay chưa?

? Dừng tiến trình emacs vừa tạo ra ở trên, kiểm tra lại xem tiến trình này còn tồn tại hay không?

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*

Ví dụ: *student@linux $renice +1 987 -u daemon root -p 32* → Lệnh trên sẽ thay đổi mức độ ưu tiên của quá trình có chỉ số là 987 và 32, và tất cả các quá trình do người dùng daemon và root sở hữu.

Chú ý: người dùng bình thường không thể thay đổi độ ưu tiên nhỏ hơn 0.

? Mở hai cửa sổ terminal, thực thi cùng lúc lệnh ls -lR với độ ưu tiên lần lượt là -19 và +19, kiểm tra xem lệnh nào thực thi xong trước?

Phần 2: Bài tập thực hành

Chụp ảnh màn hình từng phần trong bài tập tuần 03 và nộp lại trên google Classroom.

Phần 3: Liên lạc

STT	Họ và tên	Email	ĐT
1	Hà Mỹ Linh	halinh.hus@gmail.com	