Assignment-2

Name: Suraj P Das

USN: ENG25CY1006

Section: 3C

Roll no: 66

1. **Pwd:**
   - pwd command displays the full path of the current working directory. The output shows the hierarchical structure of directories from the root directory to the user's current location within the file system.

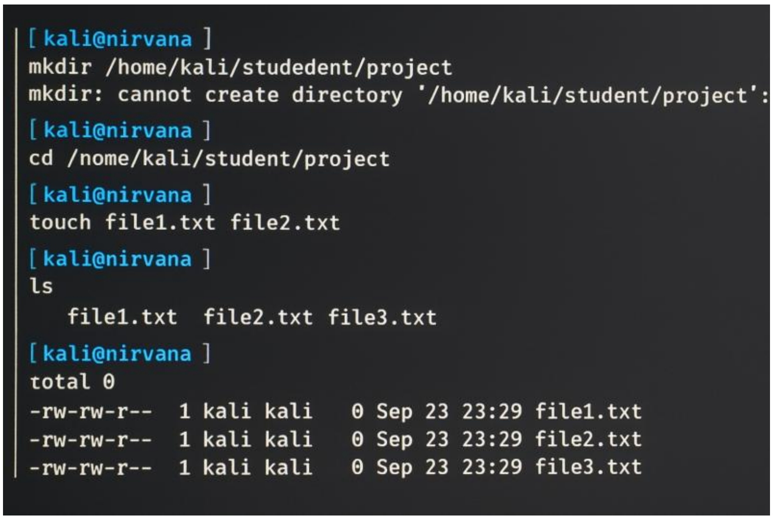
   whoami:
   - whoami command outputs the effective username of the current user. This is particularly useful when working in environments where user accounts might be switched as it reveals the identity of the user currently executing commands.

   hostname:
   - hostname command displays the name of the host system (computer) on which the command is executed. This is the network name assigned to the machine, which can be useful for identification in networked environments.

2. **The command to create a directory project inside the /home/student folder and keep three .txt file into it:**

   -
     ```
     mkdir /home/kali/student/project
       cd /home/kali/student/project
       touch file1.txt file2.txt file3.txt
       ls -l
     ```

- **Snapshot:**



```
[ kali@nirvana ]
mkdir /home/kali/studedent/project
mkdir: cannot create directory '/home/kali/student/project':
[ kali@nirvana ]
cd /nome/kali/student/project
[ kali@nirvana ]
touch file1.txt file2.txt
[ kali@nirvana ]
ls
   file1.txt  file2.txt file3.txt
[ kali@nirvana ]
total 0
-rw-rw-r--  1 kali kali   0 Sep 23 23:29 file1.txt
-rw-rw-r--  1 kali kali   0 Sep 23 23:29 file2.txt
-rw-rw-r--  1 kali kali   0 Sep 23 23:29 file3.txt
```

3. The difference between absolute path and relative path.

1. **Absolute path:**
   - An absolute path provides the complete and unambiguous location of a file or directory, starting from the root of the file system.
   - It explicitly states the entire hierarchy of directories from the root to the target item.
   - This path will work regardless of the current working directory.

   Example:

   - /home/student/project/file1.txt
2. **Relative path:**
   - A relative path specifies the location of a file or directory in relation to the current working directory.
   - It does not start from the root, but rather describes the path from the current location to the target.

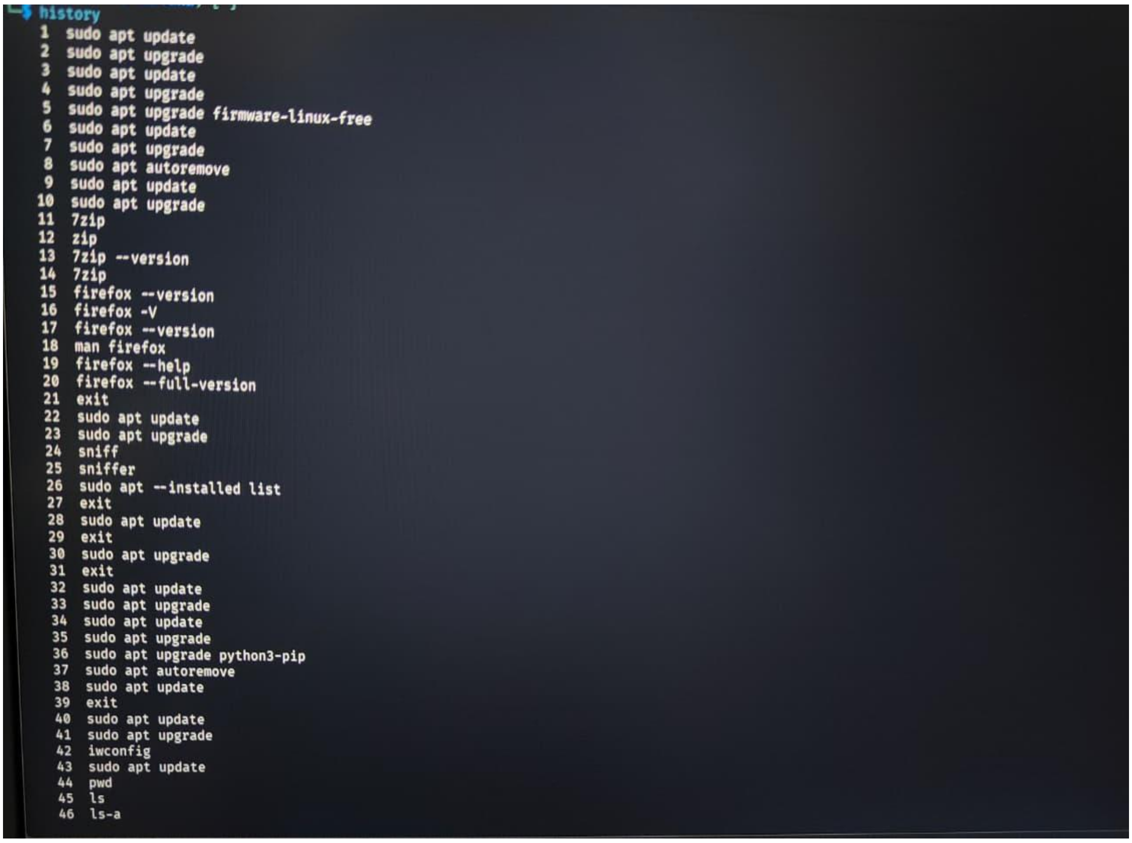
Example:

- /home/student

  cd project

  cat project/file1.txt

4. The history command will give the already executed command traces in the terminal. It displays a list of commands that have already been executed in the current terminal session. Each command is shown with a number so you can easily re-run it.
    - **Snapshot:**



```
└─$ history
    1  sudo apt update
    2  sudo apt upgrade
    3  sudo apt update
    4  sudo apt upgrade
    5  sudo apt upgrade firmware-linux-free
    6  sudo apt update
    7  sudo apt upgrade
    8  sudo apt autoremove
    9  sudo apt update
   10  sudo apt upgrade
   11  7zip
   12  zip
   13  7zip --version
   14  7zip
   15  firefox --version
   16  firefox -V
   17  firefox --version
   18  man firefox
   19  firefox --help
   20  firefox --full-version
   21  exit
   22  sudo apt update
   23  sudo apt upgrade
   24  sniff
   25  sniffer
   26  sudo apt --installed list
   27  exit
   28  sudo apt update
   29  exit
   30  sudo apt upgrade
   31  exit
   32  sudo apt update
   33  sudo apt upgrade
   34  sudo apt update
   35  sudo apt upgrade
   36  sudo apt upgrade python3-pip
   37  sudo apt autoremove
   38  sudo apt update
   39  exit
   40  sudo apt update
   41  sudo apt upgrade
   42  iwconfig
   43  sudo apt update
   44  pwd
   45  ls
   46  ls-a
```

5. Comparison of functionality between find and locate command:

Feature	**find**	**locate**
Search method	Real-time scan of filesystem	Searches in pre-built database
Speed	Slower	Faster
Accuracy	Always accurate, up-to-date	May be outdated if not run
Use case	When accuracy is critical, or searching with conditions	When you need quick filename lookup
Example	find /home -name file1.txt	locate file1.txt

- locate is faster
- because it doesn't scan the filesystem in real time.
- It queries a pre-built **index/database** of files, so results come almost instantly.

6. In Linux, the command used to modify file permissions is chmod.

Syntax:

```
chmod [options] permissions filename
```

Types of permissions:

- r → read
- w → write
- x → execute

Example:

- Give execute permission to the user for a file named script.sh:

```
chmod u+x script.sh
```

7. In Linux, file permissions are shown in a 10-character string.

- -rw-r--r—

1. **First character (-)** → File type
   - - = regular file
   - d = directory
   - l = symbolic link
2. **Next three characters (rw-)** → Owner (user) permissions.
   - r = read allowed
   - w = write allowed
   - - = execute not allowed
3. **Next three characters (r--)** → Group permissions.
   - r = read allowed
   - -- = no write, no execute
4. **Last three characters (r--)** → Others (world) permissions
   - r = read allowed
   - -- = no write, no execute

8. The difference between chown and chgrp:
The chown and chgrp commands in Linux are used to modify ownership attributes of files and directories, specifically focusing on user and group ownership.

chown:

- This command changes the user owner of a file or directory.
- It can also be used to change both the user owner and the group owner simultaneously.
- chown can change both user and group ownership.
- chown offers more flexibility as it can modify both aspects of ownership in a single command.

Example:

- ls -l file.txt

```
-rw-r--r-- 1 jagadeesh student 0 Sep 24 21:00 file.txt
sudo chown admin file.txt
ls -l file.txt
-rw-r--r-- 1 admin student 0 Sep 24 21:00 file.txt
```

Chgrp:

- This command changes only the group owner of a file or directory.
- chgrp is limited to changing only the group ownership.

Example:

-
  ```
  ls -l file.txt
  -rw-r--r-- 1 admin student 0 Sep 24 21:00 file.txt
  sudo chgrp staff file.txt
  ls -l file.txt
  -rw-r--r-- 1 admin staff 0 Sep 24 21:00 file.txt
  ```

9. To set file permissions in a way that allows multiple users to access a file but only permits the owner to write to it, you would use the chmod command in a Unix-like operating system. This can be achieved using either symbolic or numeric modes**.**

Using Numeric (Octal) Mode:

- The octal representation for the desired permissions is 644.
- The first digit 6 (binary 110) grants the owner read and write permissions (4+2).
- The second digit 4 (binary 100) grants the group read-only permission.

- The third digit 4 (binary 100) grants others read-only permission.

To apply these permissions to a file named your_file.txt, execute the following command:

- chmod 644 your_file.txt

Using Symbolic Mode:

- Alternatively, you can use symbolic mode to achieve the same result.
- chmod u=rw,g=r,o=r your_file.txt
- u=rw: Sets read and write permissions for the owner.
- g=r: Sets read-only permission for the group.
- o=r: Sets read-only permission for others.

Both methods achieve the same outcome: the owner can read and modify the file, while members of the file's group and all other users can only read its content.

10. To check the manual page for any Linux command, use the man command followed by the name of the command you want to learn about.

Command:

- man <command_name>

When you type man followed by a command name, it retrieves and displays the manual page for that command, offering comprehensive details on how to use it, including all available options and flags. This command is crucial for learning about the tools and commands available in the Linux operating system.