

CS35L – WINTER 2018

| | |
|---------------|--------------|
| Slide set: | 1.2 |
| Slide topics: | Linux basics |
| Assignment: | 1 |

Assignment Graders

This lists the TA(s) grading each assignment. Each of the TA(s) will be responsible for answering the Piazza questions belonging to his/her/their assignment since he/she/they will be grading it. You should post your questions on Piazza first but if you feel like your question is not being answered on Piazza, then you should email only the TA(s) responsible for that assignment. You can find each TA's email address and office hours on the CCLE home page for CS 35L.

Assignment 1: Zhaowei Tan and Justin Wood

Assignment 2: Justin Wood

Assignment 3: Mevlut Turker Garip

Assignment 4: Alan Litteneker

Assignment 5: Rahul Dhavalikar

Assignment 6: Guangyu Zhou

Assignment 7: Mevlut Turker Garip and Rahul Dhavalikar

Assignment 8: Zhaowei Tan

Assignment 9: Alan Litteneker and Guangyu Zhou

Assignment 10: Each TA will grade his/her own section's submissions.

THINGS TO KNOW

- Assignments: 10 times
 - Lab and Homework: submitted on CCLE every Saturday (except last two)
 - All assignments should be done individually!!!
- Grading
 - Assignments – 50% (equally weighted)
 - Final Exam – 50% (open book, open notes)

THINGS TO KNOW

- Policy for late submission
 - $2^N \%$ of the assignments value for being N days late
 - **Assignment 10** should be submitted on time
 - No submission will be accepted after the last day of instruction
- Attendance
 - No mandatory except the presentation, but highly encouraged
 - You are not required to finish the lab here in class
 - You can do everything using your computer by access to Seasnet server remotely.

THINGS TO KNOW

- For some of the later labs you will need a [Seeed Studio BeagleBone Green Wireless Development Board](#). You may wish to get the higher-priced [Seeed Studio BeagleBone Green Wireless IOT Kit](#), as this is a superset of the basic unit needed for 35L, and is used by CS 111 this quarter (and likely in later quarters, though this is not guaranteed).
- These units are available from Seeed, Amazon, Digi-Key, Mouser Electronics, Verical, and other sources.

THINGS TO KNOW

- Get a Seasnet account ASAP!
- Login and do your Homework on the following server
 - `ssh [username]@lnxsrv06.seas.ucla.edu`
 - `ssh [username]@lnxsrv07.seas.ucla.edu`
 - `ssh [username]@lnxsrv09.seas.ucla.edu`
- We are going to test your assignment solutions on these servers

Course Overview

- Week 1 – Introduction to Linux
- Part I: Basic tools and languages
 - Week 2 – Shell Scripting and Regular Expression
 - Week 3 – Modify and Rewrite Software (makefile, Python)
 - Week 7 – Secure Shell
 - Week 9 – Change Management (Git)
- Part II: C programming, basic concepts for Operating System
 - Week 4 – C programming and Debugging
 - Week 5 – System Call
 - Week 6 – Multiple Thread Programming
 - Week 8 – Dynamic Linking

Lab1: hints

- For lab questions(ans1.txt)
 - Answer 15 questions using natural language
 - For each question, list all the commands used to solve it
 - Give some explanations about your choice of commands
 - Will be graded manually

Assignment 1 is available

- Check:
- Deadline:

11:55 PM on April 7th, 2018

The Basics

- First: learn to use man command!
- Supplement materials: Linux command cheat sheet



<http://www.rain.org/~mkummel/unix.html>

- Online document:

<http://www.linuxdevcenter.com/cmd/>



FOLLOW UP – LAST LAB



- Basic commands
 - Structure
 - Permissions
- 
- 

SPECIAL PERMISSIONS

- sticky bit (o+t)
 - On shared directories, it locks files within the directory from being modified/deleted by users other than the file creator, owner of the directory, or root, even if others have write permissions (Example: /tmp)
 - Using +t or 1
- setuid, setgid (u+s, g+s)
 - “set user ID upon execution”
 - Run an executable with the permissions/privileges of the executable’s owner or group
 - +s or 4,2
 - For example, the setuid permission on the passwd command makes it possible for a normal user to change passwords by updating few system files like /etc/passwd and /etc/shadow which can’t be updated by non-root accounts.


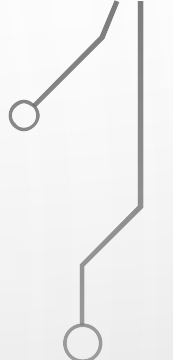



PROCESS: PS AND KILL

- Process
 - An instance of a computer program in execution
 - ps
 - List processes that are currently running
 - kill
 - Terminate a certain process
 - Usage
 - kill PID
- 
- 



DAEMON


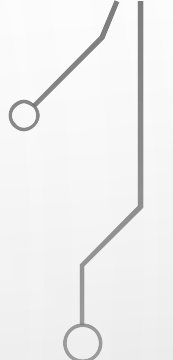

- A process that runs in the background
 - Example: cron
 - Enables users to schedule jobs to run periodically at certain times (cron jobs)
 - Usage: Full Backup every month
- 
- 
- 

DIFF

- A file comparison utility that outputs the differences between two files.
- Shows the changes between one version of a file and a former version of the same file
- Usage
 - `diff original_file new_file`
 - `diff -u original_file new_file`



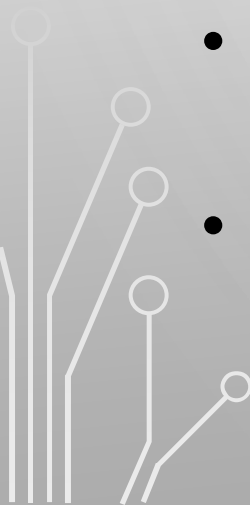
WGET

- A computer program that retrieves content from web servers
 - Usage
 - `wget <URL>`
- 
- 
- 



EMACS

“The customizable, extensible, self documenting, real-time display editor”

- Customizable (no programming)
 - Users can customize font, colors, etc. in ~/.emacs
 - Extensible (programming required)
 - Run Lisp scripts to define new commands (dired)
 - Self-documenting
 - C-h r (manual) and C-h t (tutorial)
 - Real-time
 - Edits are displayed onscreen as they occur
- 

GETTING STARTED

- Install emacs
 - Should be installed already
- Emacs has both GUI and CLI
- All emacs commands start with “C” or “M”
 - “C” = ctrl; “M” = alt (Windows) / option(Mac)
- Starting emacs
 - `emacs <filename>`
- Exiting emacs
 - C-x C-c
- Saving a file
 - C-x C-s

LEARNING TO USE EMACS



- Navigating with file
 - Move up/down/left/right: C-p, C-n, C-b, C-f (arrow keys also work)
 - Move to the beginning/end of a line: C-a, C-e
 - Move to the first/last line of the text: M- < M->
 - (use shift for < and >)
 - Move to particular line number: M-g g [number]
- Search and replace file
 - C-s: search forward
 - C-r: search backward
 - M-%: replace (usage: M-% [source] Enter [dest]) (y/n)
- Erasing a line
 - C-k: erase from current cursor to end of line

LEARNING TO USE EMACS

- Copy and paste in a file
 - Begin: C-@ (press Ctrl+Shift+2)
 - Use the <up> and <down> buttons to select the contents
 - End: C-w (cut), M-w(copy), C-y (paste)
 - Undo command: C-x u
- Emacs: Cutting is called **killing**, and pasting is called **yanking**



DIRECTORY EDIT (DIRED)

- C-x+d
 - Creates an Emacs buffer containing list of dir
 - Allows you to operate on files
 - remove, rename, encrypt, decrypt, edit
 - Allows you to navigate filesystem
 - Switch to different directories and list content
- 
- 

BUFFERS



- Visiting Emacs scratch buffer:
 - Copy current buffer to file
C-x C-s
 - List all the buffers
C-x C-b
 - Save current buffer with a specified file name
C-x C-w [filename]
 - Add a new file to buffer
C-x C-f
 - Visit *scratch* buffer
C-x b

OTHER EMACS TRICKS...

- Emacs as shell
 - M-! <command>, **M-x shell** (interactive shell)
- Emacs as IDE
 - **M-x compile**, then specify command to compile
 - Tip for homework: gcc hello.c -o hello
 - Run the executable by running the shell command
 - ./hello
- Running Lisp code
 - **M-x emacs-lisp-mode**
 - C-x C-e : Evaluate expression up to point



COLUMN NUMBER CHECK

- Emacs editor
 - M-x column-number-mode or
 - Options → Show/Hide → Check “Column Numbers”
 - The number inside the box is the Column Number.
- 
- 

EMACS – RECORD***

- To record:

- Ctrl+x+(
- <Type>
- Ctrl+x+)

- To run:

- Ctrl+x+e



ORG

- https://orgmode.org/worg/org-tutorials/orgtutorial_dto.html
- 
- 

Starting Emacs

To enter GNU Emacs 20, just type its name: **emacs**
To read in a file to edit, see Files, below.

Leaving Emacs

suspend Emacs (or iconify it under X) **C-z**
exit Emacs permanently **C-x C-c**

Files

read a file into Emacs **C-x C-f**
save a file back to disk **C-x C-s**
save all files **C-x s**
insert contents of another file into this buffer **C-x i**
replace this file with the file you really want **C-x C-v**
write buffer to a specified file **C-x C-w**
version control checkin/checkout **C-x C-q**

Getting Help

The help system is simple. Type **C-h** (or **F1**) and follow the directions. If you are a first-time user, type **C-h t** for a tutorial.
remove help window **C-x 1**
scroll help window **C-M-v**
apropos: show commands matching a string **C-h a**
show the function a key runs **C-h c**
describe a function **C-h f**
get mode-specific information **C-h m**

Error Recovery

abort partially typed or executing command **C-g**
recover a file lost by a system crash **M-x recover-file**
undo an unwanted change **C-x u** or **C-_**
restore a buffer to its original contents **M-x revert-buffer**
redraw garbaged screen **C-l**

Incremental Search

search forward **C-s**
search backward **C-r**
regular expression search **C-M-s**
reverse regular expression search **C-M-r**
select previous search string **M-p**
select next later search string **M-n**
exit incremental search **RET**
undo effect of last character **DEL**
abort current search **C-g**
Use **C-s** or **C-r** again to repeat the search in either direction. If Emacs is still searching, **C-g** cancels only the part not done.

Motion

| | | |
|---|---------------|-----------------|
| entity to move over | backward | forward |
| character | C-b | C-f |
| word | M-b | M-f |
| line | C-p | C-n |
| go to line beginning (or end) | C-a | C-e |
| sentence | M-a | M-e |
| paragraph | M-{ | M-} |
| page | C-x [| C-x] |
| sexp | C-M-b | C-M-f |
| function | C-M-a | C-M-e |
| go to buffer beginning (or end) | M-< | M-> |
| scroll to next screen | | C-v |
| scroll to previous screen | | M-v |
| scroll left | | C-x < |
| scroll right | | C-x > |
| scroll current line to center of screen | C-u | C-l |

Killing and Deleting

| | | |
|---|------------------|-----------------|
| entity to kill | backward | forward |
| character (delete, not kill) | DEL | C-d |
| word | M-DEL | M-d |
| line (to end of) | M-0 C-k | C-k |
| sentence | C-x DEL | M-k |
| sexp | M-- C-M-k | C-M-k |
| kill region | | C-w |
| copy region to kill ring | | M-w |
| kill through next occurrence of <i>char</i> | | M-z char |
| yank back last thing killed | | C-y |
| replace last yank with previous kill | | M-y |

Marking

| | |
|--------------------------------|----------------------------|
| set mark here | C-@ or C-SPC |
| exchange point and mark | C-x C-x |
| set mark <i>arg</i> words away | M-0 |
| mark paragraph | M-h |
| mark page | C-x C-p |
| mark sexp | C-M-@ |
| mark function | C-M-h |
| mark entire buffer | C-x h |

Query Replace

| | |
|--|---------------------------------|
| interactively replace a text string | M-% |
| using regular expressions | M-x query-replace-regexp |
| Valid responses in query-replace mode are | |
| replace this one, go on to next | SPC |
| replace this one, don't move | , |
| skip to next without replacing | DEL |
| replace all remaining matches | ! |
| back up to the previous match | ^ |
| exit query-replace | RET |
| enter recursive edit (C-M-c to exit) | C-r |

Multiple Windows

When two commands are shown, the second is for "other frame."
delete all other windows **C-x 1**
split window, above and below **C-x 2** **C-x 5 2**
delete this window **C-x 0** **C-x 5 0**
split window, side by side **C-x 3**
scroll other window **C-M-v**
switch cursor to another window **C-x o** **C-x 5 o**
select buffer in other window **C-x 4 b** **C-x 5 b**
display buffer in other window **C-x 4 C-o** **C-x 5 C-o**
find file in other window **C-x 4 f** **C-x 5 f**
find file read-only in other window **C-x 4 r** **C-x 5 r**
run Dired in other window **C-x 4 d** **C-x 5 d**
find tag in other window **C-x 4 .** **C-x 5 .**
grow window taller **C-x ^**
shrink window narrower **C-x {**
grow window wider **C-x }**

Formatting

| | |
|--|----------------|
| indent current line (mode-dependent) | TAB |
| indent region (mode-dependent) | C-M-\ |
| indent sexp (mode-dependent) | C-M-q |
| indent region rigidly <i>arg</i> columns | C-x TAB |
| insert newline after point | C-o |
| move rest of line vertically down | C-M-o |
| delete blank lines around point | C-x C-o |
| join line with previous (with <i>arg</i> , next) | M-^ |
| delete all white space around point | M-\ |
| put exactly one space at point | M-SPC |
| fill paragraph | M-q |
| set fill column | C-x f |
| set prefix each line starts with | C-x . |
| set face | M-g |

Case Change

| | |
|------------------|----------------|
| uppercase word | M-u |
| lowercase word | M-l |
| capitalize word | M-c |
| uppercase region | C-x C-u |
| lowercase region | C-x C-l |

The Minibuffer

The following keys are defined in the minibuffer.
complete as much as possible **TAB**
complete up to one word **SPC**
complete and execute **RET**
show possible completions **?**
fetch previous minibuffer input **M-p**
fetch later minibuffer input or default **M-n**
regexp search backward through history **M-r**
regexp search forward through history **M-s**
abort command **C-g**

Type **C-x ESC ESC** to edit and repeat the last command that used the minibuffer. Type **F10** to activate the menu bar using the minibuffer.

GNU Emacs Reference Card

Buffers

| | |
|-----------------------|---------|
| select another buffer | C-x b |
| list all buffers | C-x C-b |
| kill a buffer | C-x k |

Transposing

| | |
|----------------------|---------|
| transpose characters | C-t |
| transpose words | M-t |
| transpose lines | C-x C-t |
| transpose sexps | C-M-t |

Spelling Check

| | |
|---------------------------------------|-------------------|
| check spelling of current word | M-\$ |
| check spelling of all words in region | M-x ispell-region |
| check spelling of entire buffer | M-x ispell-buffer |

Tags

| | |
|--|------------------------|
| find a tag (a definition) | M-. |
| find next occurrence of tag | C-u M-. |
| specify a new tags file | M-x visit-tags-table |
| regex search on all files in tags table | M-x tags-search |
| run query-replace on all the files | M-x tags-query-replace |
| continue last tags search or query-replace | M-, |

Shells

| | |
|---------------------------------------|-----------|
| execute a shell command | M-! |
| run a shell command on the region | M- |
| filter region through a shell command | C-u M- |
| start a shell in window *shell* | M-x shell |

Rectangles

| | |
|-------------------------------------|---------|
| copy rectangle to register | C-x r r |
| kill rectangle | C-x r k |
| yank rectangle | C-x r y |
| open rectangle, shifting text right | C-x r o |
| blank out rectangle | C-x r c |
| prefix each line with a string | C-x r t |

Abbrevs

| | |
|--|-----------|
| add global abbrev | C-x a g |
| add mode-local abbrev | C-x a l |
| add global expansion for this abbrev | C-x a i g |
| add mode-local expansion for this abbrev | C-x a i l |
| explicitly expand abbrev | C-x a e |
| expand previous word dynamically | M-/ |

Regular Expressions

| | | |
|--|---------|--------------------------|
| any single character except a newline | . | (dot) |
| zero or more repeats | * | |
| one or more repeats | + | |
| zero or one repeat | ? | |
| quote regular expression special character c | \c | |
| alternative ("or") | | |
| grouping | (...) | |
| same text as nth group | \n | |
| at word break | \b | |
| not at word break | \B | |
| entity | | match start match end |
| line | ^ | \$ |
| word | \< | \> |
| buffer | \' | \' |
| class of characters | | match these match others |
| explicit set | [...] | [^ ...] |
| word-syntax character | \w | \W |
| character with syntax c | \sc | \Sc |

International Character Sets

| | |
|------------------------------------|------------------------------|
| specify principal language | M-x set-language-environment |
| show all input methods | M-x list-input-methods |
| enable or disable input method | C-\ |
| set coding system for next command | C-x RET c |
| show all coding systems | M-x list-coding-systems |
| choose preferred coding system | M-x prefer-coding-system |

Info

| | |
|---|---------|
| enter the Info documentation reader | C-h i |
| find specified function or variable in Info | C-h C-i |

Moving within a node:

| | |
|-------------------|---------|
| scroll forward | SPC |
| scroll reverse | DEL |
| beginning of node | . (dot) |

Moving between nodes:

| | |
|--|---|
| next node | n |
| previous node | p |
| move up | u |
| select menu item by name | m |
| select nth menu item by number (1-9) | n |
| follow cross reference (return with 1) | f |
| return to last node you saw | l |
| return to directory node | d |
| go to any node by name | g |

Other:

| | |
|-------------------------|-----|
| run Info tutorial | h |
| quit Info | q |
| search nodes for regexp | M-s |

Registers

| | |
|--------------------------------------|-----------|
| save region in register | C-x r s |
| insert register contents into buffer | C-x r i |
| save value of point in register | C-x r SPC |
| jump to point saved in register | C-x r j |

Keyboard Macros

| | |
|-------------------------------------|-------------------------|
| start defining a keyboard macro | C-x (|
| end keyboard macro definition | C-x) |
| execute last-defined keyboard macro | C-x e |
| append to last keyboard macro | C-u C-x (|
| name last keyboard macro | M-x name-last-kbd-macro |
| insert Lisp definition in buffer | M-x insert-kbd-macro |

Commands Dealing with Emacs Lisp

| | |
|-------------------------------------|------------------|
| eval sexp before point | C-x C-e |
| eval current defun | C-M-x |
| eval region | M-x eval-region |
| read and eval minibuffer | M-: |
| load from standard system directory | M-x load-library |

Simple Customization

| | |
|-------------------------------|---------------|
| customize variables and faces | M-x customize |
|-------------------------------|---------------|

Making global key bindings in Emacs Lisp (examples):

```
(global-set-key "\C-cg" 'goto-line)
(global-set-key "\M-#" 'query-replace-regexp)
```

Writing Commands

```
(defun command-name (args)
  "documentation" (interactive "template")
  body)
```

An example:

```
(defun this-line-to-top-of-window (line)
  "Reposition line point is on to top of window.
With ARG, put point on line ARG."
  (interactive "P")
  (recenter (if (null line)
                0
                (prefix-numeric-value line))))
```

The `interactive` spec says how to read arguments interactively. Type `C-h f interactive` for more details.

Copyright © 1997 Free Software Foundation, Inc.
v2.2 for GNU Emacs version 20, June 1997
designed by Stephen Gildea

Permission is granted to make and distribute copies of this card provided the copyright notice and this permission notice are preserved on all copies.

For copies of the GNU Emacs manual, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA