



Unix Editors

Unix Editors

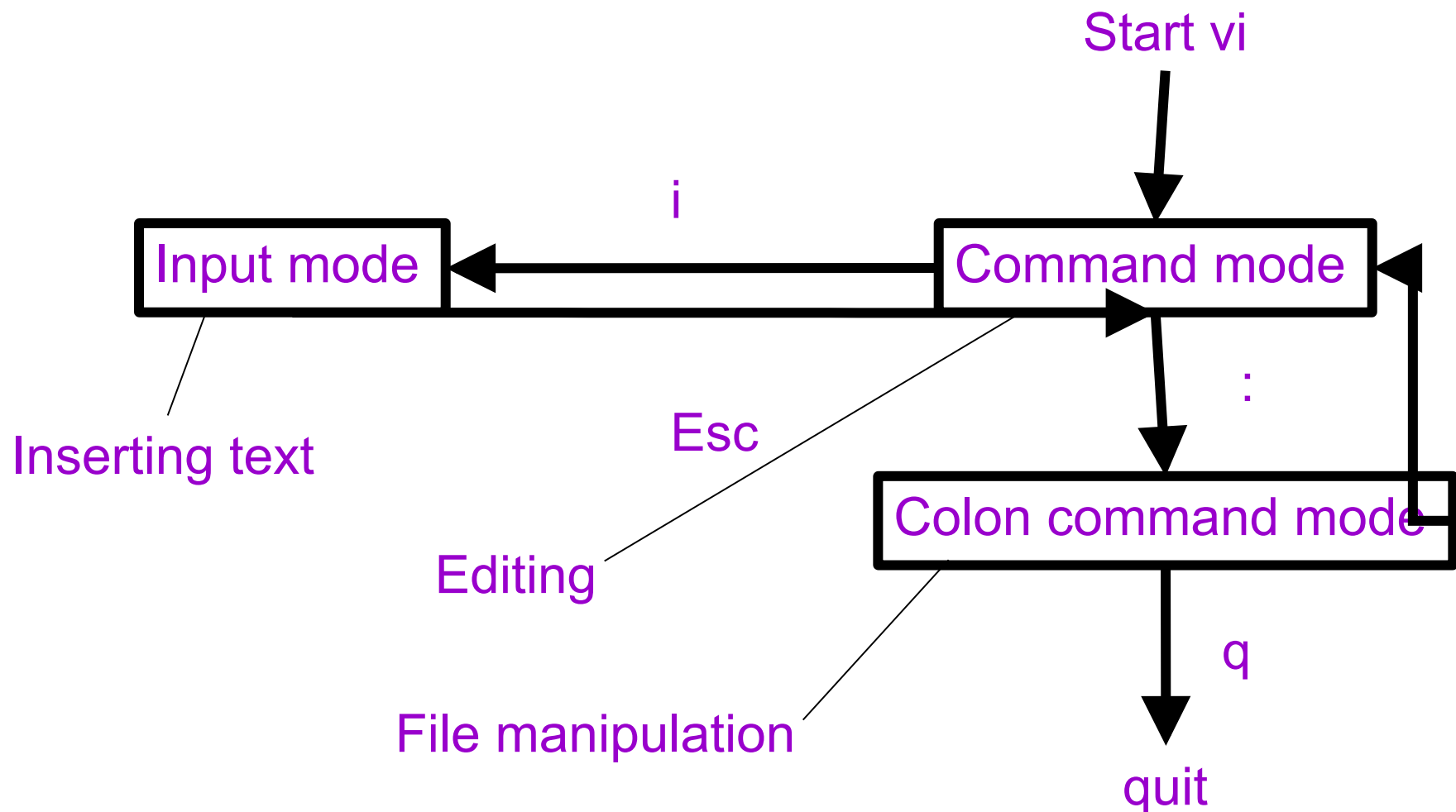
- ◆ Editors in Unix come in two general flavours:
 - **modal** editors have "modes"
 - ✦ generally input mode and command mode
 - input mode allows entry of text
 - command mode allows positioning within the file and more sophisticated text modification
 - ✦ primary Unix examples: **ed** and **vi**
 - **modeless** editors have only one mode
 - ✦ positioning and text manipulation are done by special key sequences (like arrow keys and function keys)
 - could also be done by mouse actions or menus
 - ✦ primary Unix examples: **emacs** or **pico**

ed

- ◆ ed is the original line editor
- ◆ Still one of the most powerful editors available
 - Isn't a screen editor so people dislike it
 - ✦ It doesn't give you a “local” or screen picture of what is in your file
 - Ability to make massive changes with one command
- ◆ We will meet its power base later:
regular expressions
- ◆ Many of its capabilities have been incorporated into newer editors like vi and emacs
- ◆ Still used in some shell scripts.

vi or vim (1)

- ◆ Developed by UCB and comes with all versions of Unix.
- ◆ Difficult to use initially, but very fast for experienced users.
- ◆ Has three modes:



vi (2)

◆ Starting vi

`compute > vi filename`

✧ `filename` is either a new or existing filename

◆ The following happen:

- If the filename you typed was an existing file, you will see the first page of the file on your screen.
- If you typed a new filename, you will be faced with a blank screen, and you may type the file.
- The file name will appear at the bottom left of the screen.
- Line and column numbers of the cursor may also appear at the bottom.

vi (3)

◆ Several basic commands

- **arrow keys** move the cursor
- **i** change to input mode
- **esc** go to command mode
- **x** delete the current character
3x delete 3 characters
- **dd** delete the current line
3dd delete three lines
- **u** undo the last change
- **/** search for the text following **/**
- **:w** write to file (**:** colon command mode)
- **:wq** save and quit
- **:q** quit (if no change after the last saving)
- **:q!** exit without save

vi (4)

Cursor Movement

- **h** move one char left
- **j** move one line down
- **k** move one line up
- **l** move one char right

- **w** move to next word
- **e** move to end of current word
- **b** move to beginning of previous word
- **0** move to the beginning of the current line
- **\$** move to the end of the current line

Screen Movement

- **H** move to top of screen
- **L** move to bottom of screen
- **^F** scroll down one page
- **^B** scroll up one page
- **^U** scroll up one half page
- **^D** scroll down one half page

Search

- **/** search for something
n for next occurrence
- **?** search backwards for something

Adding Text

- **o** (**O**) opens new line below (above) the current line
- **i** (**I**) inserts text before current char (beginning of line)
- **a** (**A**) appends text after current char (end of line)

vi (5)

Deletion Commands

- ♦ **x** delete character under cursor
- ♦ **D** delete to end of line
- ♦ **dd** delete entire line
- ♦ **d\$** delete to end of line
- ♦ **d0** delete to beginning of line
- ♦ **dw** delete the current word
- ♦ **db** delete the previous word

Other Commands

- ♦ **.** redo last modification command
- ♦ **u** undo last modification command
- ♦ **yy** or **Y** copy current line
- ♦ **p** paste
- ♦ **:w file** write the buffer to this file
- ♦ **:r file** read this file into the buffer

Change Commands

- ♦ **s** substitute a string for current char (end with ESC)
- ♦ **r** replace current char with another
- ♦ **R** overwrite text (end with ESC)
- ♦ **C** replace to end of current line (end with ESC)
- ♦ **c\$** replace to end of current line (end with ESC)
- ♦ **c0** replace to beginning of current line (end with ESC)
- ♦ **cw** replace the current word (end with ESC)
- ♦ **cb** replace the previous word (end with ESC)

Spell checking

- ♦ **:set spell** spell checking
- ♦ **]s** next misspelled word
- ♦ **z=** possible suggestions

vi (6)

◆ Basic search and replace

- `:%s/foo/bar/g` find each occurrence (in all lines) of 'foo' and replace it with 'bar'
- `:%s/foo/bar/gc` change each 'foo' and to 'bar', but ask for confirmation first
- `:s/foo/bar/g` find each occurrence (in the current line) of 'foo' and replace it with 'bar'
- `:s/foo/bar/` find first occurrence (in the current line) of 'foo' and replace it with 'bar'
- `:5,9s/foo/bar/g` find each occurrence (from line 5 to 9) of 'foo' and replace it with 'bar'
- search pattern can use **regular expression**

◆ Output format: could output postscript and html

◆ Get help with command **vimtutor**

emacs (1)

◆ Emacs: (Editor MACroS)

- developed by Richard Stallman, founder of GNU project, and James Gosling, father of Java, amongst many others
- modeless
- has versions for Unix, Windows, and other systems
- menu-driven and mouse-driven under X-windows
- to avoid using X-windows (for instance in a terminal)

✦ **emacs -nw**

◆ Emacs uses special keys (ESC and CTRL) to perform editor functions other than input

◆ This editor can do everything

- Contains a complete programming language (a LISP interpreter) which can be used to write functions for use in the editor

emacs (2)

◆ Key combination: a sequence of (special) keys

- C-x "Control X"
 - ✦ Hold down Control key while typing x.
- C-x C-c
 - ✦ Hold down Control key while typing x and c.
 - ✦ Or hold down control key while typing x, then release, then hold down control while typing c.
- C-x u
 - ✦ Hold down the Control key, keep it down while typing x. Release the Control key and type u.
- ESC x "Escape x" or "Meta x"
 - ✦ What always works:
 - Type the Escape key. Release. Type x.
 - ✦ What sometimes works (and is convenient):
 - Hold down the Alt key and x key at the same time

emacs (3)

◆ Starting Emacs on a file:

`compute > emacs -nw myfile`

✦ `myfile` is either a new or existing filename.

◆ The following happen:

- If the filename you typed was an existing file, you will see the first page of the file on your screen.
- If you typed a new filename, you will be faced with a blank screen, and you may type the file.
- The file name will appear at the bottom of the screen.

emacs (4)

◆ When you encounter problems ...

- Emacs is a very powerful editor
 - ✦ No matter what key combination you press, it probably does something!
 - ✦ Sometimes it does something you didn't want!
- UNDO
 - ✦ To undo last operation: **Ctrl-_** (Control & underscore)
 - ✦ You can also use: **Ctrl-x u**
 - ✦ Can be repeated to keep undoing operations
- Cancel
 - ✦ If you get to a mode which you don't want
 - e.g: you typed **Ctrl-x** and emacs expects more
 - ✦ Type **Ctrl-g**
 - It will usually back you out of almost anything

emacs (5)

Cursor Movement

- ◆ Arrow keys move the cursor around screen.
- ◆ Alternatively, use:
 - **Ctrl-f** Forward a character (Right)
 - **Ctrl-b** Back a character (Left)
 - **Ctrl-n** Next line (Down)
 - **Ctrl-p** Previous line (Up)

Other Movements:

- ◆ **Ctrl-a** Beginning of line.
- ◆ **Ctrl-e** End of line.
- ◆ **Ctrl-v** View next screen.
- ◆ **ESC v** View previous screen.
- ◆ **ESC <** Start of file.
- ◆ **ESC >** End of file.
- ◆ **ESC f** Forward a word.
- ◆ **ESC b** Back a word.
- ◆ **ESC x goto-line** Goes to a given line number.

emacs (6)

Cut and Paste

- ◆ To move a block of text
 - Move cursor to start of block
 - **Ctrl-@** Set mark
 - Move cursor to end of block
 - **Ctrl-w** Wipe out (Cut)
 - **ESC w** Copy.
 - Move cursor to new location
 - **Ctrl-y** Yank back last thing killed or copied (Paste).
 - The **Ctrl-y** may be repeated for multiple copies.

Text Deletion

- ◆ **Backspace**
 - Kill character before cursor
- ◆ **Ctrl-d**
 - Delete character at cursor
- ◆ **ESC d**
 - Delete next word.
- ◆ **Ctrl-k** Kill line
 - delete from cursor to the end of line.
- ◆ **Ctrl-x u** Undo last change.
 - Repeat to undo as many changes as you wish.
- ◆ **ESC x revert-buffer**
 - Undo all changes since last save.

emacs (7)

Save / Exit

- ◆ **Ctrl-x Ctrl-s**
 - Save file (over-write original)
- ◆ **Ctrl-x Ctrl-c**
 - Exit from emacs.
- ◆ **Ctrl-x Ctrl-w**
 - Save in different file
 - You are prompted for name

Emacs creates extra files.

- ◆ When you save using **Ctrl-x Ctrl-s**, the old file will be kept as **filename~**.
- ◆ If you exit without saving, the modified unsaved file will be saved as **#filename#**.

Other Commands

- ◆ Check spelling
 - Type **ESC \$**
Check spelling of 1 word.
 - **ESC x ispell-buffer**
Check spelling of file.
- ◆ Insert a file
 - **Ctrl-x i**
Insert a file at current cursor position.
- ◆ Reformat regions
 - **ESC q** Reformat paragraph
 - To reformat a region:
 - ❖ Move cursor to the start of the block.
 - ❖ **Ctrl-@**
 - ❖ Move cursor to the end of the block.
 - ❖ **ESC x fill-region**

emacs (8)

Searching

- ◆ Search allows you to search for a string
- ◆ Search from the cursor position to the end of file.
- ◆ To search for a string, type **Ctrl-s**
 - ✦ prompt for search string
 - ✦ **Ctrl-s** again for next occurrence
 - ✦ **Ctrl-g** to quit

Search and Replace

- ◆ Replace all occurrences of one string with another
 - **ESC x replace-string**
 - ✦ you are prompted for the text and the replacement text
- ◆ Query-replace asks before replacing each occurrence.
 - Type: **ESC %**
 - you are prompted for search & replace strings.
 - At each occurrence, respond:
 - ❖ **y/n** to replace/not replace.
 - ❖ **!** to replace all remaining
 - ❖ **ESC** to exit
 - ❖ **?** for lots more options

Pico

- ◆ **pico** is the PIne COmposer
 - the text editor used in the University of Washington's popular **pine** e-mail program
- ◆ **pico** is a modeless editor like **emacs**
 - always in “insert” mode
 - command keys available are always listed at the bottom of the screen
 - examples:
 - ✦ Ctrl-g Gets help
 - ✦ Ctrl-r Reads a file
 - ✦ Ctrl-o Writes a file
 - ✦ Ctrl-x Exits **pico**