ECE327 - Introduction to Software Engineering

Lab 1 - Linux/Unix Tutorial

Your First Technical Course for Industry Preparation

Shuyue Jia

Ph.D. student

Boston University

BU Department of Electrical & Computer Engineering

BOSTON UNIVERSITY

Outline

- Introduction to Linux/Unix
- The Linux Shell
- Files and Directories
- Navigating Through Directories
- Commands for Files
- Code Editors
- Security and Access Rights
- Check Manuals
- Zipping Files and Directories

Part 1 – Introduction to Linux/Unix

- Linux/Unix: Computer **Operating System** (OS)
- **Unix**: *Proprietary*, developed at AT&T Bell Labs in the 1970s
  - By Kenneth Lane Thompson & Dennis Ritchie (Father of C)
  - **ACM A. M. Turing Award**, 1983 (Ken and Dennis)
- **Linux**: *Free and open-source*, developed by Linus Torvalds in the 1990s
- Core Philosophy:
  - Small, modular utilities that perform tasks well
  - Combine tools through pipelines to solve problems

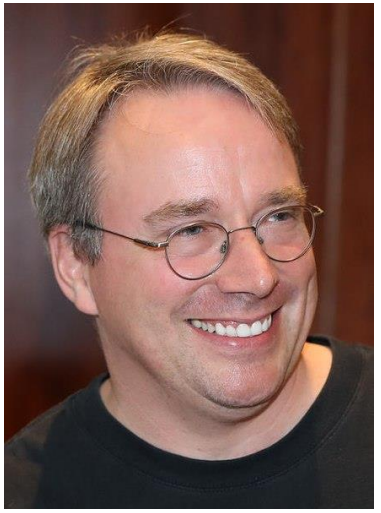


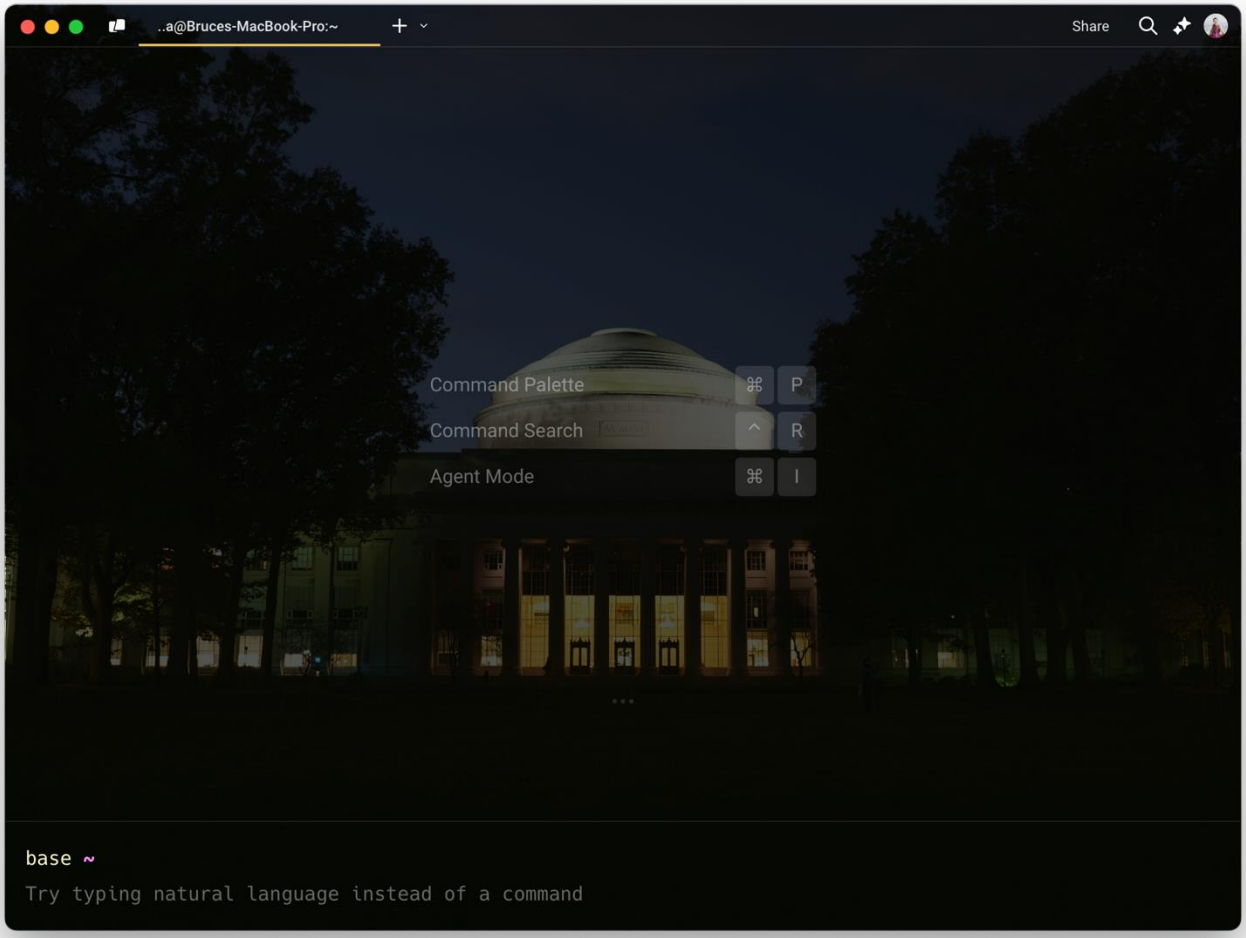
Image Credits:
1. Ken Thompson and Dennis Ritchie 1973: in the public domain
2. Linus Torvalds: CC BY 3.0

Part 1 – Introduction to Linux/Unix

- **Key Features**:
  - **Portability**: runs on various hardware platforms
  - **Multiuser Support**: multiple users at the same time
  - **Multitasking**: multiple processes at the same time
  - **Hierarchical File System**: organized directory structure
  - **Security**: permissions and user roles
- **Standardization and Foundation for Modern OS**
  - Portable Operating System Interface (POSIX), IEEE 1003/ISO/IEC 9945
  - Linux and macOS

Part 2 – The Linux Shell – Shells

- Textual UI to the OS kernel
- Accessible via the Terminal



macOS

- Terminal
- Hyper
- Warp
- iTerm2

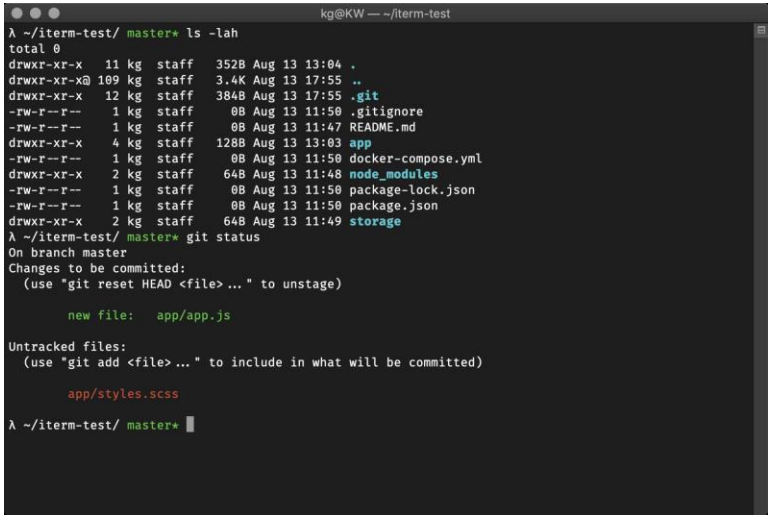
Part 2 – The Linux Shell – Zsh


```
brucejia — -bash — 80×27
Last login: Mon Jan 20 12:21:55 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Bruces-MacBook-Pro:~ brucejia$ cat /etc/shells
# List of acceptable shells for chpass(1).
# Ftpd will not allow users to connect who are not using
# one of these shells.

/bin/bash
/bin/csh
/bin/dash
/bin/ksh
/bin/sh
/bin/tcsh
/bin/zsh
(base) Bruces-MacBook-Pro:~ brucejia$ echo $SHELL
/bin/bash
(base) Bruces-MacBook-Pro:~ brucejia$ bash --version
GNU bash, version 3.2.57(1)-release (arm64-apple-darwin24)
Copyright (C) 2007 Free Software Foundation, Inc.
(base) Bruces-MacBook-Pro:~ brucejia$ chsh -s /bin/zsh
Changing shell for brucejia.
Password for brucejia:
(base) Bruces-MacBook-Pro:~ brucejia$
```

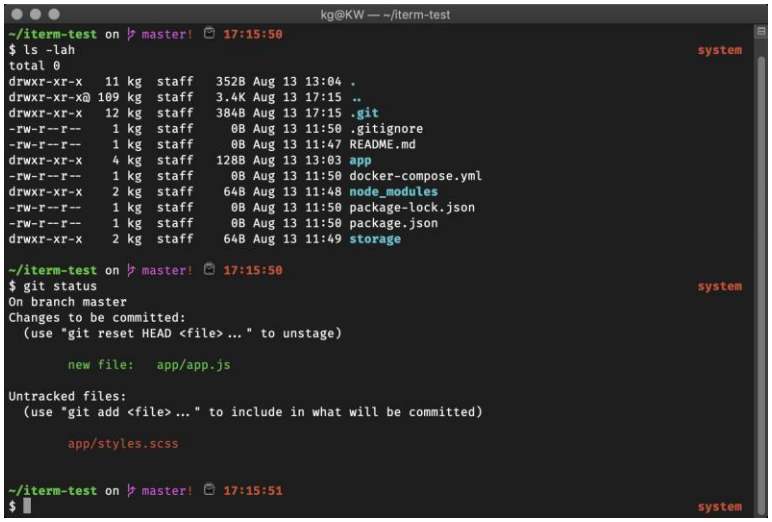
- Oh My Zsh
- Installation
- Themes
- Plugins
- Autosuggestions
- Syntax-Highlighting
- Autocomplete

Image Credits: Oh My Zsh

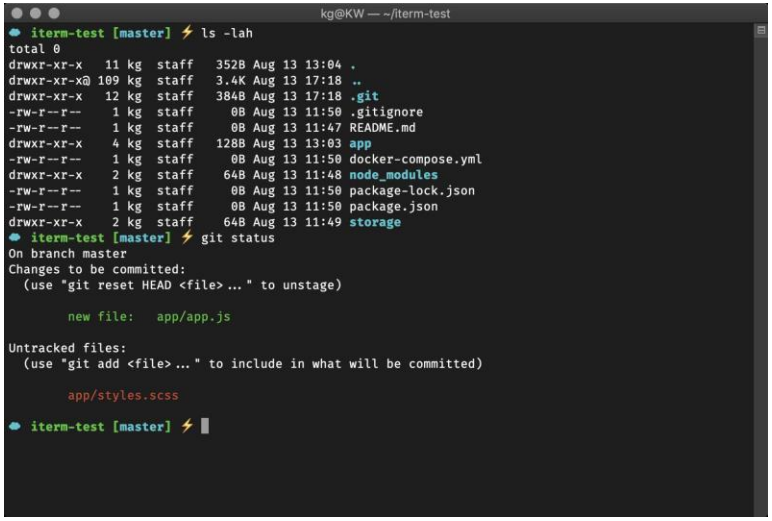
Part 2 – The Linux Shell – Themes



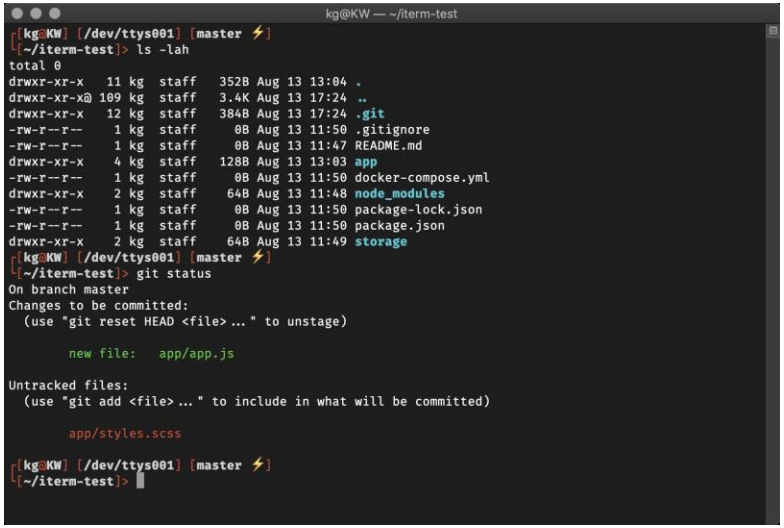
lambda



amuse



cloud



darkblood

Part 2 – The Linux Shell – Plugins

- Oh My Zsh
- Installation
- Themes
- Plugins
- Autosuggestions
  Syntax-Highlighting
  Autocomplete

Image Credits: Oh My Zsh

Part 2 – The Linux Shell

- Nice Features of Terminal
  - **History**: Up/Down Arrows

    ```
    history
    ```

  - **Auto-completion**: `Tab` key

- Mostly every command takes arguments

  ```
  <command> [<list of arguments>]
  ls -l
  cd ./my/directory
  ```

Part 3 – Files and Directories

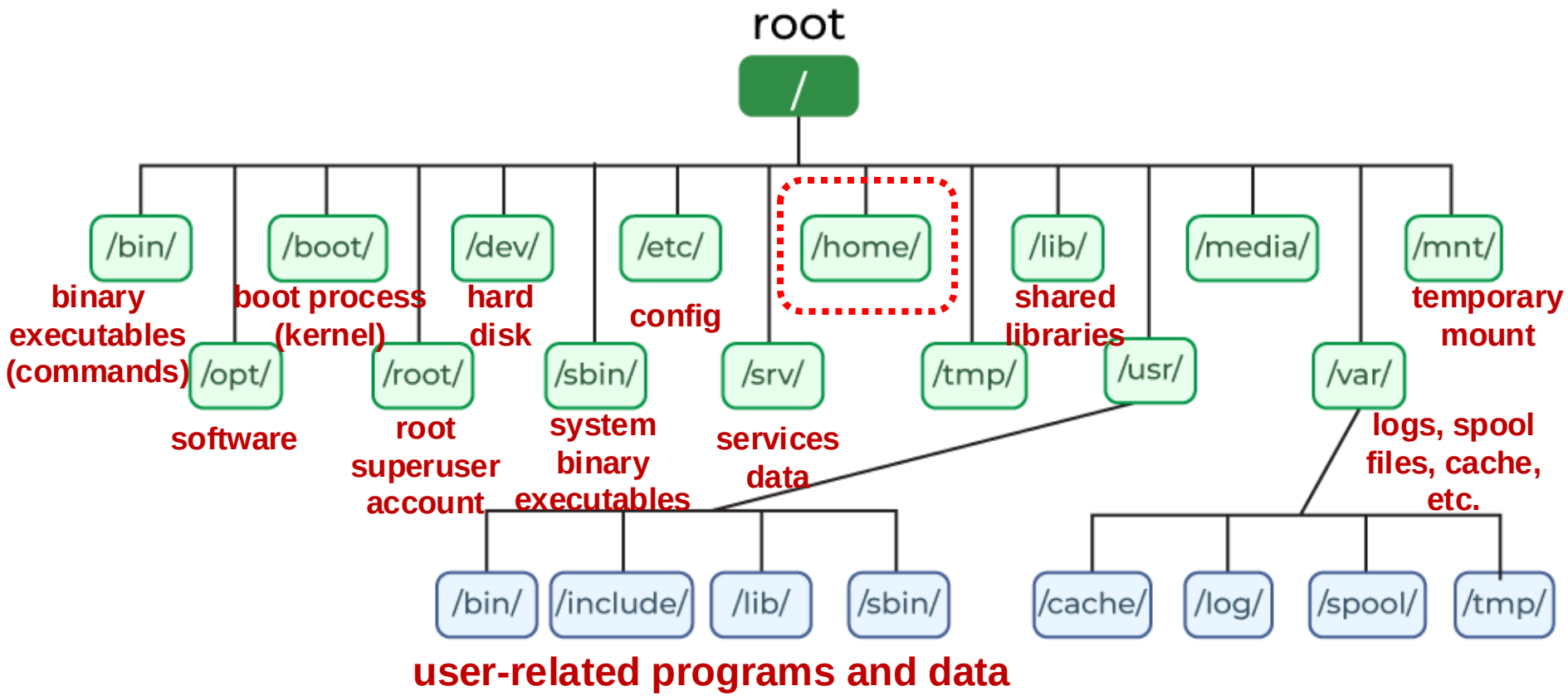
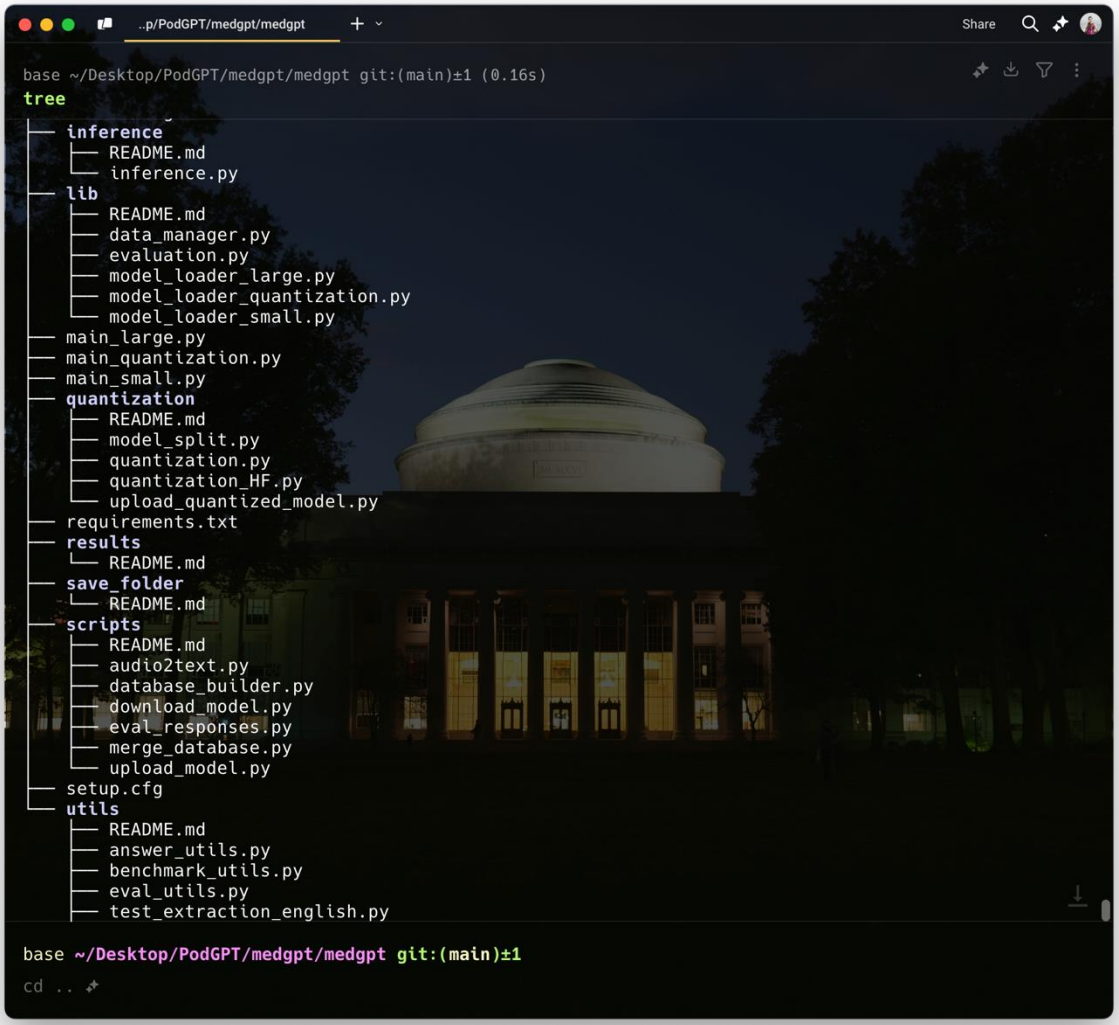


Image Credits: in the public domain

Part 3 – Files and Directories

$ tree



🖼️ Structure of the code

At the root of the project, you will see:

```
├── config_benchmark.yml
├── config_chatgpt.yml
├── config_large.yml
├── config_quantization.yml
├── config_small.yml
├── main_large.py
├── main_quantization.py
├── main_small.py
├── lib
│   ├── data_manager.py
│   ├── evaluation.py
│   ├── model_loader_large.py
│   ├── model_loader_quantization.py
│   └── model_loader_small.py
├── inference
│   └── inference.py
├── quantization
│   ├── model_split.py
│   ├── quantization.py
│   ├── quantization_HF.py
│   ├── quantization_GPTQModel.py
│   └── upload_quantized_model.py
├── download_files
│   ├── download_model_from_hf.py
│   └── download_model_to_local.py
├── requirements.txt
├── benchmark
├── results
├── save_folder
├── scripts
│   ├── audio2text.py
│   ├── database_builder.py
│   ├── download_model.py
│   ├── deployment.py
│   └── upload_model.py
└── utils
    ├── answer_utils.py
    ├── benchmark_utils.py
    ├── eval_utils.py
    └── utils.py
```

Image Credits: https://github.com/vkola-lab/PodGPT?tab=readme-ov-file#%EF%B8%8F-structure-of-the-code

Part 3 – Files and Directories – Listing Files

- `ls`
  - **Ls**ts all files and directories within your current directory
- `ls -l`
  - Expanded view of files and directories (`-l` is **l**ong listing format)
- `ls -a`
  - **Ls**ts **a**ll files including hidden ones (`-a` is **a**ll files)
  - Names of hidden files start with “.”

Part 3 – Files and Directories – File Storage

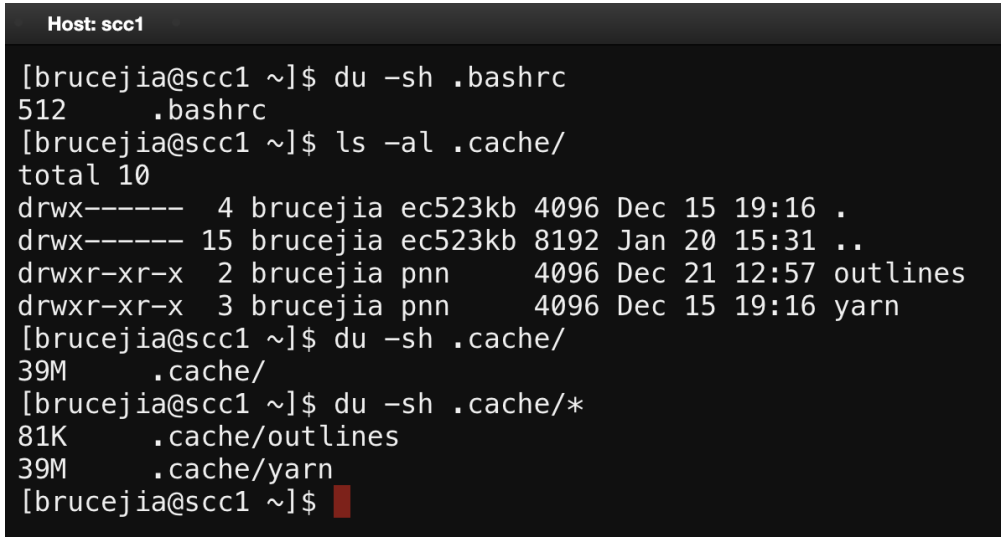
- `df -h`
  - **D**isk **f**ree, displays the amount of available disk space on file systems

Host: scc1

```
[brucejia@scc1 ~]$ df -h
Filesystem                                Size  Used Avail Use% Mounted on
devtmpfs                                  126G     0  126G   0% /dev
tmpfs                                     126G   26M  125G   1% /dev/shm
tmpfs                                     126G  2.1G  123G   2% /run
tmpfs                                     126G     0  126G   0% /sys/fs/cgroup
/dev/sda2                                  20G   13G  5.8G  69% /
/dev/sda7                                 2.0G   43M  1.8G   3% /tmp
/dev/sda5                                 7.8G  1.5G  6.0G  20% /var
/dev/sda1                                 488M   72M  380M  16% /boot
/dev/sda6                                 2.0G   32K  1.8G   1% /home
/dev/sda8                                 876G   55G  778G   7% /scratch
sccsvcp:/var/spool/mail                    16G  2.5G   13G  17% /var/spool/mail
scc-fs6vp:/gpfs4/local                     14P   11P  2.8P  81% /usr/local
scc-fs7vp:/gpfs4/share                     14P   11P  2.8P  81% /share
scc-fs7vp:/gpfs4/home/usr3                 14P   11P  2.8P  81% /usr3
scc-fs7vp:/gpfs4/projectnb                 14P   11P  2.8P  81% /projectnb
sccdr:/gpfsdr1/project                    466T  222T  245T  48% /project-backup
scc-fs7vp:/gpfs4/home/usr1                 14P   11P  2.8P  81% /usr1
scc-fs7vp:/gpfs4/home/usr4                 14P   11P  2.8P  81% /usr4
sccdr:/gpfsdr1/stash                      466T  222T  245T  48% /stash
scc-fs7vp:/gpfs4/home/usr2                 14P   11P  2.8P  81% /usr2
scc-fs7vp:/gpfs4/project                   14P   11P  2.8P  81% /project
```

Part 3 – Files and Directories – File Storage

- `du -sh`
  - **D**isk **u**sage, displays detailed information about the disk space



Check a folder's size

```
du -sh <folder_name>
```

Check a file's zie

```
du -sh <file_name>
```

Check the folders' size inside a folder

```
du -sh <folder_name>/*
```

Part 4 – Navigating Through Directories

- `pwd:` **p**rints **w**orking **d**irectory
- `cd :` **c**hange **d**irectory
  - `cd .` : go to the current directory
  - `cd ..` : go to directory one level above
  - `cd` : go to the home directory
  - `cd -` : go to the previous directory
- **Absolute Path** : `cd /absolute/path/to/directory`
- **Relative Path** : `cd ./relative/path/to/directory`

While you are writing codes,
the **relevant path** is **highly recommended**!

Part 5 – Commands for Files

- `touch`: create a new file

```
touch main.cpp
```

- `mkdir`: **m**a**k**e a new **dir**ectory

```
mkdir EC327_Lab1
```

- `cp`: **c**o**p**y file

```
cp main.cpp main01.cpp

cp main.cpp /destination
```

- `mv`: rename or **m**o**v**e file

```
mv main1.cpp main2.cpp

mv main.cpp /path/to/destination
```

- `rm`: **r**emo**v**e file

```
rm main.cpp
```

It will **NOT** open for editing in the Terminal!

Important!
If you want to copy a folder of files,
Please use `-r` "recursive"

```
cp -r source/ destination/
```

Important!
If you want to delete a folder of files,
Please use `-r` "recursive" and `-f` "force"

```
rm -rf source_folder
```

NOTICE:
Do NOT easily use `rm -rf`

Part 5 – Commands for Files

- `wc`: **W**ord **c**ount, print **number of lines**, **words**, and **characters**

```
wc <file_name.txt>
```

`wc -l`: print **number of lines**

```
wc -l <file_name.txt>
```

`wc -w`: print **number of words**

```
wc -w <file_name.txt>
```

`wc -c`: print **number of bytes**

```
wc -c <file_name.txt>
```

`wc -m`: print **number of characters**

```
wc -m <file_name.txt>
```

`wc -L`: print the length of longest (number of characters) line

```
wc -L <file_name.txt>
```

Part 5 – Commands for Files – Reading Files

`less <filename>`

- Example: `less stuff.txt`

Navigating

- `spacebar`: go to the next page
- `b` : go back a page
- `q` : quit back to shell
- `v` : open file in editor

`less -N <filename>`: shows line numbers

Part 6 – Code Editors – `vim`

- `vi <filename>`: opens vi editor
- Two modes:
  - `i`: switches to insert mode
  - `ESC`: go back to EDIT mode
- Some EDIT mode commands
  - `:q` : quit
  - `:q!` : force quit
  - `:wq` : write and quit
  - `dd` : delete this line
  - `5G` : go to line 5
  - `u` : undo
  - `G` : go to the last line
  - `gg` : go to the first line
  - `o` : make a new line below
  - `5j` : jump 5 lines below
  - `5k` : go back 5 lines above

VERY IMPORTANT!

If you want to learn more `vim` commands or need help:

https://www.cs.colostate.edu/helpdocs/vi.html

Part 6 – Code Editors – emacs

- `emacs <filename>:` starts emacs or open a file
- Some emacs commands (`C` is the `Ctrl/Control` key):
  - `C-f` : Move to the next character
  - `C-b` : Move backward to a character
  - `C-n` : Move to the next line
  - `C-p` : Move to the previous line
  - `C-x C-s` : save file
  - `C-x C-c` : exit emacs

If you want to learn more documentation or download emacs:

https://www2.lib.uchicago.edu/keith/tcl-course/emacs-tutorial.html

Part 7 – Security and Access Rights

- `ls -l:` list files in expanded form

```
localhost:Labs ernstl$ ls -l
total 0
drwxr-xr-x  2 ernstl  staff  68 Sep  6 15:43 Lab1
-rw-r--r--  1 ernstl  staff   0 Sep  6 15:49 inc.h
-rw-r--r--  1 ernstl  staff   0 Sep  6 15:49 main.cpp
-rw-r--r--  1 ernstl  staff   0 Sep  6 15:49 my.h
```

- `d rwx r-x r-x : user group world`
- `d :` Directory (if it is file, it is "–")
- `r :` read
- `w :` write
- `x :` execute

Example:

User: ernstl

Group: staff

Part 8 – Check Manuals

- Man is an abbreviation for **Manual**

  `man <command>` : will open the manual in the `vim` editor

- Examples:

```
man ls
man cd
man g++
```

Part 9 – Zipping Files and Directories

- `zip` and `unzip` command:

  `zip <filename.zip> <file>`

  `zip -r <filename.zip> <directory>`

  `unzip <filename.zip>`

- **Zip Bomb Issue**

  If your zip file is very large and triggers ZIPBOMB DETECTION, you can use the following method to unzip your file:

```
UNZIP_DISABLE_ZIPBOMB_DETECTION=TRUE unzip <filename.zip>
```

Extra Part – Additional Readings

- **Unix tutorial for beginners**

https://users.cs.duke.edu/~alvy/courses/unixtut/

- **Learn Unix in 10 minutes**

https://networking.ringofsaturn.com/Unix/learnUNIXin10minutes.php

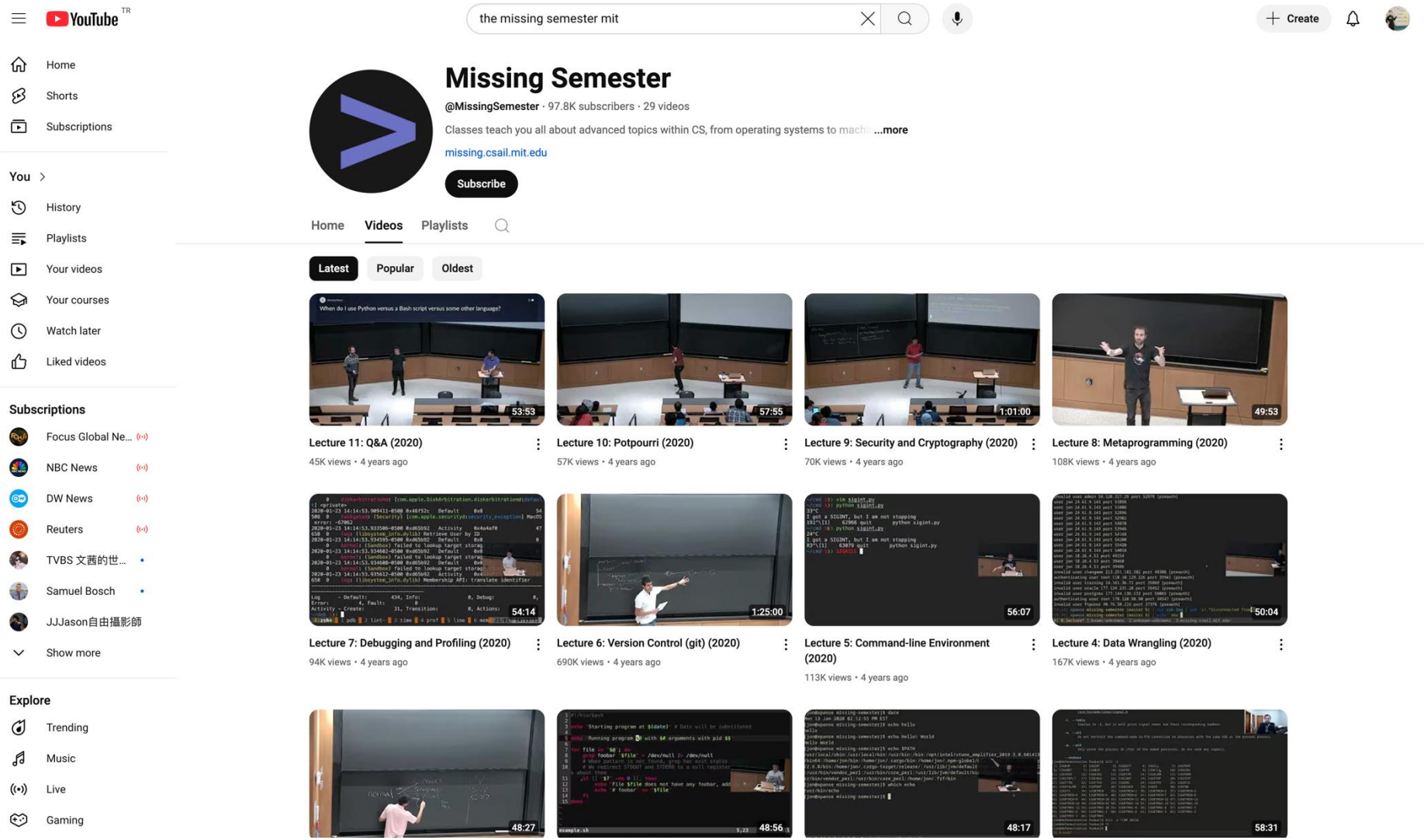
- **Linux Commands Cheat Sheet**

https://www.geeksforgeeks.org/linux-commands-cheat-sheet/

- **The Missing Semester, MIT CSAIL**

https://www.youtube.com/@MissingSemester/videos

The Missing Semester, MIT CSAIL



Thank you very much for your attention!