SOME COOL PTS LEARNT FROM LABS

- Xargs: xargs reads items from the standard input, delimited by blanks. Especially useful with find command.
- For chmod, the permissions format is
  First digit refers to user, second to group, third to other. And each digit is as per bitcode, which has 3 bits corresponding to read, write, execute(in that order). Bit is 1 if that permission is given and 0 otherwise.

- If something is supposed to be line start and something is supposed to be line end, make sure to explicitly put ^ and $ in regex. Because it is possible that special characters are possible that you are not able to see and you end up breaking your brain trying to figure out what the error is.

- In grep for writing normal style regex, -E tag can be especially useful(written in man page but still too useful so writing here). Also using -o tag you can make it so that the resulting file does not have the entire line and rather just the matching part is printed.

- For sort you may in general have some table type thing or something. So in that, for sorting, -t tag and -k tag are especially useful. Remember them because the man page for them is very confusing and not written well. With -t tag you specify the separator of fields(like the delimiter in cut command) and with -k tag you specify based on what you want to sort. Like if you want to sort based on the second column write -k2. Now an even crazier thing, say in the second column also you want to sort based on the digits from 3rd digit onwards, then you can write -k2.3.
  With a comma you can also put the ending point of comparison here. If you don't put it, then it is taken as end of line by default.
- **Find -name command uses wildcard not regex. Imp point about wildcard: Here * means match anything while in regex it matches 0 or more of the character before it. But both are fine with using [abc], [0-9] etc.**

NOTES

- which command: The `which` command in Linux is used to locate the executable file associated with a given command. When you enter a command in the terminal, `which` helps identify which executable file will be executed when that command is invoked.
  -a option: Display all matches found, instead of just the first.

- Unix/Linux assigns **integer IDs** (file descriptors) to standard input/output/error:

  - `0` **(stdin)** → Standard Input (keyboard by default)
  - 1 **(stdout)** → Standard Output (terminal by default)
  - 2 **(stderr)** → Standard Error (error messages go here)
  - EXAMPLES:
  - ls nonexistent_file 2> error.log # Redirects stderr to a file
  - ls file1 file2 > output.log 2>&1 # Redirects stdout & stderr to output.log (`2>&1` means **redirect stderr (2) to where stdout (1) is going**.)
  - The **null device** (`/dev/null`) is used to suppress output.
- Within `/home/`, each user has their **own home directory**.
- cd: "/" at start means starting from the root directory. "~" at start means starting from my home directory.
  - option: switches you to the previous(as in one in which you were before present one) directory.
- ls:
  - Helps in checking file details like permissions or file size.
  - -l option: Shows detailed info. Permissions, number of links, owner, group, size, modification date.
  - -lh option: Displays file size is human readable format.
  - -lt option: Sorts output by time of last modification. Newest first.
  - -a option: shows hidden files too.
  - -R option: lists subdirectories recursively
  - -S option: sort by file size largest first
  - -X option: sort alphabetically by entry extension
- mkdir:
  - If the directory already exists and if you don't use -p, mkdir will return an error.
  - -p option: Creates directory only if it doesn't exist and creates parent directories also as needed(which doesn't happen with this option)
  - -v option: v stands for verbose i.e. it displays a msg for each directory that is created.
- vi/vim:
  - Save and exit: :wq
  - Exit without saving: :q!
  - Save without exiting: :w
  - Move to start of line: 0

    - Move to end of line: $
    - Move to start of file: gg
    - Move to end of file: G
    - Delete a line: dd
    - Copy a line: yy
    - Paste: p
    - Undo: u
    - Redo: Ctrl+r
    - Searching forward: /term
    - Searching Backward: ?term
    - Repeat search: n(for next occurence)
- nano: Text editor on unix based systems that is more user friendly. Keyboard shortcuts displayed at the bottom of the screen, making it more accessible for beginners. nano -c filename enables line numbers.
- gedit: Default text editor for GNOME desktop environment. GUI based editor.

See remaining commands of the basic part in ma'am's notes.

    - wc: with -c you get byte-count and with -m you get character count.
    -

- `$HOME` is an **environment variable** that stores the **absolute path** of the home directory of the logged-in user.
- **username** is stored in `$USER`. **current directory** is stored in `$PWD`. **shell type** is stored in `$SHELL`.

- `ls -l` lists files **with details** (permissions, owner, size, etc.). In this output, **directories** start with `"d"` in the **first column** (e.g., `drwxr-xr-x`). `grep "^d"` filters only lines **starting with "d"**, which means **directories**.
-