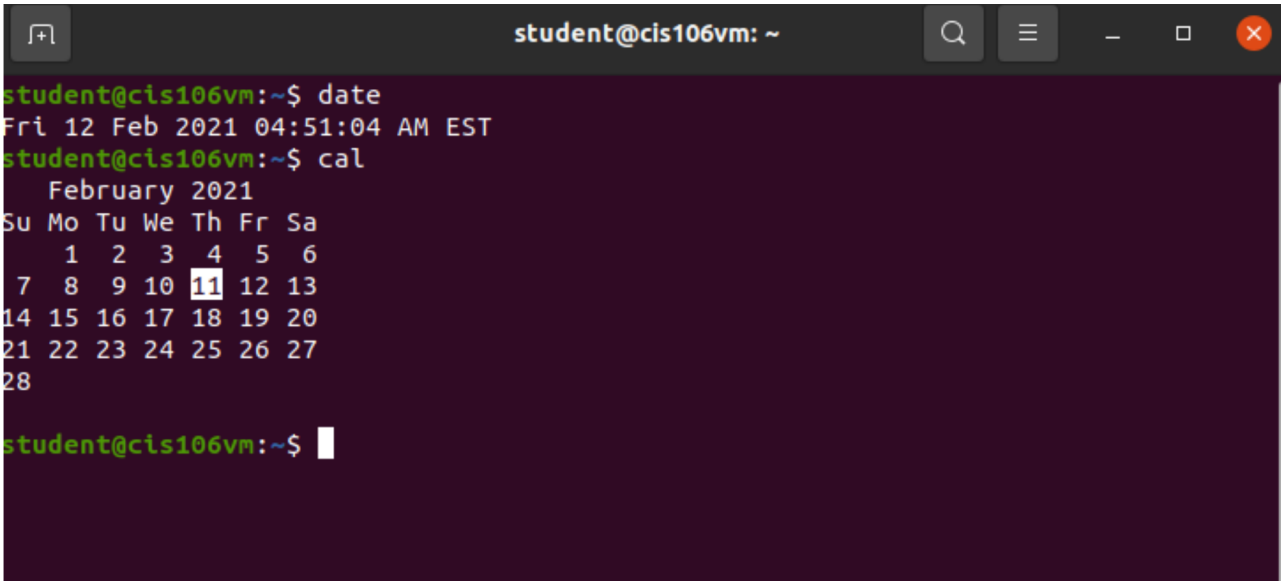
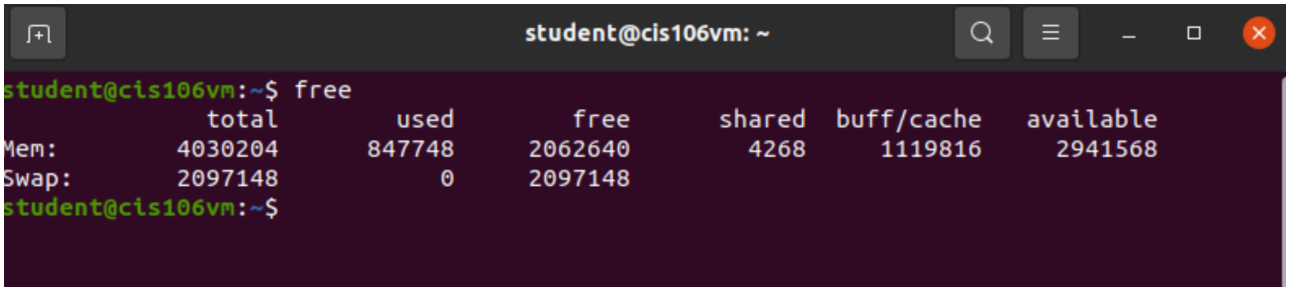
Study Guide For Final Spring 2022

List of Commands

- date Display the current time and date.

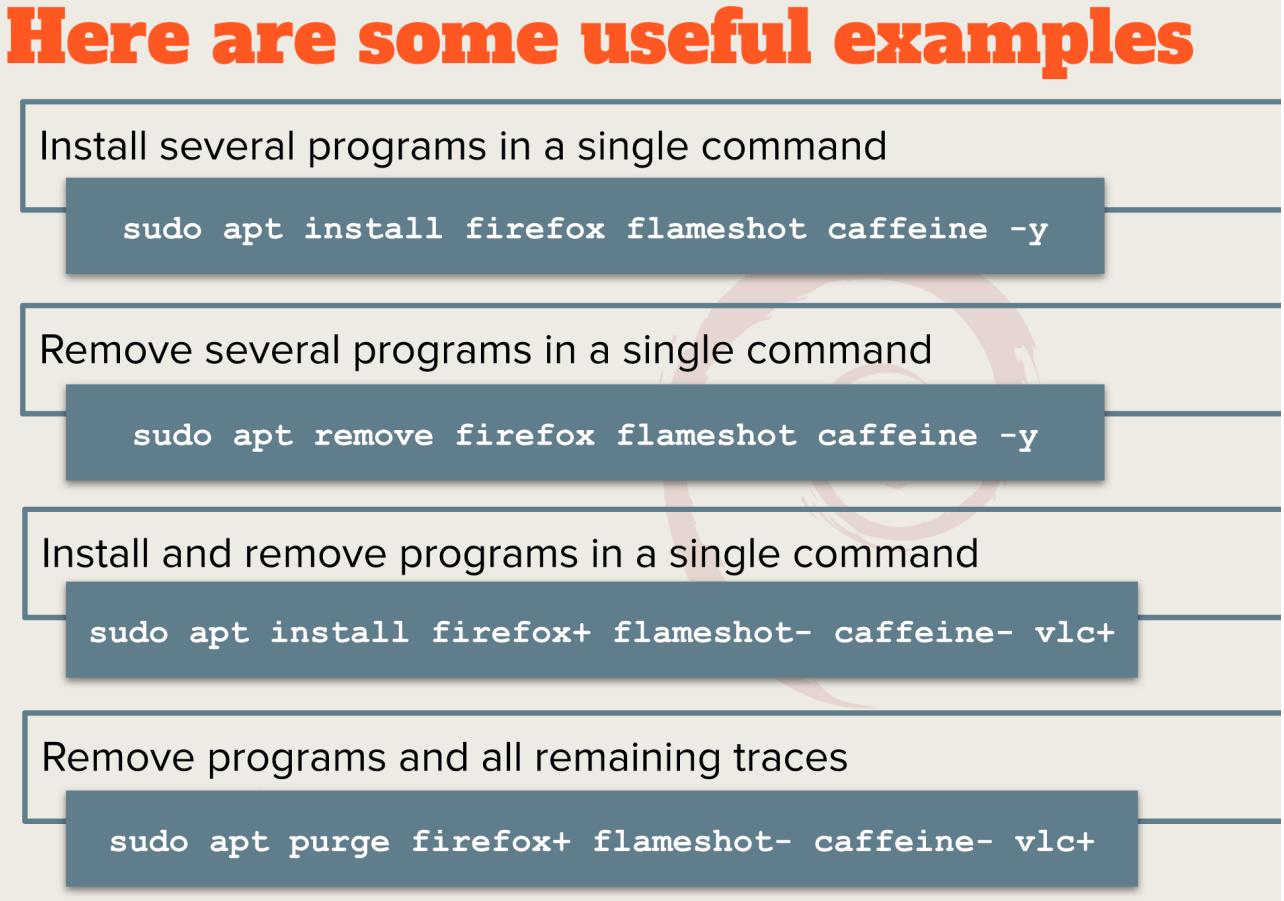


- uname Displays information about your system.
- du
- free Displays the amount of free memory.



echo

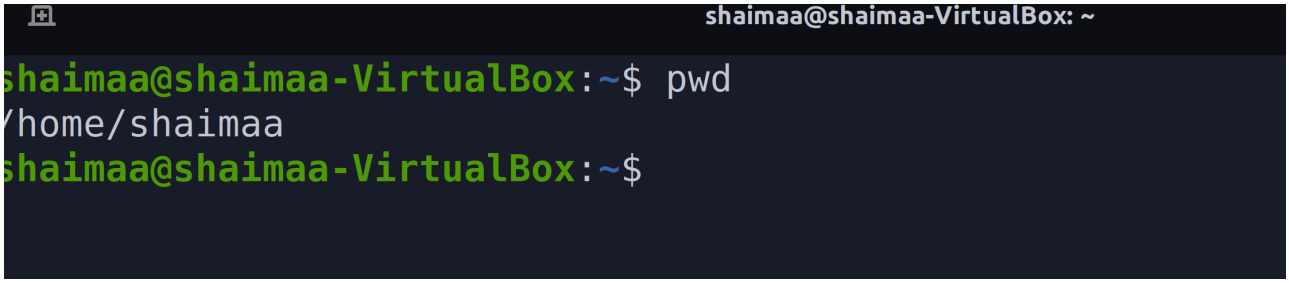
- apt for advanced package tool- is set of tools for managing Debian packages.



apt-get Command	apt Command
apt-get install	apt install
apt-get remove	apt remove
apt-get update	apt update
apt-get upgrade	apt upgrade
apt-get dist-upgrade	apt full-upgrade
apt-get remove --purge	apt purge
apt-get autoremove	apt autoremove
apt-get search	apt search
apt-get show	apt show
dpkg --get-selections	apt list --installed
apt-get purge	apt purge

- pwd Displays the current working directory where you are currently working.

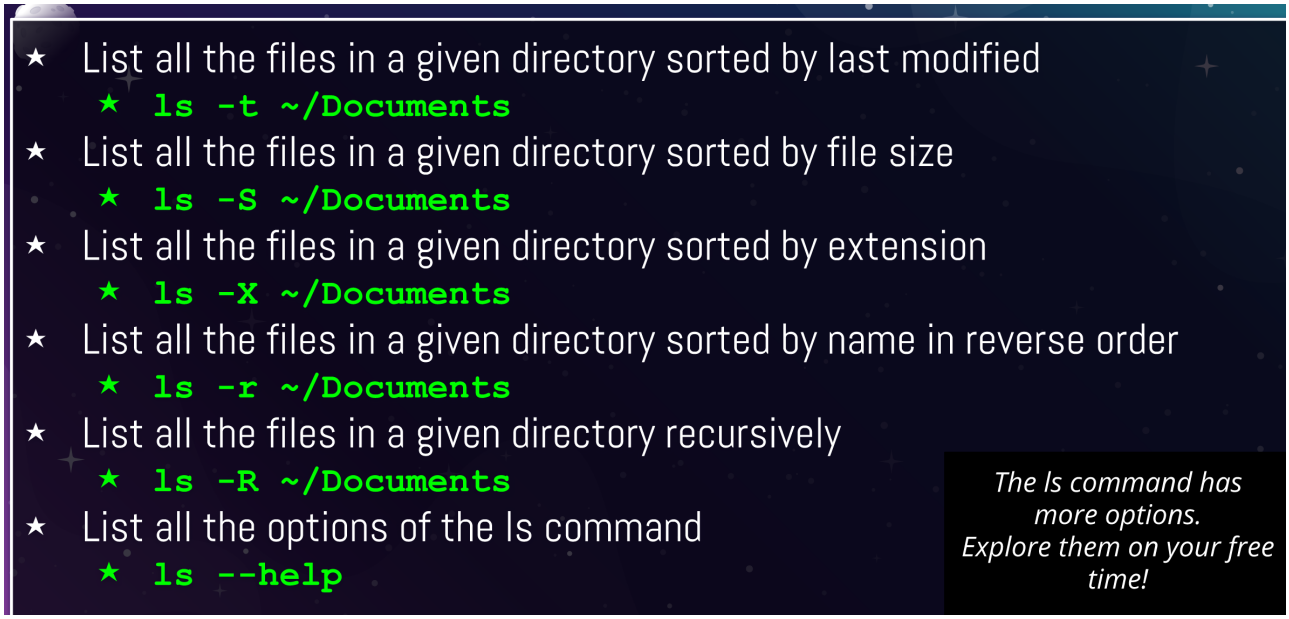
- Example:



- cd

change the current working directory. in other words, moves you around. How to use it : cd + destination

- ls is used for listing the content of a given directory or the file directory itself. Example:



```
1.  wget https://cis106.com/assets/practicels.tar.xz
2.  tar Jxf practicels.tar.xz
3.  cd practicels/
4.  ls
5.  ls -a
6.  ls -l
7.  ls -lh
8.  ls -lht
9.  ls -lhgGS
10. ls -mr
11. ls -1X
```

- tree
- man Man manual pages are documentation files that describe Linux shell commands, executable programs, systems,calls,special files,and so forth. man pages are not step by step guides ,but instead quick reference *Example:

Section	Description	Examples
1	Executable programs or shell commands	man ls, man pwd
2	System calls, which are system requests that programs make to the kernel	man kill, man read
3	Library calls (to access functions in program libraries)	man xcrypt, man stdin
4	Special files, such as the floppy disk, that are usually found in /dev	man fd, man tty
5	File formats and conventions	man passwd, man hosts
6	Games	man tetravex, man AisleRiot
7	Macro packages and conventions	man man (7), man gruff (7)
8	System administration commands	man yast, man suseconfig

Open the man page of the passwd command
- **man passwd**

Open a specific man page for the passwd command
- **man 5 passwd**

Show the man page section of the passwd command
- **man -f passwd**

Show all the available pages of a command
- **man -a passwd**

Searches for a man page for a given word or regular expression or phrase.
- **man -k file**

- mkdir is used for creating a single directory or multiple directories.
- Example:

Examples of the `mkdir` command

- Create a directory in the present working directory
  - `mkdir wallpapers`
- Create a directory in a different directory using relative path
  - `mkdir wallpapers/ocean`
- Create a directory in a different directory using absolute path
  - `mkdir ~/wallpapers/forest`
- Create a directory with a space in the name
  - `mkdir wallpapers/new\ cars`
  - `mkdir wallpapers/'cities usa'`
- Create a directory with a single quote in the name
  - `mkdir wallpapers/"majora's mask"`
- Create multiple directories
  - `mkdir wallpapers/cars wallpapers/cities wallpapers/forest`
- Create a directory with a parent directory at the same time.
  - `mkdir -p wallpapers_others/movies`

- touch is used for creating files
  - To create a file called list
    - `touch list`
  - To create several files:
    - `touch list_of_cars.txt script.py names.csv`
  - To create a file using absolute path:
    - `touch ~/Downloads/games.txt`
  - To create a file using relative path (assuming you pwd is you home directory):
    - `touch Downloads/games2.txt`
  - To create a file with a space in its name:
    - `touch "list of foods.txt"`

- rm remove files rm by default does not removes directories.T o remove a directory use rm with the -r option.

- Example
  Remove a file
  - `rm list`

  Remove a file and prompt confirmation before removal
  - `rm -i list`

  Remove all the files inside a directory and ask before removing more than than 3 files
  - `rm -I Downloads/games/*`

  Remove an empty directory
  - `rmdir Downloads/games`

  Remove an non-empty directory
  - `rm -r Downloads/games`

- cp copies files/directories from source to a destination the cp command uses the same structure as the mv command
  - To copy a file
    - `cp Downloads/wallpapers.zip Pictures/`
  - To copy a directory with absolute path
    - `cp -r ~/Downloads/wallpapers ~/Pictures/`
  - To copy the content of a directory to another directory
    - `cp Downloads/wallpapers/* ~/Pictures/`
  - To copy multiple files in a single command
    - `sudo cp -r script.sh program.py home.html assets/ /var/www/html/`

- mv moves and renames directories.

- Example
  - To move a file from a directory to another using relative path
    - `mv Downloads/homework.pdf Documents/`
  - To move a directory from one directory to another using absolute path
    - `sudo mv ~/Downloads/theme /usr/share/themes`
      - *Notice that in this command I am using sudo since the destination is owned by root.*
  - To move a file from one directory to another combining absolute path and relative path
    - `mv Downloads/english_homework.docx /media/student/flashdrive/`
      - *Notice that in this command I am moving the file "english_homework.docx" to the directory where the flash drive is mounted.*
  - To move multiple directories/files to a different directory
    - `mv games/ wallpapers/ rockmusic/ /media/student/flashdrive/`

- stat a data structure that contains all the information about a file except the file name and content.

- Example stat script.sh
- Wildcards (,?,[]) *the main wildcard is a star ,or asterisk()* character. A star alone matches anything and nothing and matches any number of characters
- Example ls *.txt wil match all files that end in txt regardless of the size of the file name .
  - To match all files that have a vowel after letter f:
    - `ls f[aeiou]*`
  - To match all files that do not have a vowel after letter f:
    - `ls f[!aeiou]*`
  - To match all files that have a range of letters after f:
    - `ls f[a-z]*`
  - To match all files whose name has at least one number:
    - `ls *[0-9]*`
  - To match all the files whose name does not have a number in their file name:
    - `ls *[!0-9].*`
  - To match all files whose name begins with a letter from a-p or start with letters s or c:
    - `ls [a-psc]*`
  - To match all files whose name begins with any of these two sets of characters: letters from a-f or p-z:
    - `ls [a-fp-z]*`
  - To match all files whose name begins with any 3 combination of numbers and the current user's username:
    - `ls [0-9][0-9][0-9]$USER`
- Brace expansion {} is not a wildcard but another feature of bash that allows you to generate rabbitry string to use with commands.
- Example
  - To create a whole directory structure in a single command:
    - `mkdir -p music/{jazz,rock}/{mp3files,vidoes,oggfiles}/new{1..3}`
  - To create a N number of files use:
    - `touch website{1..5}.html`
    - `touch file{A..Z}.txt`
    - `touch file{001..10}.py`
    - `touch file{{a..z},{0..10}}.js`
  - Remove multiple files in a single directory
    - `rm -r {dir1,dir2,dir3,file.txt,file.py}`
- cat the cat command is use for displaying the content of a file . cat is short for concatenate which is the command intended.
- Usage cat + option +file(s) to display

- Example

```
- Display the content of a file with line numbers
    - cat -n ~/Documents/todo.md
- Display the content of a file with line numbers excluding empty lines
    - cat -b ~/Documents/todo.md
- Display the content of a file a $ at the end of every line
    - cat -E ~/Documents/todo.md
- Display the content of a file suppressing repeating empty lines to a single empty line
    - cat -s ~/Documents/todo.md
```

- head the head command displays the top N number of lines of a given file by default it prints the first 10 line If more than one file name is provided then data from each file is proceeded by its file name.

- Example

```
Display the first 10 lines of a file
    ■ head ~/Documents/Book/dracula.txt
Display the first 5 lines of a file
    ■ head -5 ~/Documents/Book/dracula.txt
```

- tail the head command displays the top N number of lines of a given file by default it prints the first 10 line If more than one file name is provided then data from each file is proceeded by its file name.

- Example

```
Display the first 10 lines of a file
    ■ head ~/Documents/Book/dracula.txt
Display the first 5 lines of a file
    ■ head -5 ~/Documents/Book/dracula.txt
```

- cut the / etc/passwd contains one lne for each user account, with seven fields delimited by colons(😃

- Example

```
Display the last 10 lines of a file
    ■   tail ~/Documents/Book/dracula.txt
Display the last 5 lines of a file
    ■   tail -5 ~/Documents/Book/dracula.txt
```

- tr used for translating or deleting or characters from standard output.
- Example

```
    ■   cat file.txt | tr '.' ','
Translate white space into tabs.
    ■   cat program.py | tr "[:space:]" '\t'
Translate tabs into space.
    ■   cat file.py | tr -s "[:space:]" ' '
```

- paste is use for joining files horizontally in columns
- Example

```
    ■   paste users.lst ip_address.lst
Merge two files using a different delimiter
    ■   paste -d ":" users1.lst ip_addresses.lst
```

- wc the wc is used for printing the number of lines characters and bytes in a file.

Display the number of characters in a file

```
wc -m users.txt
```

Display the number of lines in a file

```
wc -l users.txt
```

Display the number words in a file

```
wc -w users.txt
```

- grep is used to search text in given file. grep works line by line basis.
- Usage grep + option search criteria + file(s)

More examples of grep

- Search any line that contains the word 'dracula' regardless of the case
  - `grep -i 'dracula' ~/Documents/Books/dracula.txt`
- Search any line that contains the word dracula regardless of case and with number line
  - `grep -in 'dracula' ~/Documents/Books/dracula.txt`
- Search for all the lines that do not contain the word 'war'
  - `grep -v 'war' ~/Documents/Books/war-and-peace.txt`
- Search and display only the matched string (pattern)
  - `grep -o 'pride' ~/Documents/Books/war-and-peace.txt`
- Search and match only the word
  - `grep -o 'dracula' ~/Documents/Books/dracula.txt`

- output redirection q26

Saving the output of a command command output + > +file

Other examples of |

- Display only the ip addresses from the output of the ip command

```
ip addr | grep -Eo '[[:digit:]]{1,3}\.[[:digit:]]{1,3}\.[[:digit:]]{1,3}\.[[:digit:]]{1,3}'
```

- Display only the 2nd line in a file
  - `head -2 file.lst | tail -1`

vim or nano (basic stuff: open a file, close a file, edit a file)

Editing a file with vim

- You can tell vim that you want to edit another file by using the e command
- **:e** new.txt -> will open new.txt and allow you to edit
- You can use auto completion here
- Ctrl + g will show the file that you are currently editing in the status line
- You can also use :f in command mode to see the file that you are currently working on.

Navigating a file

- In normal mode use the keys:
  - `H = left`
  - `J = down`
  - `K = up`
  - `L = right`

You can prefix the number of times by adding the number after the letter
10H will move 10 character to the left

Saving and quitting vim

- To save a text file you need to enter normal mode using : and the use the w key
  - `:w` will save the file
  - `:w new.txt` will save the file as new.txt
  - `:wq` will save the file and quit
  - `:wqa!` will save the file and close all files open in the buffer

tar

Examples of the tar command

Action	Example
create archive	tar -cf example.tar file1 file2 file3
extract archive	tar -xf example.tar
Extract archive in a different directory	tar -xf example.tar --directory ~/Downloads
extract an specific file	tar -xf example.tar file3
list the contents of an archive	tar -tf example.tar
add files to an archive	tar -rf example.tar file4
update files inside an archive	tar -uf example.tar file4
to add members of an archive to another archive	tar -Af example.tar example2.tar
to delete specific members of an archive	tar --delete -f example.tar file3
to compare files with members of an archive	tar -df example.tar file2

- The option -f is always required.
- The -v option displays the details of the operation. It is not required to use it but it is good practice
- Files inside an archive are called members.

Practice 1 | tar

1. Run the following script: https://robertalberto.com/cis106/mandata.sh
   a. **Use the command:** `curl https://robertalberto.com/cis106/mandata.sh | bash`
2. Change your current working directory to: `~/managingData/practice1`
3. Create an archive called `fortunes.tar` containing all the files in the `practice1` directory.
4. Remove all the files, except the `fortunes.tar` file, inside the `practice1` directory.
5. List all the members of the archive.
6. Create a new file called `newfile.txt` and add some text to it.
7. Add the `newfile.txt` to the archive.
8. Add more text to `newfile.txt` and update the archive.
9. Remove `newfile.txt` from the archive and from `practice1` directory.
10. Create a new directory called `newDir` and extract the archive inside the new directory.

gz, bzip2, or xz chmod