brew install curl
brew install wget

download files using curl
curl --output latest.zip https://wordpress.org/latest.zip
curl -o vue-v2.6.10.js https://cdn.jsdelivr.net/npm/vue/dist/vue.js
curl -o curl-8.13.0.tar.gz https://curl.se/download/curl-8.13.0.tar.gz

extract tar.gz file
tar -xzvf archive.tar.gz
tar -xzvf archive.tar.gz -C /target/directory

list content without extracting
tar -tvf archive.tar.gz

download files using wget
`wget https://wordpress.org/latest.zip`
wget https://github.com/ranaroussi/yfinance/blob/main/README.md
wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.17.2.tar.xz
wget -O latest-hugo.zip https://github.com/gohugoio/hugo/archive/master.zip
wget -P /home/tsiameh/Desktop http://mirrors.mit.edu/centos/7/isos/x86_64/CentOS-7-x86_64-Minimal-1804.iso
wget -b https://download.opensuse.org/tumbleweed/iso/openSUSE-Tumbleweed-DVD-x86_64-Current.iso
wget -q --show-progress https://code.jquery.com/jquery-3.6.0.min.js
`wget -O wordpress-install.zip https://wordpress.org/latest.zip`

create a zip file
zip archive.zip file1.txt file2.txt

zip a directory recursively
zip -r archive.zip /path/to/directory

 # Create a password-protected zip

```
zip -e secure.zip file.txt
```

 # Update an existing zip file

```
zip -u archive.zip newfile.txt
```

 # Extract to a specific directory

```
unzip archive.zip -d /target/directory
```

 # List contents without extracting

```
unzip -l archive.zip
```

Login into remote server
brew install openssh
ssh ts75230@99.48.1.100 -p 22

copy files from local to remote server
```
scp file.txt user@host:/home/ts75080/Desktop -P 22
```

download files from remote server to local
```
scp -P 22 user@host:/path .
scp -P 22 user@host:/path /user/tsiameh/Desktop/PythonCourse
```

Sync directories
```
rsync -avz /local/dir username@server-ip:/remote/dir
```

SED and AWK Usage

Basic Text Replacement
```
# Replace "apple" with "orange" in file.txt
sed 's/apple/orange/g' file.txt
```

Case-Insensitive Replacement
```
# Replace "hello" (any case) with "Hi"
sed 's/hello/Hi/gI' file.txt
```

Example Input:
```
Hello world, hello everyone
```
Output:
```
Hi world, Hi everyone
```

Print Specific Columns
```
# Print 1st and 3rd columns of data.csv (comma-delimited)
awk -F',' '{print $1, $3}' data.csv

awk -F'\t' '{print $1,$3}' data.tsv  (tab-delimited)

awk -F',' '$2 > 25 {print $1,$3}' data.csv  # Only people over 25

awk -F',' '$2 > 25 {print $0}' data.csv | sed 's/Alice/Trump/g'
```

Example Input (`data.csv`):
```
John,25,USA
Alice,30,Canada
```

Output:

```
John USA
Alice Canada
```

Combining `sed` and `awk`

```
# Extract process IDs (PID) from `ps aux`, then replace "python" with "PY"
ps aux | awk '{print $2}' | sed 's/python/PY/g'
```

`chmod` (Change File Permissions) Examples

Basic Permission Assignment

```
# Give owner full permissions (7), group read/execute (5), others read/execute (5)
chmod 755 script.sh
```

Result:

```
-rwxr-xr-x script.sh
```

(Owner: read/write/execute, Group/Others: read/execute)

Symbolic Permissions (Human-Readable)

```
# Add execute permission for all users
chmod a+x script.sh

# Remove write permission from others
chmod o-w secret.txt

# Set owner=rwx, group=rx, others=rx (same as 755)
chmod u=rwx,g=rx,o=rx script.sh
```

Recursive Permission Change

```
# Apply 755 to all files/directories in /var/www
chmod -R 755 /var/www
```

`chown` (Change File Owner/Group) Examples

Basic Ownership Change

```
# Change owner to 'nginx' and group to 'www-data'
chown nginx:www-data /var/www/html
```

Before:

```
-rw-r--r-- 1 root root 1204 Jun 10 index.html
```

After:

```
-rw-r--r-- 1 nginx www-data 1204 Jun 10 index.html
```

Recursive Ownership

```
# Change owner/group for all files in a directory
chown -R deploy:deploy /opt/myapp
```

Change Only Owner or Only Group

```
# Change owner only (keep group)
chown jenkins /opt/backups

# Change group only (keep owner)
chown :developers /src/code
```

Numeric to `rwx` Permission Mapping

Number	Binary	Permission	`rwx` Equivalent
0	000	No access	`---`
1	001	Execute	`--x`
2	010	Write	`-w-`
3	011	Write + Execute	`-wx`
4	100	Read	`r--`
5	101	Read + Execute	`r-x`
6	110	Read + Write	`rw-`
7	111	Read + Write + Execute	`rwx`

Common Permission Combinations

Numeric	`rwx` Form	Typical Use Case
777	`rwxrwxrwx`	Dangerous! Global read/write/execute (temporary directories only)
755	`rwxr-xr-x`	Scripts/executables (owner: full, others: read/execute)
644	`rw-r--r--`	Config files (owner: read/write, others: read-only)
600	`rw-------`	Private files (owner only, no group/others access)
750	`rwxr-x---`	Group-shared scripts (others: no access)
1777	`rwxrwxrwt`	Sticky bit (e.g., `/tmp` – anyone can add files, but only owners can delete thei

How to Calculate Numeric Permissions

1. **Convert `rwx` to binary**:
   - `r` (read) = **4**
   - `w` (write) = **2**
   - `x` (execute) = **1**
   - `-` (no permission) = **0**
2. **Sum the values for each group**:

   Example: `rwxr-xr--`
   - Owner (`rwx`): 4 (r) + 2 (w) + 1 (x) = **7**
   - Group (`r-x`): 4 (r) + 0 + 1 (x) = **5**
   - Others (`r--`): 4 (r) + 0 + 0 = **4**

     → **754**

Table of Contents

Introduction to Bash

Bash (Bourne Again SHell) is a Unix shell and command language. It's widely used for scripting to automate tasks on Linux/Unix systems.

Key features:

- Command execution
- Scripting capabilities
- Variable handling
- Flow control structures

Creating and Running Scripts

1. Create a file with `.sh` extension
2. Add shebang at top: `#!/bin/bash`
3. Make it executable: `chmod +x script.sh`
4. Run it: `./script.sh`

Command	Description	Example
`ls`	List files	`ls -l`
`cd`	Change directory	`cd /home`
`pwd`	Print working directory	`pwd`
`cp`	Copy files	`cp file.txt backup/`
`mv`	Move/rename files	`mv old.txt new.txt`
`rm`	Remove files	`rm file.txt`
`mkdir`	Create directory	`mkdir new_folder`
`rmdir`	Remove empty directory	`rmdir empty_dir`
`touch`	Create empty file	`touch newfile.txt`
`find`	Search for files	`find /home -name "*.txt"`

File Viewing/Editing

`cat`	Display file content	`cat file.txt`
`less`	View file page by page	`less large.log`
`head`	Show first lines	`head -n 5 file.txt`
`tail`	Show last lines	`tail -f log.txt`
`grep`	Search text	`grep "error" log.txt`
`sed`	Stream editor	`sed 's/old/new/g' file.txt`
`awk`	Text processing	`awk '{print $1}' file.txt`
`nano`	Text editor	`nano file.txt`
`vim`	Advanced editor	`vim file.txt`
`diff`	Compare files	`diff file1.txt file2.txt`

System Information

`uname`	System info	`uname -a`
`top`	Process viewer	`top`
`htop`	Interactive process viewer	`htop`
`ps`	Process status	`ps aux`
`free`	Memory usage	`free -h`
`df`	Disk space	`df -h`
`du`	Directory size	`du -sh *`
`uptime`	System uptime	`uptime`
`whoami`	Current user	`whoami`
`history`	Command history	`history`

Networking

`ping`	Test connection	`ping google.com`
`wget`	Download files	`wget https://example.com/file.zip`
`curl`	Transfer data	`curl -O https://example.com/file.zip`
`ssh`	Remote login	`ssh user@host`
`scp`	Secure copy	`scp file.txt user@host:/path`
`ifconfig`	Network interfaces	`ifconfig`
`netstat`	Network stats	`netstat -tulnp`
`dig`	DNS lookup	`dig example.com`
`traceroute`	Network path	`traceroute google.com`
`hostname`	System hostname	`hostname`

Permissions & Users

`chmod`	Change permissions	`chmod 755 script.sh`
`chown`	Change owner	`chown user:group file.txt`
`sudo`	Run as superuser	`sudo apt update`
`su`	Switch user	`su - username`
`passwd`	Change password	`passwd`
`useradd`	Add user	`sudo useradd newuser`
`usermod`	Modify user	`sudo usermod -aG sudo user`
`groupadd`	Add group	`sudo groupadd newgroup`
`id`	User identity	`id`
`who`	Show logged-in users	`who`

 # GITHUB

git clone https://github.com/ranaroussi/yfinance.git

50 common Bash commands and their real-world applications:

- `ls`: List directory contents - Used to view files and subdirectories in a directory.
- `pwd`: Print working directory - Displays the current directory path.
- `cd`: Change directory - Navigates to a specified directory.
- `mkdir`: Make directory - Creates a new directory.
- `rmdir`: Remove directory - Deletes an empty directory.
- `rm`: Remove files or directories - Deletes files or directories (use with caution).
- `touch`: Create a file - Creates an empty file or updates the timestamp of an existing one.
- `cp`: Copy files or directories - Copies files or directories to a new location.
- `mv`: Move or rename files or directories - Moves or renames files or directories.
- `cat`: Concatenate and display files - Displays the content of a file.
- `echo`: Display a line of text - Prints text to the terminal.
- `less`: View file contents page by page - Allows scrolling through a file's content.
- `head`: Display the beginning of a file - Shows the first few lines of a file.
- `tail`: Display the end of a file - Shows the last few lines of a file.
- `grep`: Search for a pattern in files - Searches for specific text within files.
- `find`: Search for files and directories - Locates files and directories based on criteria.
- `man`: Display command manual pages - Shows the manual for a specific command.
- `chmod`: Change file permissions - Modifies file access permissions.
- `chown`: Change file owner and group - Changes the owner and group of a file.
- `tar`: Archive files - Compresses and extracts files from archives.
- `gzip`: Compress files - Compresses files using gzip compression.
- `gunzip`: Decompress files - Decompresses files compressed with gzip.
- `zip`: Package and compress files - Creates a zip archive.
- `unzip`: Extract files from a zip archive - Extracts files from a zip archive.
- `diff`: Compare files - Shows the differences between two files.
- `cmp`: Compare two files byte by byte - Checks if two files are identical.
- `sort`: Sort lines in a file - Sorts lines of text in a file.
- `uniq`: Remove duplicate lines - Removes duplicate lines from a file or input.
- `cut`: Extract sections from lines - Extracts specific parts of lines.
- `paste`: Merge lines of files - Merges lines from multiple files.
- `wc`: Count words, lines, characters - Counts words, lines, and characters in a file.
- `history`: Display command history - Shows previously executed commands.
- `alias`: Create command aliases - Creates shortcuts for commands.

34abe. `unalias`: Remove command aliases - Removes command aliases.

- `ps`: Display running processes - Shows running processes.
- `top`: Display real-time processes - Shows real-time information about running processes.
- `kill`: Terminate processes - Terminates a running process.

- `df`: Display disk space usage - Shows disk space usage.
- `du`: Display directory space usage - Shows disk space usage for directories.
- `free`: Display memory usage - Shows memory usage.
- `uname`: Print system information - Displays system information.
- `uptime`: Show system uptime - Shows how long the system has been running.
- `whoami`: Display current user - Shows the current user.
- `sudo`: Execute a command as another user - Runs a command as the superuser or another user.
- `apt-get` or `yum`: Package management (Debian/Red Hat) - Installs, updates, and removes software packages.
- `ssh`: Secure Shell - Connects to a remote server securely.
- `scp`: Secure copy - Copies files securely between systems.
- `ping`: Check network connectivity - Tests network connectivity to a host.
- `ifconfig` or `ip`: Configure network interfaces - Configures network interfaces.
- `date`: Display or set date and time - Shows or sets the system date and time.

- Cal: display calendar

Top 50 Linux Commands You Must Know as a Regular User

1. **ls** - The most frequently used command in Linux to list directories
2. **pwd** - Print working directory command in Linux
3. **cd** - Linux command to navigate through directories
4. **mkdir** - Command used to create directories in Linux
5. **mv** - Move or rename files in Linux
6. **cp** - Similar usage as mv but for copying files in Linux
7. **rm** - Delete files or directories
8. **touch** - Create blank/empty files
9. **ln** - Create symbolic links (shortcuts) to other files
10. **clear** - Clear the terminal display
11. **cat** - Display file contents on the terminal
12. **echo** - Print any text that follows the command
13. **less** - Linux command to display paged outputs in the terminal
14. **man** - Access manual pages for all Linux commands
15. **uname** - Linux command to get basic information about the OS
16. **whoami** - Get the active username
17. **tar** - Command to extract and compress files in linux
18. **grep** - Search for a string within an output
19. **head** - Return the specified number of lines from the top
20. **tail** - Return the specified number of lines from the bottom
21. **diff** - Find the difference between two files

22. **cmp** - Allows you to check if two files are identical
23. **comm** - Combines the functionality of diff and cmp
24. **sort** - Linux command to sort the content of a file while outputting
25. **export** - Export environment variables in Linux
26. **zip** - Zip files in Linux
27. **unzip** - Unzip files in Linux
28. **ssh** - Secure Shell command in Linux
29. **service** - Linux command to start and stop services
30. **ps** - Display active processes
31. **kill and killall** - Kill active processes by process ID or name
32. **df** - Display disk filesystem information
33. **mount** - Mount file systems in Linux
34. **chmod** - Command to change file permissions
35. **chown** - Command for granting ownership of files or folders
36. **ifconfig** - Display network interfaces and IP addresses
37. **traceroute** - Trace all the network hops to reach the destination
38. **wget** - Direct download files from the internet
39. **ufw** - Firewall command
40. **iptables** - Base firewall for all other firewall utilities to interface with
41. **apt, pacman, yum, rpm** - Package managers depending on the distribution
42. **sudo** - Command to escalate privileges in Linux
43. **cal** - View a command-line calendar
44. **alias** - Create custom shortcuts for your regularly used commands
45. **dd** - Majorly used for creating bootable USB sticks
46. **whereis** - Locate the binary, source, and manual pages for a command
47. **whatis** - Find what a command is used for
48. **top** - View active processes live with their system usage
49. **useradd and usermod** - Add a new user or change existing user data
50. **passwd** - Create or update passwords for existing users

https://www.digitalocean.com/community/tutorials/linux-commands
https://www.youtube.com/watch?v=ZtqBQ68cfJc