CS 35L

Linux Basics

Slide Set 1.1

Lab 7 – Spring 2020

What's this class about?

"Fundamentals of commonly used **software tools** and **environments**, particularly **open-source** tools to be used in upper division computer science courses"

Course Information

- TA: **Joe Halabi**
  - joe.halabi@gmail.com
  - Office Hours TBD (will be posted on CCLE once finalized)

- Syllabus & Course Website
  - https://web.cs.ucla.edu/classes/spring20/cs35L/
- Questionsly for class discussions
  - Link found on CCLE
- No textbook; online material will be referred to extensively.

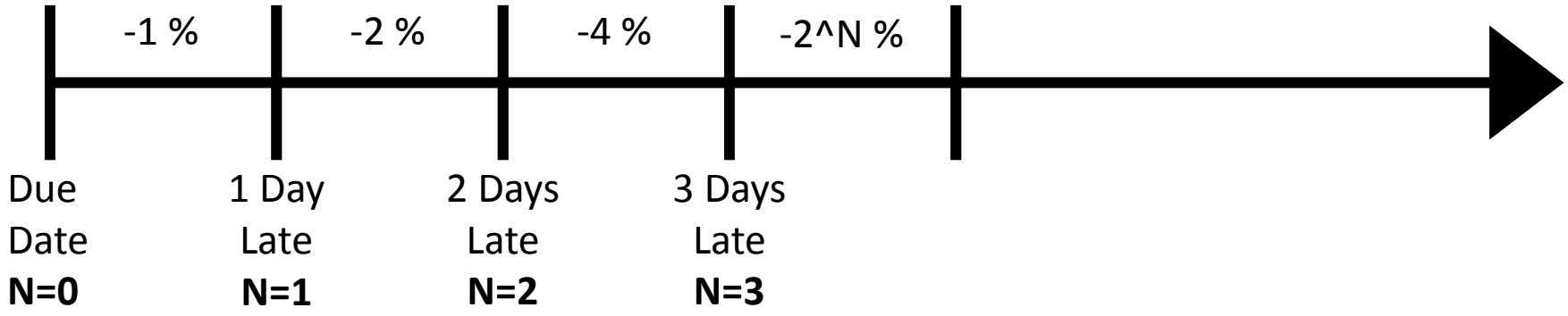
- Prerequisites: CS 31

Grading

- Assignments: 50% (equally weighted)
  - 9 regular assignments
    - Lab exercises (expected to be done in the lab)
    - Homework (expected to be done at home)
    - Generally due on Gradescope on Friday at 11:55 PM
  - 1 Presentation + Report (Assignment 10)
    - 10 minute presentation and research report
    - Topic from select CS news publications (listed on course website)
    - Will coordinate scheduling later in the quarter
- Final Exam: 50%
- All assignments are to be done **individually**, and then submitted on Gradescope

Lateness Penalty

- Lateness penalty: **2^N** % deduction for being up to **N** days late
  - Presentation + Report assignment must be submitted on time
  - No submissions after last day of instruction



Syllabus

1. Introduction to Linux
2. SSH and its applications
3. Bash Shell Scripting
4. Modifying and Rewriting Software
5. C Programming and Debugging
6. Systems Call Programming and Debugging
7. Dynamic Linking
8. Basic Change Management
9. Change Management Exploration
10. Research Presentations

Introduction to Linux

Linux Distribution

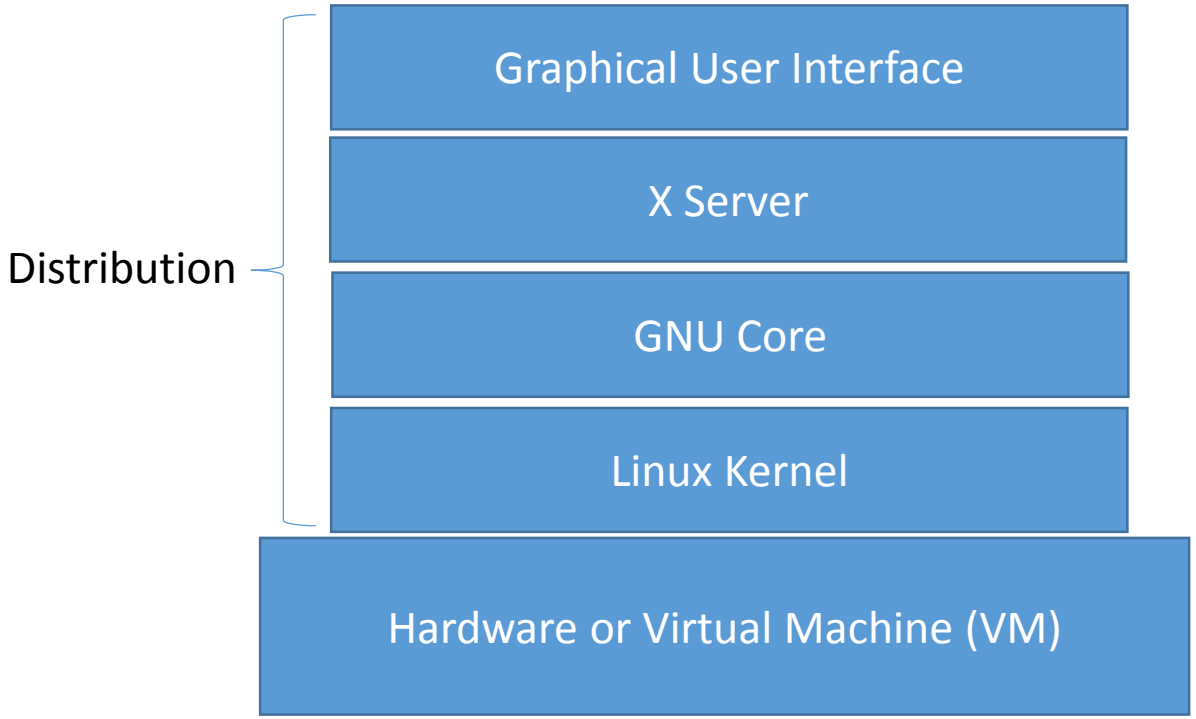
- Distribution:

  A **Linux distribution** (**distro**) is an operating system made from a software collection, which is based upon the Linux Kernel and software tools to automate the installation, configuration, upgrade, and removal of programs in a consistent manner.

- Example Distributions:
  - CentOS
  - Red Hat Enterprise Linux (RHEL)
  - Fedora
  - Ubuntu
  - Debian
  - OpenSUSE

Linux Distribution

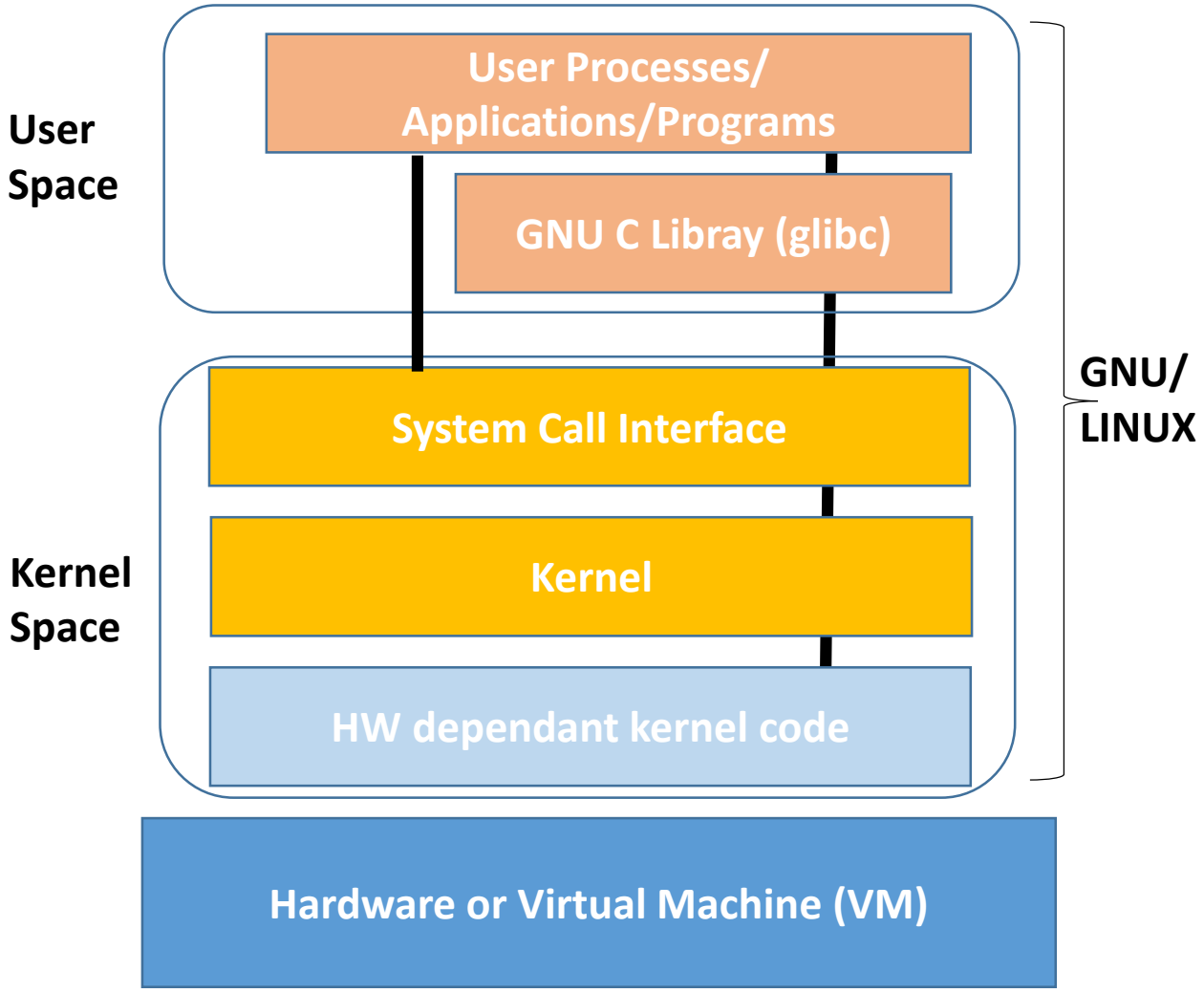
- **Linux Kernel:** Open source software that facilitates the interaction between the hardware and the applications
- **GNU core:** Is the basic file, shell and text manipulation utilities. Contains tools such as cat (display), ls (list), rm (remove), etc.
- **X Server:** The display server for the X Windows systems. Allows interaction with keyboard, mouse and screen
- **Graphical User Interface (GUI):** Display environment



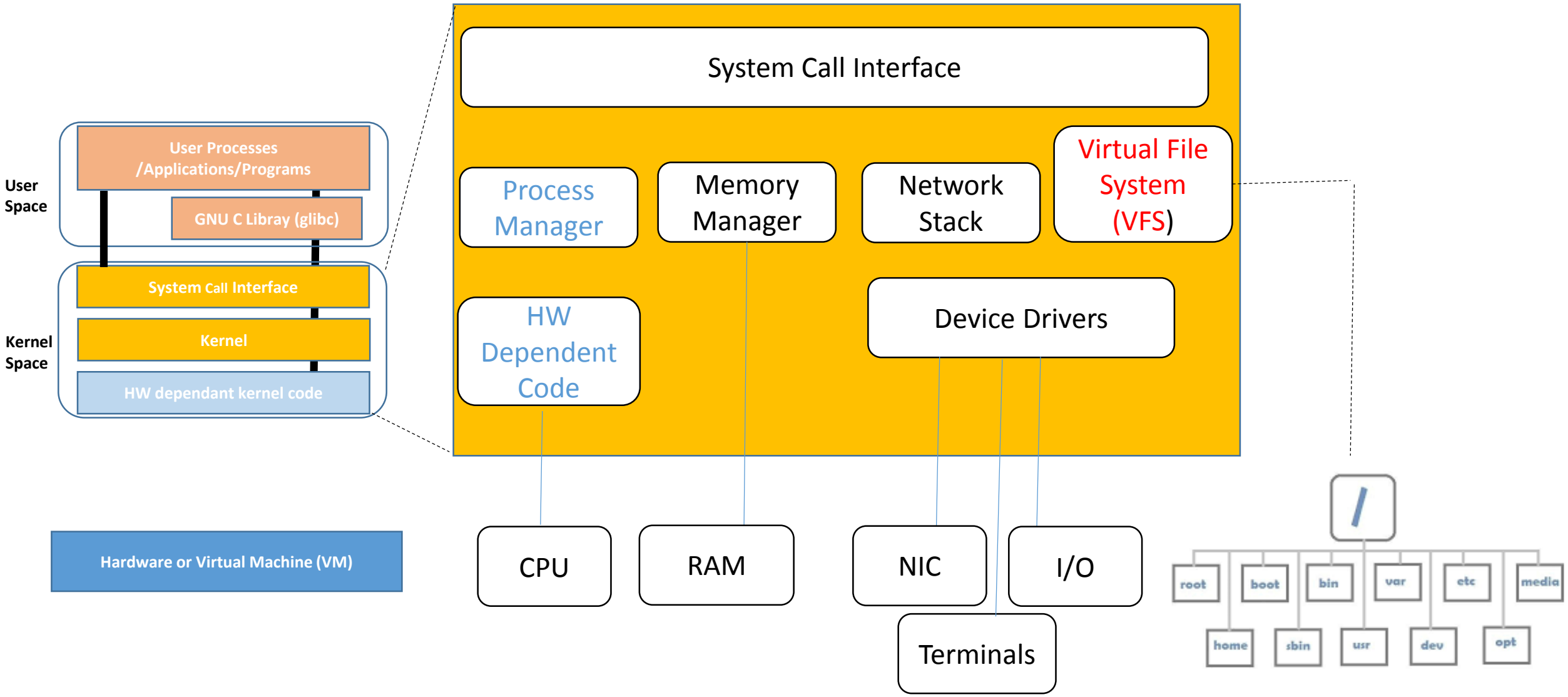
User vs. Kernel Space

- **User Space:** Space in memory allocated to the applications and Libraries. It allows the user and application to interact with the system.
  - ➢The user space varies between the different Linux distributions.

- **Kernel Space:** Contains the system call Interface, the kernel sub-systems and the HW dependant code. Connects the user space to the HW (CPU, MEM, Networking, etc.).
  - ➢The kernel is the same between the different distributions.



Kernel Sub-systems



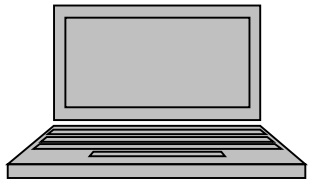
Linux GUI (Desktops)

- Gnome, KDE, Unity, Cinnamon, Mate, XFCE, LXDE …
- Range between simple, complex, resource intensive, resource light, etc.
- Basic desktop functionality: launching desktops, browsers, etc.
- Checking system resources, files, directories, etc.
- However ! Heavy lifting in Linux is still done inside a terminal via a Command Line Interface (CLI)

Linux CLI

- Opening a Terminal:
  - From Linux desktop – Open terminal
  - From MacOS – Open terminal

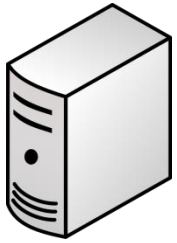
Local Client
(ssh client)



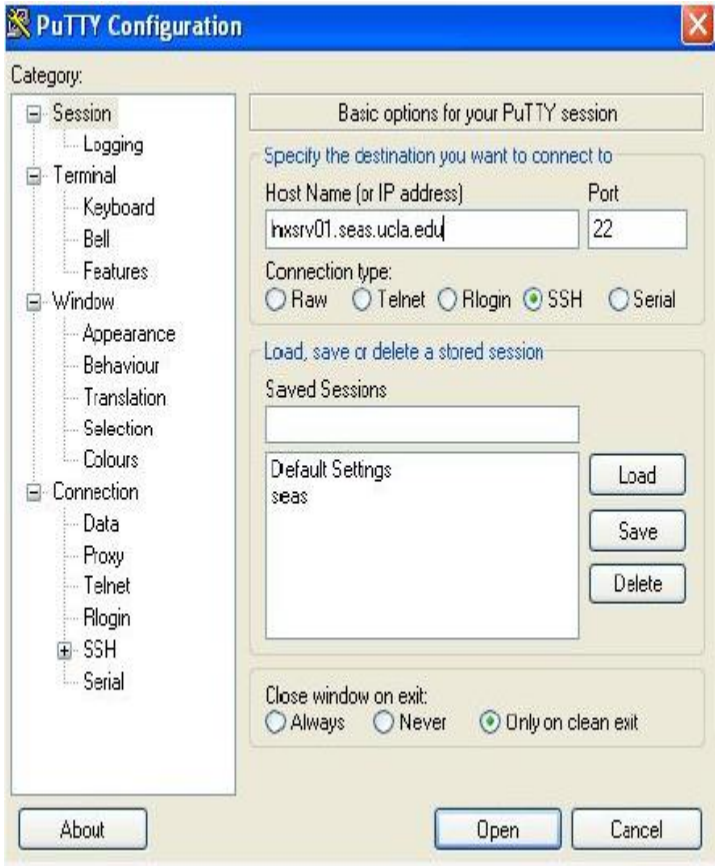
1- $ ssh username@lnxsrv.seas.ucla.edu
(uses TCP port 22)

2- pw: xxxx

Remote Server



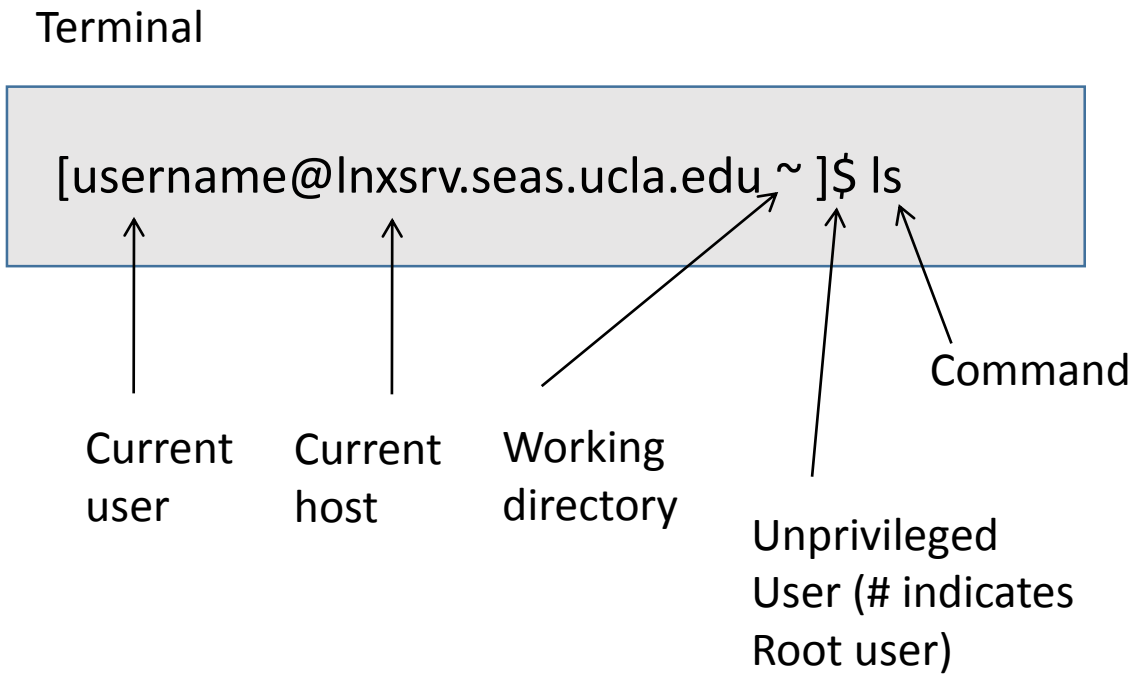
- Opening a Terminal:
  - From Windows – Open a terminal emulator (ex: PuTTY)

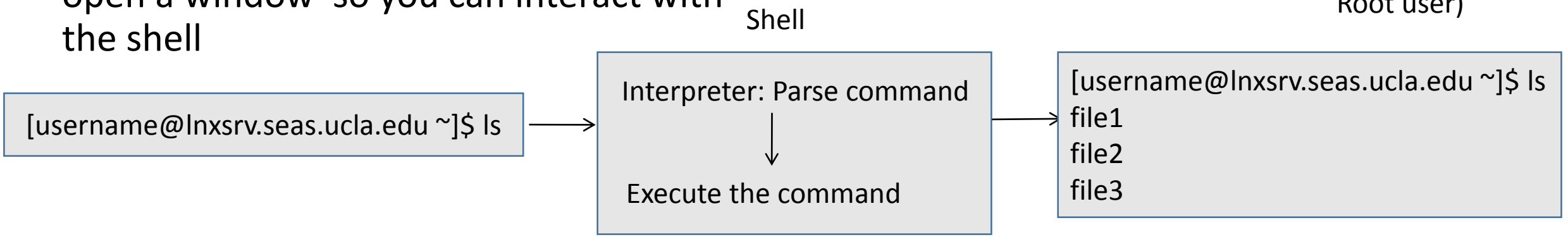


User Input

- **Linux Shell:** Is a command line interpreter that lives in the user space. It takes input from the user, acts on it, and sends the output to the display. The shell is also a programming language. Example shells are BASH (Born Again Shell)

- **Linux Terminal**: Is a program that lets you open a window so you can interact with the shell





The File System

/	# the root directory
/bin	# user binaries (ls, pwd, etc.)
/boot	# static boot files.
/dev	# device files
/etc	# configuration files
/home	# users home directories
/lib	# shared libraries
/mnt	# temporary mount points
/opt	# optional packages
/ proc	# kernel and process files
/root	# root user home directory
/run	# application state files
/sbin	# system administration libs
/srv	# data for services
/tmp	# temporary files
/usr	# user binaries
/var	# variable data files

Note: These are the basic directories, however you might see more directories depending on the specific Linux distribution

Moving Around in Linux

pwd	# print working directory
ls	# list directories/files
cd <new location>	# change working directory
.	# current directory
~	# home directory
/	# root directory or separator
cd -	# go back to last location
cd ..	# go back one level up
mkdir	# make directory
rmdir	# remove an empty directory
cat	# concatenate and print files
cp	# copy files
echo	# write arguments to standard output

mv	# move files
ln	# link files
rm	# remove directory entries
chmod	# change the file modes
kill	# terminate or signal processes
ps	# report process status

Note: You can hide files and directories by preceding them with a dot. Ex: .file1

- ls –a : show hidden files and directories
- ls –d : list only directories
- ls –l : show long listing + permission info
- ls –s : show size of each file in blocks
- ls –h : human readable form (in Byte/KB/MB…)

CLI History

- **<up arrow>:** previous command
- **<tab>:** auto-complete
- **!!:** replace with previous command
- **![str]:** refer to previous command with str
- **^[str]:** replace with command referred to as str

Absolute vs. Relative Path

- Absolute path: Starts from the root directory

cat /home/myfiles/file1

Absolute path

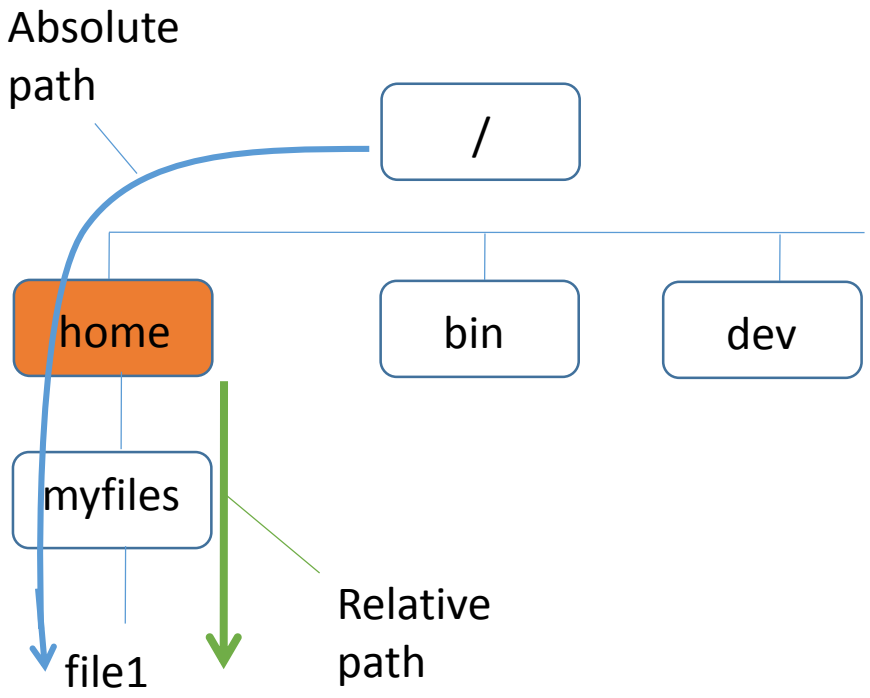
- Relative path: Starts from the working directory

cd /home/myfiles

Working directory

cat file1

Relative path



Everything is a file

- In Linux everything is a file. If it is not a file, then it is a process
- Linux has 3 types of files:
  - Ordinary/Regular files
  - Special files
  - Directories

- **Ordinary files:**
  - Text
  - Data
  - Program instructions
  - Image files
  - Compressed files
  - etc.

- **Special files:**
  - Block files: These are device files (/dev). They provide a buffered access to hw components.
  - Character files: These are device files (/dev). They provide un-buffered access to hw components.
  - **Symbolic link files**: These are pointers to other files
  - Pipes or named pipes: files that allow inter-process communication (IPC)
  - Sockets: files that allow IPC between different environments

- **Directories:** files that store ordinary files or special files in a hierarchy, starting for the root (/) directory.

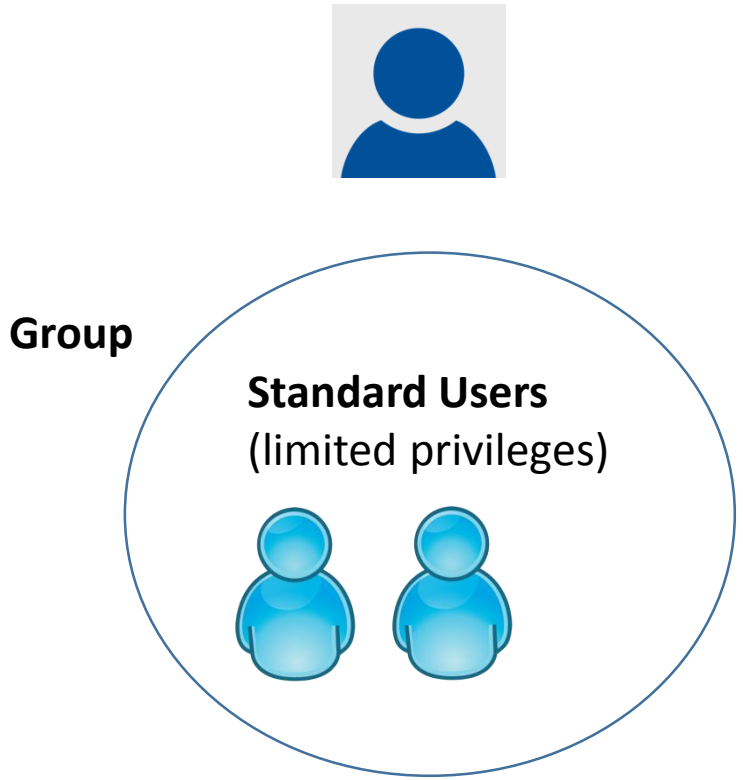
Processes

- An instance of a computer program in execution.
- Types of processes:
  - Foreground (interactive)
  - Background (non-interactive/automatic)
- Daemons**:**
  - Background processes
  - Usually started at boot time
  - Example: **cron**
    - Can schedule jobs to run periodically at certain times (cron jobs)
    - One use: Run backups every month
- Commands:
  - ps (list processes that are running)
  - kill
    - Send signals to process (usually to kill)
    - kill [signal or option] PID…

Types of Accounts

- Root User:
  - Has privilege to all system resources
  - System level administration tasks

- Standard Users:
  - Unprivileged user account
  - Provided login Shell
  - Home directory
  - “Viewing” for system configurations
  - Can perform privileged actions via sudo (super-user-do) command (don’t sudo on SEASnet servers!)
  - Users are assigned to “Groups”.



Files and Directory Permissions

```
$ ls -l
total 120024
-rw-r--r-- 1 sam 197609 47185920 Mar 22  2018 boot2docker.iso
-rwxr-xr-x 1 sam 197609 38266880 Mar 22  2018 docker.exe*
-rwxr-xr-x 1 sam 197609  7551000 Mar 22  2018 docker-compose.exe*
-rwxr-xr-x 1 sam 197609 28503040 Mar 22  2018 docker-machine.exe*
-rw-r--r-- 1 sam 197609    69694 Mar 22  2018 docker-quickstart-terminal.ico
drwxr-xr-x 1 sam 197609        0 Sep 17  2018 installers/
drwxr-xr-x 1 sam 197609        0 Sep 17  2018 kitematic/
```

File (-)
Directory (d)
Soft link (l)

User (u) **Group (g)** **Other (o)**

Example: d rwx r-x r-x sam ... installers/

Directory (installers)

User: read write execute

Group (sam:) read execute

Other: read execute

- **Modifier Rights:** read (r), write (w), execute (x)
- **Access Rights:** user, group, other, all

Change mode (chmod)

- Chmod changes the permissions (mode) of the file

- **Octal representation**:
  - chmod 777 file1 # changes the file1 permissions to rwx rwx rwx

- **Symbolic representation**:
  - chmod ug = rx file # changes the file1 user and group permissions to rx

Change mode (chmod) - Symbolic

Reference	Class	Description
u	user	Owner of the file
g	group	Users who are members of the group
o	other	Users who are not owner or member of group
a	all	u + g + o

Operator	Description
+	Adds the specified modes to the specified classes
-	Removes the specified modes from the specified classes
=	The modes specified are to be made the exact modes of the specified classes

Mode	Name	Description
r	read	Read a file or a list a directory's contents
w	write	Write to a file or directory
x	execute	Execute a file or recurse a directory tree

chmod [references] [operator] [modes],… file…

chmod uo=rw file1 (make user and others rw)
chmod a-x file1 (remove execution from all)
chmod u+r file1 (add read permission to user)

Special Modes:
s – setuid/setgid – execute with permissions of the owner/group
t – sticky bit – (for directories) only owners can rename/unlink files in directory

Change mode (chmod) - Octal

#	binary	Permission
7	1 1 1	r w x
6	1 1 0	r w -
5	1 0 1	r - x
4	1 0 0	r - -
3	0 1 1	- w x
2	0 1 0	- w -
1	0 0 1	- - x

chmod octal_number file…

Ex: chmod 777 file1 (make user: rwx, group, rwx, other: rwx)
chmod 744 file1 (make user: rwx, group: r--, other: r--)

man pages

- man: Display manual page for a Linux command
- DESCRIPTION: man is the system's manual pager. Each page argument given to man is normally the name of a program, utility or function. The manual page associated with each of these arguments is then found and displayed.
- Hit “q” to quit the man pages.
- **man** [command name] (e.g. man ls, man pwd, man rmdir, etc.)
- **man** [section] [command name]
  - 1- User Commands
  - 2- System Calls
  - 3- C Library Functions
  - 4- Devices and Special Files
  - 5- File Formats and Conventions
  - 6- Games et. al.
  - 7- Miscellanea
  - 8- System Administration tools and Daemons

Note:

/keyword: forward slash followed by word to search for within man page

Linux Wildcards

- Asterisk (*) – matches on one or more occurrences of a character
  - `cd ~ ,  $touch file1 file2 file3`
  - `$ls -l f*   # lists all files (and directories) that start with f`
- Question mark (?) – matches on a single occurrence of a character
  - `cd ~ ,  $touch fee fo fi fum`
  - `$ls -l f??   # lists all files that start with f and have two more characters (fee, fum)`
- Brackets ([]) – matches any occurrence of characters enclosed in brackets
  - `cd ~ , $touch fee fo fim fum`
  - `ls –l f[foiu]m   # list all files that start with f, end with m and have f,o,I,u,m as middle character  (fim, fum)`

Redirection

- **> *file*:** write stdout to a file (potentially overwriting)
- **>> *file*:** append stdout to a file
- **< *file*:** use contents of a file as stdin

“find” command

- find - search for files in a directory hierarchy
  - find [starting-point] [expression]
    - [starting-point] # is a directory tree rooted at starting-point
    - [expression] # is a query specification to match on files and take action

- [expression]
  - -name # name of a file
  - -type # type of a file (regular, directory, symbolic link, etc.)
  - -user # owner of a file
  - -perm # permission of a file
  - -maxdepth # how many levels to search
  - -mtime # last modified by day

“find” Examples

- find /bin –name ‘m?’
  - Lists all the two letter files that start with m
  - /bin/mv
  - /bin/mt

- find /bin –type l –perm ‘a=rwx’
  - Lists all files of type soft link with u+g+o permissions as rwx

Other useful commands

- which <command> locate the binary of the command
  - Ex: $ which ls
    /bin/ls

- whatis <command> returns name section of man page
  - Ex: $whatis ls
    ls (1) - list directory contents
    ls (1p) - list directory contents

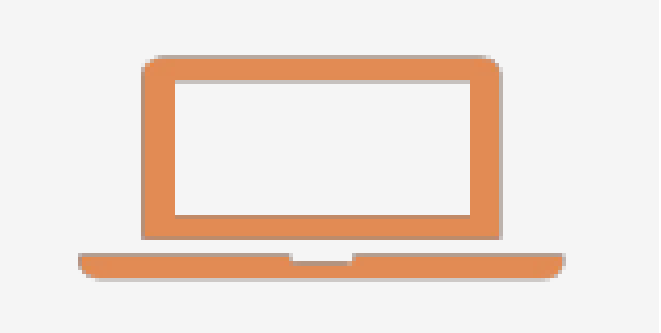
- whereis <command> locate the binary, source, and manual page files for the command
  - Ex: $whereis ls
    /bin/ls
    /usr/share/man/man1/ls.1.gz
    /usr/share/man/man1p/ls.1p.gz

Lookup the following commands

- touch - change file timestamps
- cat - concatenate files and print on the standard output
- head - output the first part of files
- tail - output the last part of files
- du - estimate file space usage
- ps - report a snapshot of the current processes
- kill - send a signal to a process
- diff - compare files line by line
- cmp - compare two files byte by byte
- wc - print newline, word, and byte counts for each file
- sort - sort lines of text files
- find - search for files in a directory hierarchy (careful !!)

How to connect to Linux Server

- Connect to your SEAS account from your PC/Mac



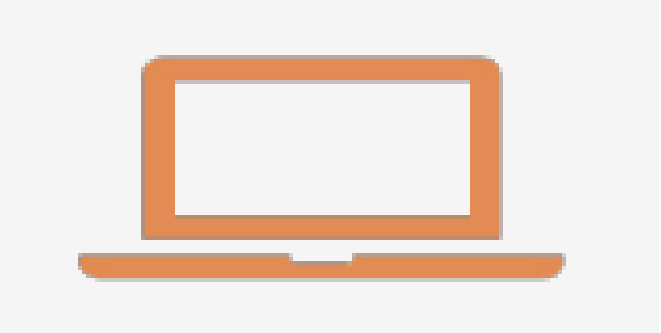
- VPN: www.it.ucla.edu/bol
  - Services -> VPN

- Putty or SSH: www.seasnet.ucla.edu/lnxsrv
  - lnxsrv.seas.ucla.edu (lnxsrv06, lnxsrv07, or lnxsrv09)

Connect to SEAS from OS X or Linux

- Terminal

- $ ssh username@lnxsrv.seas.ucla.edu
  - (lnxsrv06, lnxsrv07, or lnxsrv09)
  - username = SEAS username

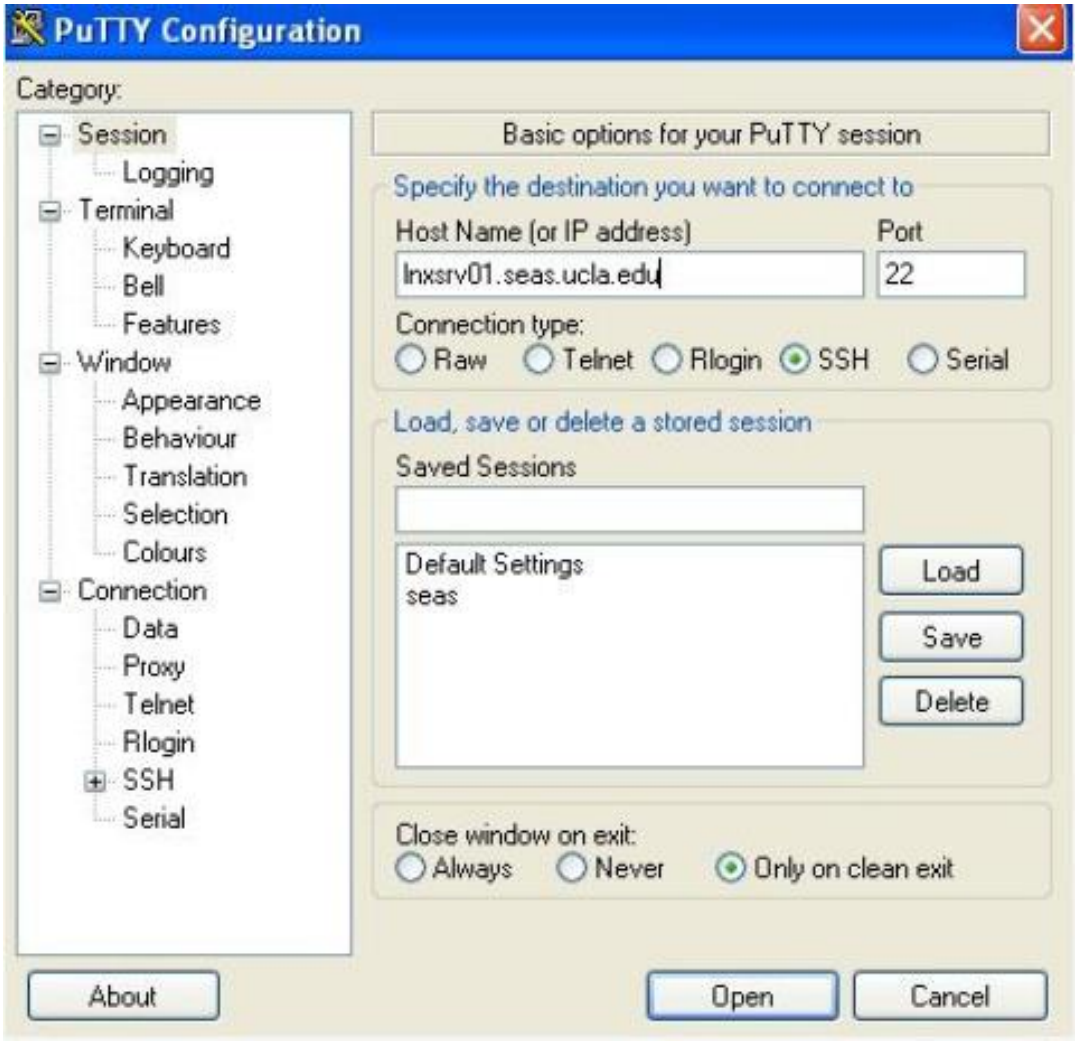


Connect to SEAS from Windows

- Putty

- Recommended
- Small and easy to use
- Host name: lnxsrv.seas.ucla.edu (lnxsrv06, lnxsrv07, or lnxsrv09)
- User name: your SEAS username

- Detailed setup steps:
  - www.seasnet.ucla.edu/lnxsrv



SCP

- Secure Copy Protocol

- `scp username@serveraddr:filepath destination-filename`

- **Copy the file "foobar.txt" from a remote host to the local host**
- scp your_username@remotehost.edu:foobar.txt /some/local/directory
- **Copy the file "foobar.txt" from the local host to a remote host**
- scp foobar.txt your_username@remotehost.edu:/some/remote/directory

- On Windows: Putty comes with **pscp** (command prompt)
- http://www.hypexr.org/linux_scp_help.php

Assignment 1 - Lab

- ans1.txt for LABORATORY Section of assignment

```
John Smith
123456789
Assignment 1
Lab
------------

1. Answer to question 1

2. Answer to question 2

3. Answer to question 3

....
```

Assignment 1 - HW

- key1.txt for HOMEWORK Section of assignment

```
John Smith
123456789
Assignment 1
Homework
------------

1. C-s H e l l o SPC W O R L D

2. C-s H T M L

3. C-p

4. M-x g o t o - l i n e Enter 1 2 3 Enter

....
```

Assignment 1 – Emacs Quickstart

Note:
C: Ctrl
M: Alt or Meta
SPC: Space

- Start Emacs: type **emacs** in shell
- Create new file: C-x C-f *filename* (e.g. C-x C-f ans.txt)
- Save current file: C-x C-s
- Quit Emacs: C-x C-c