CL-2006 Operating System

LAB - 02

Introduction to Linux basic commands

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

Fall 2026

Objective

This lab is all about getting familiar with Linux operating system by running commands on Ubuntu terminal.

Contents:

Contents

Introduction:

Linux is an open source operating system (OS). An operating system is the software that directly manages a system's hardware and resources, like CPU, memory, and storage. The OS sits between applications and hardware and makes the connections between all of your software and the physical resources that do the work.

Linux Vs Windows:

Windows is widely used operating system while Linux has its own advantages. Following are the advantages discussed in the table below.

LINUX	WINDOWS
Linux is an open source operating system i.e. user can change source code as per requirement	Windows OS is a commercial operating system and its closed source i.e. its source code is inaccessible.
Linux has Monolithic kernel i.e. Whole operating system works in the kernel space.	Windows has a hybrid kernel i.e. Combination of microkernel and monolithic kernel.
File names are case-sensitive in Linux.	File names are not case-sensitive in Windows.
Booting can be done from any disk.	Booting can only be done from the prime disk.
Linux is highly reliable and secure. It has a deep- rooted emphasis on process management, system security, and uptime.	Windows is less reliable than Linux. Over the recent years, Windows reliability has been improved a lot. However, it still has some system instabilities and security weaknesses because of its oversimplified design.

Linux Components:

Major components of Linux are:

- Kernel
- System user space
- Applications

Kernel:

Kernel is the base component of the OS. Without it, the OS doesn't work. The kernel manages the system's resources and communicates with the hardware. It's responsible for memory, process, and file management.

System user space:

System user space is the administrative layer for system-level tasks like configuration and software installation. This includes the shell or command line, processes that run in the background, and the desktop environment.

Applications:

A type of software that lets you perform a task. Applications include everything from desktop tools and programming languages to multiuser business suites. Most Linux distributions offer a central database to search for and download additional apps.

2.1 Commands

A command is a request from a programmer, an operator, or a user to Linux operating system asking that a specific function be performed. For e.g., a request to list all files in your current directory will be the command **ls**.

2.1.1 Syntax of Commands

The general way commands are entered in Linux is as such:

```
command -option(s) argument(s)
```

Here,

- A **command** tells the operating system what to do (what action to be performed, copy a file, display a date etc.)
- **Option(s)** tells the way of action to be performed. For example, ls command displays directory contents, and –r option tells the way in which the directory should be displayed. Here –r displays directory contents in reverse (alphabetically) order.
- **Argument** tells that on what objects (file, directory, devices, etc.) the command and its arguments are applied. For example if we need to display all files starting with alphabet a, you will give "ls a*" and press enter.

Note: Make sure you don't forget that there is always a space between the command, the options, and the arguments.

2.2 More about Shell

2.2.1 The Asterisk *

The asterisk (*) symbol is a wildcard that can be used in various contexts.

- It can denote *everything*. For example, in Linux, typing rm * will delete all files in the current directory.
- It can be used as a filter. For instance, typing ls ab* will list all files and folders that start with ab.

2.2.2 Case Sensitivity

Linux commands are case-sensitive. All standard Linux commands are in lowercase letters. For example:

- Typing ls will list the directory contents.
- Typing LS, lS, or Ls will result in a command syntax error.

2.2.3 Auto-Completion

Auto-Completion is a shortcut feature to quickly enter long commands or ones you may not fully remember. To practice:

- Type a key letter, e.g., f, and press TAB. You will see a list of commands starting with f.
- Add more letters, e.g., fd, and press TAB again. The list narrows down.

You can also use auto-completion for directories. For example, if you want to access the home directory of a user named abcdefghijklmnopqrstuvwxyz:

- From the root directory, type cd /home/a and press TAB. The rest of the name will auto-complete, saving you time.

2.2.4 Redirection

Redirection uses > and < symbols to capture output or input. The types of redirection are:

- > Redirect output to a file.
- 1> Same as >.
- 2> Redirect error output to a file.
- < Redirect input from a file.

Example Commands:

```
cd
ls

touch newfile
ls

ls > newfile
cat < newfile
ls -lh

ls -lh 1> newfile
cat < newfile
lsot

lsot 2> newfile
cat < newfile
rm newfile
```

2.3 Basic Commands for Files & Directory:

2.3.1 pwd command

Use the pwd command to find the path of your current working directory. Simply entering pwd will return the full current path – a path of all the directories that starts with a forward slash (/). For example, /home/username.

The pwd command uses the following syntax:

pwd [option]

It has two acceptable options:

-L or –logical prints environment variable content, including symbolic links.

-P or –physical prints the actual path of the current directory.

2.3.2 cd command

To navigate through the Linux files and directories, use the cd command. Depending on your current working directory, it requires either the full path or the directory name.

Running this command without an option will take you to the home folder. Keep in mind that only users with sudo privileges can execute it.

Let's say you're in /home/username/Documents and want to go to Photos, a subdirectory of Documents. To do so, enter the following command:

cd Photos

If you want to switch to a completely new directory, for example, /home/username/Movies, you have to
enter cd followed by the directory's absolute path:

cd /home/username/Movies

Here are some shortcuts to help you navigate:

cd ~[username] goes to another user's home directory.

cd .. moves one directory up.

cd- moves to your previous directory.

2.3.3 ls command

The ls command lists files and directories within a system. Running it without a flag or parameter will
show the current working directory's content

To see other directories' content, type ls followed by the desired path. For example, to view files in the Documents folder, enter:

ls /home/username/Documents

Here are some options you can use with the ls command:

ls -R lists all the files in the subdirectories.

ls -a shows hidden files in addition to the visible ones.

ls -lh shows the file sizes in easily readable formats, such as MB, GB, and TB.

2.3.4 mkdir command

Use the mkdir command to create one or multiple directories at once and set permissions

for each of them. The user executing this command must have the privilege to make a new folder in the parent directory, or they may receive a permission denied error.

Here's the basic syntax:

mkdir [option] directory_name

For example, you want to create a directory called Music:

mkdir Music

To make a new directory called Songs inside Music, use this command:

mkdir Music/Songs

The mkdir command accepts many options, such as:

-p or –parents create a directory between two existing folders. For example, mkdir -p Music/2020/Songs will make the new "2020" directory.

-m sets the file permissions. For instance, to create a directory with full read, write, and execute permissions for all users, enter mkdir -m777 directory_name.

-v prints a message for each created directory.

2.3.5 rmdir command

To permanently delete an empty directory, use the rmdir command. Remember that the user running this command should have sudo privileges in the parent directory.
For example, you want to remove an empty subdirectory named personal1 and its main folder mydir:

rmdir -p mydir/personal1

2.3.6 cp command

Use the cp command to copy files or directories and their content. Take a look at the following use cases.

- To copy one file from the current directory to another, enter cp followed by the file name and the destination directory. For example:

cp filename.txt /home/username/Documents

- To copy files to a directory, enter the file names followed by the destination directory:

cp filename1.txt filename2.txt filename3.txt /home/username/Documents

- To copy the content of a file to a new file in the same directory, enter cp followed by the source file and the destination file:

cp filename1.txt filename2.txt

- To copy an entire directory, pass the -R flag before typing the source directory, followed by the destination directory:

cp -R /home/username/Documents /home/username/Documents_backup

2.3.7 cat command

Concatenate, or cat, is one of the most frequently used Linux commands. It lists, combines, and writes file content to the standard output. To run the cat command, type cat followed by the file name and its extension. For instance:

cat filename.txt.

Here are other ways to use the cat command:

cat > filename.txt creates a new file.

cat filename1.txt filename2.txt > filename3.txt merges filename1.txt and filename2.txt and stores the output in filename3.txt.

2.3.8 mv command

The primary use of the mv command is to move and rename files and directories. Additionally, it
doesn't produce an output upon execution.

Simply type mv followed by the filename and the destination directory. For example, you want to move filename.txt to the /home/username/Documents directory:

mv filename.txt /home/username/Documents.

You can also use the mv command to rename a file:

mv old_filename.txt new_filename.txt

2.3.9 rm command

The rm command is used to delete files within a directory. Make sure that the user performing this command has write permissions.

Remember the directory's location as this will remove the file(s) and you can't undo it.

Here's the general syntax:

rm filename

To remove multiple files, enter the following command:

rm filename1 filename2 filename3

Here are some acceptable options you can add:

-i prompts system confirmation before deleting a file.

-f allows the system to remove without a confirmation.

-r deletes files and directories recursively.

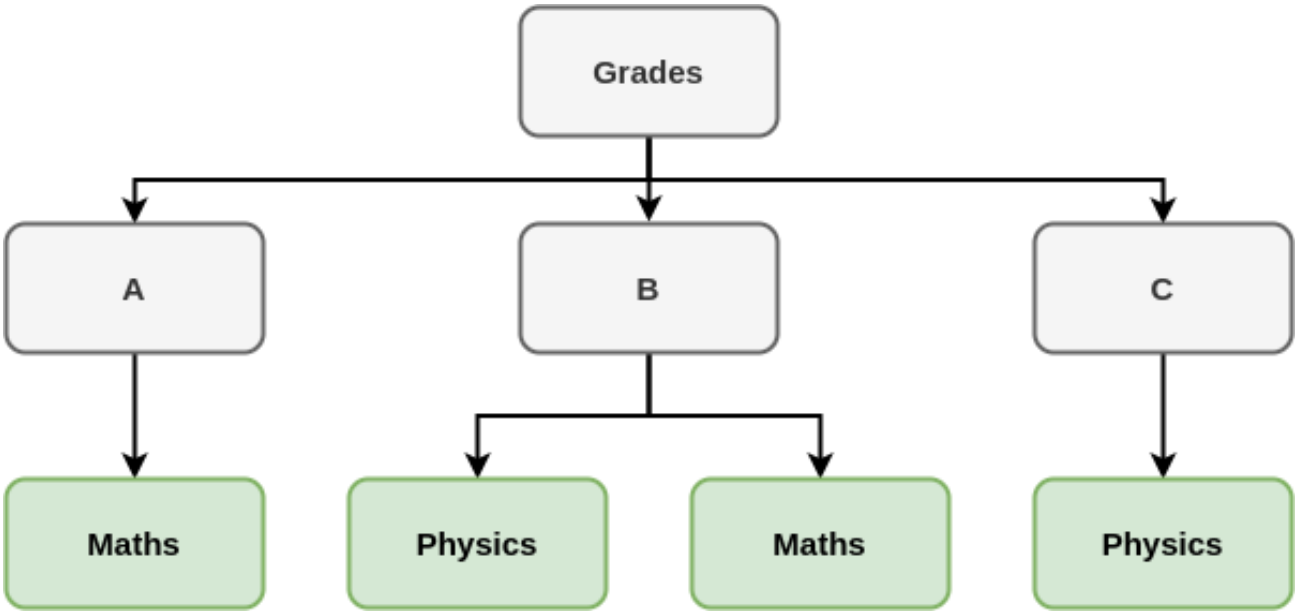
Example Commands:

```
# mkdir
temporary # cd
temporary
temporary# ls

temporary# cat > newfile

Type any text and press
CTRL+D temporary# cat
newfile temporary# mkdir
another

temporary# cp newfile another/newest
temporary# cp newfile newester
temporary# cd another

another# ls
another# ls -a
another# ls –l
another# ls -lh

another# cp newest newestest
another# cat newestest
another#

another# cd ..

temporary# mv newester another/newester
temporary# ls

temporary# ls another/n*
temporary# cd ..

# rm temporary

# rm
temporary/* # rm
temporary

# rm temporary -r -f
```

Easy? Okay, try and attempt the following exercise.

2.3.10 Exercise 1

Implement the following directory tree.

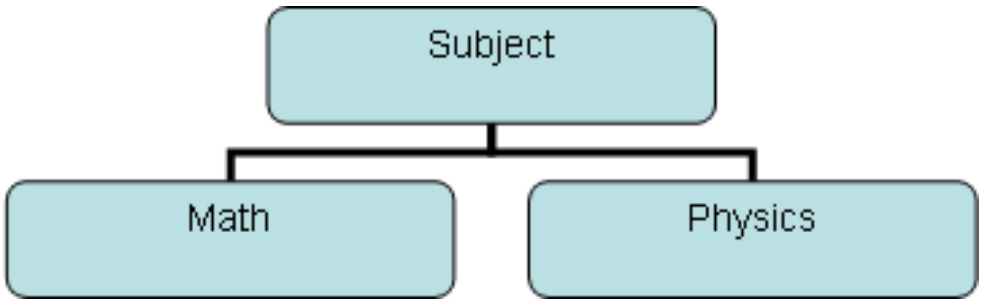


The boxes in white are directories. The boxes in light green are files. Each file should contain a random mark of your liking. Once done, delete all the files/directories that you have created.

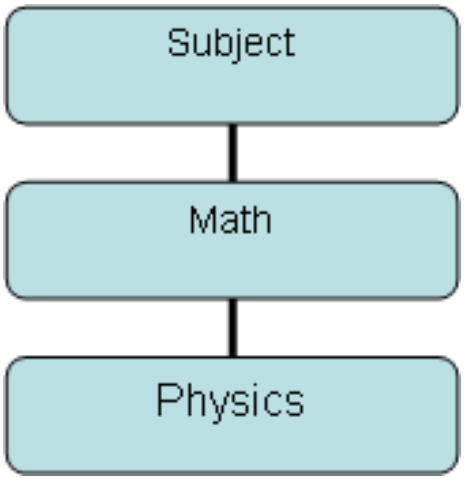
Count the total number of commands you entered to do this job. I managed using 6 commands.

2.3.11 Exercise 2

Consider the following directory tree:



- How are we going to change it using the "mv" command so that we get the directory tree as below:



- Change your directory so that your current directory is “Physics”. From here, change your directory in only one command such that your current directory becomes “subject”.
- From “subject”, issue only one command such that the directory “physics” is deleted.

2.4 Terminal Related Commands:

Following are the terminal related commands:

1. man
2. clear
3. exit
4. history
5. echo
6. alias, unalias
7. |
8. &&

2.4.1 man command/Manual

The man command provides a user manual of any commands or utilities you can run in Terminal, including the name, description, and options.
It consists of nine sections:

1. Executable programs or shell commands
2. System calls
3. Library calls
4. Games
5. Special files
6. File formats and conventions
7. System administration commands
8. Kernel routines
9. Miscellaneous

- To display the complete manual, enter:

man [command_name]

- For example, you want to access the manual for the ls command:

man ls

- Enter this command if you want to specify the displayed section:

man [option] [section_number] [command_name]

- For instance, you want to see section 2 of the ls command manual

man 2 ls

2.4.2 clear command

Enter the **clear** command to clean the Terminal screen.

2.4.3 exit command

Enter the **exit** command to exit the terminal.

2.4.4 history command

With history, the system will list up to 500 previously executed commands, allowing you to reuse them without re-entering. Keep in mind that only users with sudo privileges can execute this command. How this utility runs also depends on which Linux shell you use.
To run it, enter the command below:

history
[option]

This command supports many options, such as:

- **-c** clears the complete history list.
- **-d** offset deletes the history entry at the OFFSET position.
- **-a** appends history lines.

2.4.5 echo command

The echo command is a built-in utility that displays a line of text or string using the standard output.

Here’s the basic syntax:

echo [option]
[string]

For example, you can display the text Hostinger Tutorials by entering:

echo “Fast
Tutorials”

This command supports many options, such as:

- **-n** displays the output without the trailing newline.
- **-e** enables the interpretation of the following backslash escapes:
- **\a** plays sound alert.
- **\b** removes spaces in between a text.
- **\c** produces no further output.
- **-E** displays the default option and disables the interpretation of backslash escapes.

2.4.6 alias, unalias commands

alias allows you to create a shortcut with the same functionality as a command, file name, or text. When executed, it instructs the shell to replace one string with another.

To use the alias command, enter this syntax:

alias Name=String

For example, you want to make e the alias for the exit command:

alias e=exit

On the other hand, the unalias command deletes an existing alias.

Here’s what the general syntax looks like:

unalias [alias_name]

2.4.7 | (Pipe command)

If a user in Linux likes to combine two or more commands, pipes is the option. Pipe is represented by the symbol ‘|’

2.4.8 &&

And command **&&** is used to only allow the next command to run if the previous one is successful.

2.5 System Related Commands:

Commands related to system includes some basic commands and commands for managing users and groups in Linux that require root access.

Basic System Related Commands:

Following are the basic system related commands:

1. date
2. shutdown, poweroff , reboot
3. ps
4. kill
5. df
6. du
7. top
8. htop
9. uname

2.5.1 date command

The date command will display the system date.

2.5.2 shutdown, poweroff and reboot commands:

Command	Switch	Description	Example	Output
shutdown	None	Power off the computer	shutdown now	The system will shutdown now for maintenance
	-r	Reboots after shutdown	shutdown –r now	None
poweroff	none	Shutdowns computer	sudo poweroff	None
reboot	none	Restarts computer	sudo reboot	None

2.5.3 ps command

The process status or ps command produces a snapshot of all running processes in your system. The static results are taken from the virtual files in the /proc file system.

Executing the ps command without an option or argument will list the running processes in the shell along with:

- The unique process ID (PID)
- The type of the terminal (TTY)
- The running time (TIME)
- The command that launches the process (CMD)

Here are some acceptable options you can use:

- **-T** displays all processes associated with the current shell session.
- **-u** username lists processes associated with a specific user.
- **-A** or -e shows all the running processes.

2.5.4 kill command

Use the kill command to terminate an unresponsive program manually. It will signal misbehaving applications and instruct them to close their processes.

To kill a program, you must know its process identification number (PID). If you don't know the PID, run the following command:

ps ux

After knowing what signal to use and the program's PID, enter the following syntax:

kill [signal_option] pid

There are 64 signals that you can use, but these two are among the most commonly used:

- **SIGTERM** requests a program to stop running and gives it some time to save all of its progress. The system will use this by default if you don't specify the signal when entering the kill command.
- **SIGKILL** forces programs to stop, and you will lose unsaved progress. For example, the program's PID is 63773, and you want to force it to stop:

  kill SIGKILL 63773

2.5.5 df command

Use the df command to report the system's disk space usage, shown in percentage and kilobyte (KB).
Here's the general syntax:

df [options] [file]

For example, enter the following command if you want to see the current directory's system disk space usage in a human-readable format:

df -h

These are some acceptable options to use:

- **df -m** displays information on the file system usage in MBs.
- **df -k** displays file system usage in KBs.
- **df -T** shows the file system type in a new column.

2.5.6 du command

If you want to check how much space a file or a directory takes up, use the du command. You can run this command to identify which part of the system uses the storage excessively.

Remember, you must specify the directory path when using the du command. For example, to check
/home/user/Documents enter:

du /home/user/Documents

Adding a flag to the du command will modify the operation, such as:

- **-s** offers the total size of a specified folder.

- **-m** provides folder and file information in MB
- **-k** displays information in KB.
- **-h** informs the last modification date of the displayed folders and files.

2.5.7 top command

The top command in Linux Terminal will display all the running processes and a dynamic real-time view of the current system. It sums up the resource utilization, from CPU to memory usage.

The top command can also help you identify and terminate a process that may use too many system resources.

2.5.8 htop command

The htop command is an interactive program that monitors system resources and server processes in real time. It is available on most Linux distributions, and you can install it using the default package manager.

Compared to the top command, htop has many improvements and additional features, such as mouse operation and visual indicators.

To use it, run the following command:

htop [options]

You can also add options, such as:

- **-d** or –delay shows the delay between updates in tenths of seconds.
- **-C** or –no-color enables the monochrome mode.
- **-h** or –help displays the help message and exit.

To run the command, simply enter **top** into the CLI.

2.5.9 uname command

The uname or unix name command will print detailed information about your Linux system and hardware. This includes the machine name, operating system, and kernel. To run this command, simply enter uname into your CLI.

Here’s the basic syntax:

uname [option]

These are the acceptable options to use:

- **-a** prints all the system information.
- **-s** prints the kernel name.
- **-n** prints the system’s node hostname.

##

2.6 Managing users and groups in Linux:

For root privilege, sudo command is used.

2.6.1 sudo command

Short for superuser do, sudo is one of the most popular basic Linux commands that lets you perform tasks that require administrative or root permissions.

When using sudo, the system will prompt users to authenticate themselves with a password. Then, the Linux system will log a timestamp as a tracker. By default, every root user can run sudo commands for 15 minutes/session.

If you try to run sudo in the command line without authenticating yourself, the system will log the activity as a security event.

Here's the general syntax:

sudo (command)

You can also add an option, such as:

- **-k** or –reset-timestamp invalidates the timestamp file.
- **-g** or –group=group runs commands as a specified group name or ID.
- **-h** or –host=host runs commands on the host.

For managing users and groups following commands are used in Linux and it requires root access.

1. useradd
2. addgroup
3. adduser
4. usermod
5. deluser
6. passwd

Managing Users and Groups in Linux (root user only)				
useradd	None	Creates a new user profile or update Existing user information	useradd abc	User Created
addgroup	None	Add a group to the system	addgrop example	Adding group `example' (GID 1003) ... Done.

adduser	None	Creates a user account that can be used for login	adduser usernam e	Ask for password and some data along with confirmation and creates the account
	--ingroup	Creates user account and add that user in a group specified	adduser --ingroup sudo abc	Same as adduser <username> and also it adds the user to the group
usermod	-a -G	Modify an existing user	Usermod –a –G sudo abc	Add already existing User to already existing group
deluser	None	Deletes the user from the system	deluser abc	Removing user `abc' ... Done.
	--group	Deletes the group from the system	deluser --group example	Removing group `example' ... Done.
	-- removehome	Removes the user along with its home folder directory	deluser - removehome abc	Removing files ... Removing user `abc' ... Done.
passwd	None	Change password of the current logged in user or user specified	passwd OR passwd abc	passwd: password updated successfully

2.7 Network Related Commands:

Linux terminal provides commands for interfacing with networks, some of the basic commands are:

1. ping
2. wget
3. hostname
4. ip

2.7.1 ping command

The ping command is one of the most used basic Linux commands for checking whether a network or a server is reachable. In addition, it is used to troubleshoot various

connectivity issues.

Here's the general format:

ping [option] [hostname_or_IP_address]

For example, you want to know whether you can connect to Google and measure its response time:

ping google.com

2.7.2 wget command

The Linux command line lets you download files from the internet using the wget command. It works in the background without hindering other running processes.

The wget command retrieves files using HTTP, HTTPS, and FTP protocols. It can perform recursive downloads, which transfer website parts by following directory structures and links, creating local versions of the web pages.

To use it, enter the following command:

wget [option] [url]

For example, enter the following command to download the latest version of WordPress:

wget https://wordpress.org/latest.zip

2.7.3 hostname command

Run the hostname command to know the system's hostname. You can execute it with or without
an option. Here's the general syntax:

hostname [option]

There are many optional flags to use, including:

- **-a** or –alias displays the hostname's alias.
- **-A** or –all-fqdns displays the machine's Fully Qualified Domain Name (FQDN).
- **-i** or –ip-address displays the machine's IP address.

For example, enter the following command to know your computer's IP address:

hostname –i

2.7.4 ip command

The ip command is a Linux net-tool for system and network administrators. IP stands for Internet Protocol and as the name suggests, the tool is used for configuring network interfaces.

Older Linux distributions used the ifconfig command, which operates similarly. However,

ifconfig has a limited range of capabilities compared to the ip command.

To Use the ip command enter:

ip [OPTION] OBJECT {COMMAND | help}

OBJECTS (or subcommands) that you will use most often include:

1. link (**l**) – used to display and modify network interfaces.
2. address (**addr/a**) – used to display and modify protocol addresses (IP, IPv6).
3. route (**r**) – used to display and alter the routing table.
4. neigh (**n**) – used to display and manipulate neighbor objects (ARP table).

3 Managing Files & Directories:

3.1 File Editor and creation commands:

3.1.1 Text Editor

A text editor is a software where you can enter text in its native format and save it to file. A word processor is a software where you can take text and process its appearance, format, spell-check, paragraph settings, etc. Linux has a number of text editors (both graphical and command-line based). The one which you will use most commonly is the nano text editor.

3.1.1.1 NANO Editor

There are many text editors available for Linux. At the moment you will have access to the *nano* editor. Nano is an advanced text editor provided by GNU. Simply typing *nano* on the shell will give you the editor as shown in Figure below

```
nano
```

You may also start your nano by explicitly mentioning the file you want to work on. This file may be already existing or you may be creating a new one.

```
nano <myFile>
```

Near the end of your screen you will see a list of shortcuts. The ones which you should get yourself familiar with are as such:

- **CTRL+X** for exit
- **CTRL+O** for saving
- **CTRL+W** for searching
- **CTRL+K** for cutting
- **CTRL+U** for pasting
- **CTRL+C** for displaying cursor position

Other commands are listed at the bottom of the text-editor window.

Figure 2.4: Nano Text Editor

3.1.1.2 VIM Editor

Vim (Vi IMproved) is a highly configurable text editor built for efficient text editing. It is an improved version of the Unix-based ‘vi‘ editor. To open Vim, simply type the following on the shell:

```
vim
```

You can also start Vim by specifying a file you want to edit. If the file does not exist, Vim will create a new one:

```
vim <myFile>
```

Vim operates in **different modes**, and some of the basic modes include:

Normal Mode: This is the default mode where commands can be issued.

Insert Mode: For entering or editing text.

Command Mode: Used to perform various actions like saving or exiting.

Key commands to familiarize yourself with include:

- ESC Switch to Normal Mode.
- :w Save the file.
- :q Quit Vim.
- :wq Save and quit Vim.
- :q! Quit without saving changes.
- i Enter Insert Mode.
- / Search for text.
- dd Delete a line.
- yy Copy (yank) a line.
- p Paste a copied line.
- u Undo the last change.
- CTRL+R Redo the last undone change.

Below is a typical **example** of how to use Vim:

```
vim myFile.txt
# Press "i" to enter Insert Mode and edit your
file # Press "ESC" to return to Normal Mode

# Type ":wq" to save and exit
```

3.1.2 touch command

The touch command allows you to create an empty file or generate and modify a timestamp in the Linux command line.

For example, enter the following command to create an HTML file named Web in the Documents directory:

touch /home/username/Documents/Web.html

3.1.3 Searching

By default, searching for files or directories is performed using the *find* command. Another powerful search program is *locate*.

3.1.3.1 Using Plocate

The locate program uses a database to search for files. The program gives results much more quickly than find command. The downside to updating the database. Updating is done using updatedb command. The format of command is also simple as compared to find.

```
sudo apt update && sudo apt install plocate -y
sudo updatedb

locate *.pdf
```

To look for content that contains two or more words, use an asterisk (*). For example:

locate -i school*not

The command will search for files that contain the words school and note, whether they use uppercase or lowercase letters.

3.1.3.2 Using Find

Use the find command to search for files within a specific directory and perform subsequent operations.

The **syntax** of find command is as such:

find <where> -name <what>

For example, if I want to find all pdf files in / directory, I will give:

find / -name *.pdf

For example, you want to look for a file called notes.txt within the home directory and its subfolders:

find /home -name notes.txt

Here are other variations when using find:

find -name filename.txt to find files in the current directory.

find ./ -type d -name directoryname to look for directories.

3.1.3.3 grep command

Another basic Linux command on the list is grep or global regular expression print. It lets you find a word by searching through all the texts in a specific file.

Once the grep command finds a match, it prints all lines that contain the specific pattern. This command helps filter through large log files.

For example, you want to search for the word blue in the notepad.txt file:
grep blue notepad.txt

The command's output will display lines that contain blue.

3.1.4 Displaying Output Commands:

3.1.4.1 head command

The head command allows you to view the first ten lines of a text. Adding an option lets you change the number of lines shown. The head command is also used to output piped data to the CLI.

Here's the **general syntax:**

head [option] [file]

For instance, you want to view the first ten lines of note.txt, located in the current directory:

head note.txt

Below are some options you can add:

- **-n** or –lines prints the first customized number of lines. For example, enter head -n 5 filename.txt to show the first five lines of filename.txt.
- **-c** or –bytes prints the first customized number of bytes of each file.
- **-q** or –quiet will not print headers specifying the file name.

3.1.4.2 tail command

The tail command displays the last ten lines of a file. It allows users to check whether a file has new data or to read error messages.

Here's the **general format**:

tail [option] [file]

For example, you want to show the last ten lines of the colors.txt file:

tail -n colors.txt

3.1.4.3 cat command

Concatenate, or cat, is one of the most frequently used Linux commands. It lists, combines, and writes file content to the standard output. To run the cat command, type cat followed by the file name and its extension. For instance:

cat filename.txt.

Here are other ways to use the cat command:

cat > filename.txt creates a new file.

cat filename1.txt filename2.txt > filename3.txt merges filename1.txt and filename2.txt and stores the output in filename3.txt.

tac filename.txt displays content in reverse order.

3.2 File Permissions

All files and directories in Linux have an associated set of owner permissions that are used by the operating system to determine access. These permissions are grouped into *three sets of three bits*. The sets represent {owner, group, everyone else} whereas the bits represent {read, write, execute}. So overall, we have 9 permissions (See Table 2.1).

	Read	Write	Execute
Owner	1	1	1
Group	1	1	0
Others	0	0	1
	2^2	2^1	2^0

Table 2.1: Representation of File Permissions

If we look at the first row for Owner, we see three 1's. Which specify that the Owner has read, write, and execute permissions on a file or directory. Since all of the bits are in allow mode, we can add them up together to get $2^2 + 2^1 + 2^0 = 4 + 2 + 1 = 7$. Similarly, the second row for Group, i.e., users within the same group as that of the file owner, have read and write, but no execute permissions. This will be translated only as $2^2 + 2^1 = 4 + 2 = 6$. And lastly, every other user can only execute files and have no permission to read or write to them. This will be translated as $2^0 = 1$. So the file permissions would be *761*.

To give the permission of 761 to a file, we would use the command:

```
chmod 761 <filename>
```

You can view the file permissions using the command:

```
ls -lh
```

3.2.1 Changing File Permissions with the chmod Command

The chmod command, which stands for "change mode," is used to set permissions (read, write, execute) on a file or directory for the owner, group, and others.

3.2.1.1 Syntax

```
chmod permissions filename
```

There are two ways to use the chmod command:

1. Absolute (Numeric) Mode
2. Symbolic Mode

3.2.1.2 1. Absolute (Numeric) Mode

In this mode, file permissions are represented as a three-digit octal number. The table below explains the numeric values for different permission types:

Number	Permission Type	Symbol
0	No Permission	—
1	Execute	–x
2	Write	-w-
3	Execute + Write	-wx
4	Read	r–
5	Read + Execute	r-x
6	Read + Write	rw-
7	Read + Write + Execute	rwx

example, changing a file's permissions to 764 using:

```
chmod 764 sample
```

764 means:

- Owner can read, write, and execute (rwx).
- Group can read and write (rw-).
- Others can only read (r–).

3.2.1.3 2. Symbolic Mode

In symbolic mode, permissions are changed for specific owners (user, group, or others) using mathematical symbols. This method uses:

- + to add a permission
- – to remove a permission
- = to set permissions and override previous ones

The owners are denoted as:

- u: User/Owner
- g: Group
- o: Others
- a: All

Examples

```
chmod o+x testfile.txt # Give execute permission to others
chmod g+w testfile.txt # Give write permission to group
chmod u+rwx testfile.txt # Give all permissions to the user

chmod ugo+rwx testfile.txt # Give all permissions to everyone
chmod a-rwx testfile.txt # Remove all permissions from everyone
```

3.2.2 Directory Permissions

Directory permissions work similarly to file permissions but are applied to directory operations. For example:

```
chmod u+x test # Add execute permission to access the directory
chmod g+wrx test # Give all permissions to group

chmod o-wrx test # Remove all permissions from others
```

Changing directory permissions affects the ability to access, modify, or list its contents. Use these commands with caution to avoid accidental access issues.

3.3 File Ownership

A file's owner can be changed using the command:

```
chown <username> <filename>
```

Where, *cusername*> would be the username of any user on your system, and *cfilename*> is the target to which this setting is going to apply to. The chown command stands for "change

owner."

- ch stands for "change."
- own stands for "owner."

3.3.1 Examples

```
sudo chown ayyzenn test4
```

This command changes the directory's current owner to ayyzenn. Verify the change using:

```
ls -l
```

Similarly, to change the owner of a file:

```
sudo chown ayyzenn file.txt ls -l
```

3.3.2 Changing User and Group Ownership

If you want to change both the user and the group for a file or directory, use the following syntax:

```
chown user:group filename
```

For example:

```
sudo chown ayyzenn:ayyzenn file.txt ls -l
```

3.3.3 Handling Permission Issues

When you attempt to write to a file owned by another user, you might encounter an error:

```
cat > test.txt # Output: "bash: test.txt: Permission denied"
```

This happens because the current user does not own the file. You can resolve this by changing the ownership of the file:

```
sudo chown ayyzenn:ayyzenn test4 ls -l
```

3.3.4 Changing Group Ownership Only

To change the group ownership of a file or directory, use the chgrp command, which stands for "change group."

```
sudo chgrp ayyzenn file.txt
```

4 Wildcards:

Following are the wildcards used in Linux shell terminal.

- **WildCard '*' :** will match against none or one or a string of more than a character

  Example:'**ls file***' - list all the files in current directory starting with filename 'file'.
  '**ls *2.txt'** - list all the files in current directory ending with '2.txt'
- **WildCard '?':** can be used to match one character
  Example: '**ls file.tx?**' - list all the files that begins with 'file.tx'
- **WildCard '[]':** matches one specified character out of a group of characters
  Example: '**ls rmt[12345]**' - list all the file that begins with 'rmt' and has a 1,2,3,4 or 5 after it.