# Introduction to Linux Part 2a: additional items and editors

Anita Orendt and Wim Cardoen
Center for High Performance Computing

### More on Environment

- Environment variable variable with a name and associated value; used in shells, scripts
- Use env or printenv command to see environment variables
- Some important variables
  - \$USER
  - \$PATH paths to search for commands
  - \$LD\_LIBRARY\_PATH paths to search for libraries when linking a program (more on that later)

### **More about Processes**

 Process := Running Linux program Each process has a PID (Process ID)

```
ps :: Report snapshot of the current processes
  ps [options]
```

```
ps x Display ALL the processes to `whoami` ps ax Display ALL processes ps aux Display ALL processes (more detailed) ps auxw Display ALL processes (more detailed & unlimited width) ps -eFwww
```

# Killing processes

- kill PID
- killall processname
- kill -9 PID

### **Other Job Controls**

- Ctrl+C (^C) terminate the currently running process
- Ctrl-Z (^Z) suspends the currently running process
- & runs the job in the background
- Jobs: lists all jobs, with their number
- bg %n: puts current or specified job (%n) in the background and
- fg %n: bring suspended program back to the foreground, e.g., so that it occupies the shell until done

# Monitoring processes/usage

- uptime
- free
- top
- atop
- htop
- sar

# Moving files to/from CHPC

- Windows there are graphical tools such as WinSCP
- Mac, Windows, cloud options cyberduck another graphical tool

- Linux
  - scp command (secure shell copy) to copy files between linux systems
  - wget to download from web with URL
  - curl

 For larger data sets – look into the Data Transfer Nodes (DTNs) and transfer tools such asglobus

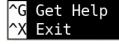
# **Editors**

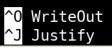
There are many choices – a few are

- Nano
- Vi
- Emacs

### Nano editor

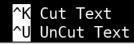
Start with command nano



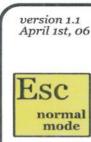




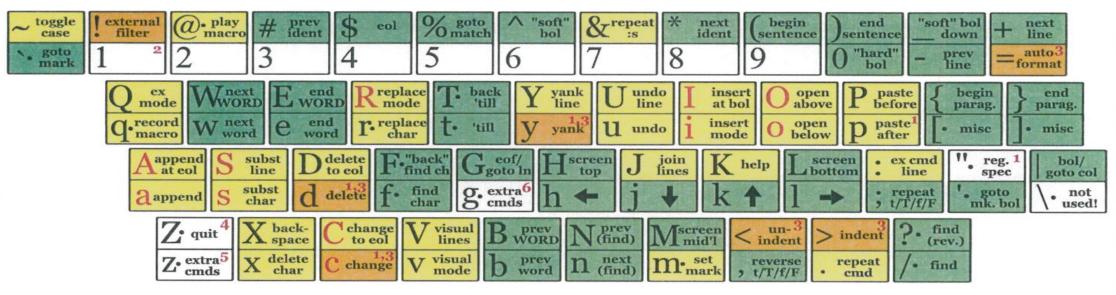








### vi / vim graphical cheat sheet



motion moves the cursor, or defines the range for an operator

command direct action command, if red, it enters insert mode requires a motion afterwards, operates between cursor & destination

extra special functions, requires extra input commands with a dot need

d a char argument afterwards bol = beginning of line, eol = end of line, mk = mark, yank = copy

words: quux(foo, bar, baz); WORDs: quux(foo, bar, baz); Main command line commands ('ex'):
:w (save), :q (quit), :q! (quit w/o saving)
:e f (open file f),
:%s/x/y/g (replace 'x' by 'y' filewide),
:h (help in vim), :new (new file in vim),

Other important comands: CTRL-R: redo (vim), CTRL-F/-B: page up/down, CTRL-E/-Y: scroll line up/down,

CTRL-V: block-visual mode (vim only)

#### Visual mode:

Move around and type operator to act on selected region (vim only)

#### **Notes:**

- (1) use "x before a yank/paste/del command to use that register ('clipboard') (x=a..z,\*) (e.g.: "ay\$ to copy rest of line to reg 'a')
- (2) type in a number before any action to repeat it that number of times (e.g.: 2p, d2w, 5i, d4j)
- (3) duplicate operator to act on current line (dd = delete line, >> = indent line)
- (4) ZZ to save & quit, ZQ to quit w/o saving
- (5) zt: scroll cursor to top, zb: bottom, zz: center
- (6) gg: top of file (vim only), gf: open file under cursor (vim only)

For a graphical vi/vim tutorial & more tips, go to www.viemu.com - home of ViEmu, vi/vim emulation for Microsoft Visual Studio

### Vi editor

☐ Editor provided with the OS ☐ There are other choices such as nano (easy) and emacs (hard) ☐ It is better to edit in a linux shell than moving file to your laptop for editing ☐ To start – command is vi (usage: vi filename) ☐ If filename exists, vi will open it up in the editor ☐ If filename does not exist, vi will create it and put you in insert mode ☐ Use arrow keys to move cursor to location ☐ Two modes – command, input ☐ in command mode — the characters typed are interpreted as commands ☐ There is also an external command mode – access with ':' ☐ In insert mode — the characters typed are added to the file ☐ The esc key exists the insert mode ☐ Many commands, we are just mentioning some basic ones to get you started

# Exiting the vi editor

- ☐ ZZ will exit and save as same filename
- $\Box : \mathbf{q}$  quit; will let you know if you have unsaved changes
- ☐:q! quit discarding changes
- $\square : \mathbf{wq}$  write and quit; same as doing : w followed by : q
  - ☐ saves as same filename
- ☐ To change name add new name after the :w or :wq
- ☐ Note if you **cntl Z** to suspend vi saves a swap filename to save current status of the edits. File is named .filename.swp. Next time you try to edit this same file it will let you know the swp file exists and asks you how to proceed. If you do not want to keep changes, delete this file

## Moving around a file

☐ down arrow or j moves down one line up arrow or k moves up one line ☐ right arrow or I moves right by one character ☐ left arrow or h moves left by one character **□** 0 or ^ moves to start of current line □ \$ moves to end of current line ☐ w moves to beginning of next word ☐ b moves to beginning of previous word □ :0 or :1 or 1G moves to start of first line **:** n or nG moves to start of nth line **\(\begin{aligned}
\begin{aligned}
\begin{alig** □ cntl-f (^f) moves forward one screen (cntl-d moves forward ½ screen) ☐ cntrl-b (^b) moves back one screen (cntrl-u moves back ½ screen)

# Inserting or Adding Text

☐ Remember – esc to exit to command mode

- $\Box$  i insert at position of cursor
- ☐ I insert at beginning of line
- ☐ a append after position of cursor
- $\Box$  A append at end of line
- $\Box$  o new line below current line
- □ O new line above current line

# Changing and Deleting Text

- $\Box$  **r** replace single character
- □ R replace characters, starting at cursor position, until esc hit

- $\Box$  x delete character that cursor is on (nx for n characters starting with one cursor is on)
- □ dd delete current line (can do n lines with ndd)
- $\Box$  **D** delete from cursor to end of line
- □ dw delete word

# **Cutting and Pasting Text**

- ☐ Y "yanks" current line into buffer (can use **nY** for n lines, current plus following lines)
- $\Box$  p paste lines in buffer after current line (P paste lines before current line)

### Other Useful Commands

- ☐ /pattern searches for next occurrence of pattern, then n goes to next occurrence; can also use / and ? To move to previous and next occurrence
- ☐?pattern searches for previous occurrence of pattern
- □ **u** to undo results of last command (use multiple times to revert back through multiple commands

### External commands

- ☐ already mentioned some of these in the moving around and exiting vi
- □ :s/old\_text/new\_text/ replaces next occurrence of old\_text in current line
- □:1,\$s/old\_text/new\_text/g replaces all occurrences of old\_text in file

### Exercise – Practice these commands!

- Try the "vimtutor" command
- Try writing some source code (C, python, fortran, etc)
- Try writing your grocery list