

| **TITLE: Exploring basic Commands of UNIX: Shell, Processes, Files** |
| --- |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**AIM:** To Explore basic commands for handling File system under Unix/Linux using shell scripts.(Creating groups, chown , chmod , directory name, tty , diff, umask).

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected Outcome of Experiment:**

**CO 1.** To introduce basic concepts and functions of operating systems.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. **Silberschatz A., Galvin P., Gagne G. “Operating Systems Principles”, Willey Eight edition.**
2. **Achyut S. Godbole , Atul Kahate “Operating Systems”, McGraw Hill Third Edition.**
3. **Sumitabha Das “ UNIX Concepts & Applications”, McGraw Hill Second**

**Edition.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:**

An operating system (OS) is a resource manager. It takes the form of a set of software routines that allow users and application programs to access system resources (e.g. the CPU, memory, disks, modems, printers network cards etc.) in safe efficient and abstract way.

* The operating system kernel is in direct control of the underlying hardware. The kernel provides low-level device, memory and processor management functions (e.g. dealing with interrupts from hardware devices, sharing the processor among multiple programs, allocating memory for programs etc.)
* Basic hardware-independent kernel services are exposed to higher-level programs through a library of system calls (e.g. services to create a file, begin execution of a program, or open a logical network connection to another computer).
* Application programs (e.g. word processors, spreadsheets) and system utility programs (simple but useful application programs that come with the operating system, e.g. programs which find text inside a group of files) make use of system calls. Applications and system utilities are launched using a shell (a textual command line interface) or a graphical user interface that provides direct user interaction.

Operating systems can be distinguished from one another by the system calls, system utilities and user interface they provide, as well as by the resource scheduling policies implemented by the kernel.

UNIX has been a popular OS for more than two decades because of its multi-user, multi-tasking environment, stability, portability and powerful networking capabilities.

Linux is a free open source UNIX OS for PCs.

Linux has all of the components of a typical OS :

* **Kernel**

The Linux kernel includes device driver support for a large number of PC hardware devices (graphics cards, network cards, hard disks etc.), advanced processor and memory management features, and support for many different types of file systems. In terms of the services that it provides to application programs and system utilities, the kernel implements most BSD and SYSV system calls, as well as the system calls described in the POSIX.1 specification.

The kernel (in raw binary form that is loaded directly into memory at system startup time) is typically found in the file /boot/vmlinuz, while the source files can usually be found in /usr/src/linux.

* **Shells and GUIs**

Linux supports two forms of command input: through textual command line shells similar to those found on most UNIX systems (e.g. sh - the Bourne shell, bash - the Bourne again shell and csh - the C shell) and through graphical interfaces (GUIs) such as the KDE and GNOME window managers.

* **System Utilities**

Virtually every system utility that you would expect to find on standard implementations of UNIX has been ported to Linux. This includes commands such as ls, cp, grep, awk, sed, bc, wc, more, and so on. These system utilities are designed to be powerful tools that do a single task extremely well (e.g. grep finds text inside files while wc counts the number of words, lines and bytes inside a file). Users can often solve problems by interconnecting these tools instead of writing a large monolithic application program.

* **Application programs**

Linux distributions typically come with several useful application programs as standard. Examples include the emacseditor, xv (an image viewer), gcc (a C compiler), g++ (a C++ compiler), xfig (a drawing package), latex (a powerful typesetting language) and soffice (StarOffice, which is an MS-Office style clone that can read and write Word, Excel and PowerPoint files).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Description of Commands and options:

**DOS commands:** Attrib, dir, at, chkdsk, shutdown, tree, create a batch file, output and input redirection

**Windows utilities**: msconfig, defragmenter, performance monitor, task manager, registry editor, event viewer, process explorer

Unix Commands:

1. Unix file operations: ls, cp, rm , mv, chmod, chown ,chgrp
2. Text file operations in Unix : cat , more , less , head, tail , grep
3. Unix directory management commands : cd, pwd , ln, mkdir, rmdir
4. Unix system status commands: hostname, w, uname
5. Process management: ps, top, kill
6. Unix users commands: whoami , id, groups, passwd , who, last

**Implementation details:**

echo – the echo command in Linux is a built-in command that allows users to display lines of text or strings that are passed as arguments.

cat – concatenate files and show contents to the standard output

head – show the first 10 lines of text file (you can specify any number of lines)

tail – show the last 10 lines of text file (any number can be specified)

chmod – change file/directory access permissions

chmod 111 – file/directory can be executed by anyone, but cannot be read or written by anyone

Ls – ls is a Linux shell command that lists directory contents of files and directories. It provides valuable information about files, directories, and their attributes.

ls -l – known as a long format that displays detailed information about files and directories.

chmod 333 – the file/directory can be written by anyone, but cannot be read by anyone

chmod 751 – the file/directory can be read, written and execute by the owner; the group can read and execute the file/directory, but cannot write it; others can only execute the file/directory

ls -a – represent all files including hidden files and directories in the listing

cp – It creates an exact image of a file on a disk with different file name

$SHELL – displays the current user’s default shell /etc/shells – displays all available shells

rmdir – removes directories in Unix

ls – list files and directories

mv - rename or move files and directories to another location

chown – change file/directory ownership

more – basic pagination when viewing text files or parsing Unix commands output

less – an improved pagination tool for viewing text files (better than more command)

grep – search for patterns in text files

grep -c – used to count the number of lines that match a specified pattern in a file

grep -i – used to display the lines that contains the specified pattern in a file

cd – used to change directory

pwd – confirm current directory

hostname – show or set server hostname

w – display system load, who’s logged in and what they are doing

uname – print Unix system information

ps – list processes

top – show tasks and system status

kill – kill a process

whoami – show your username

id – print user identity

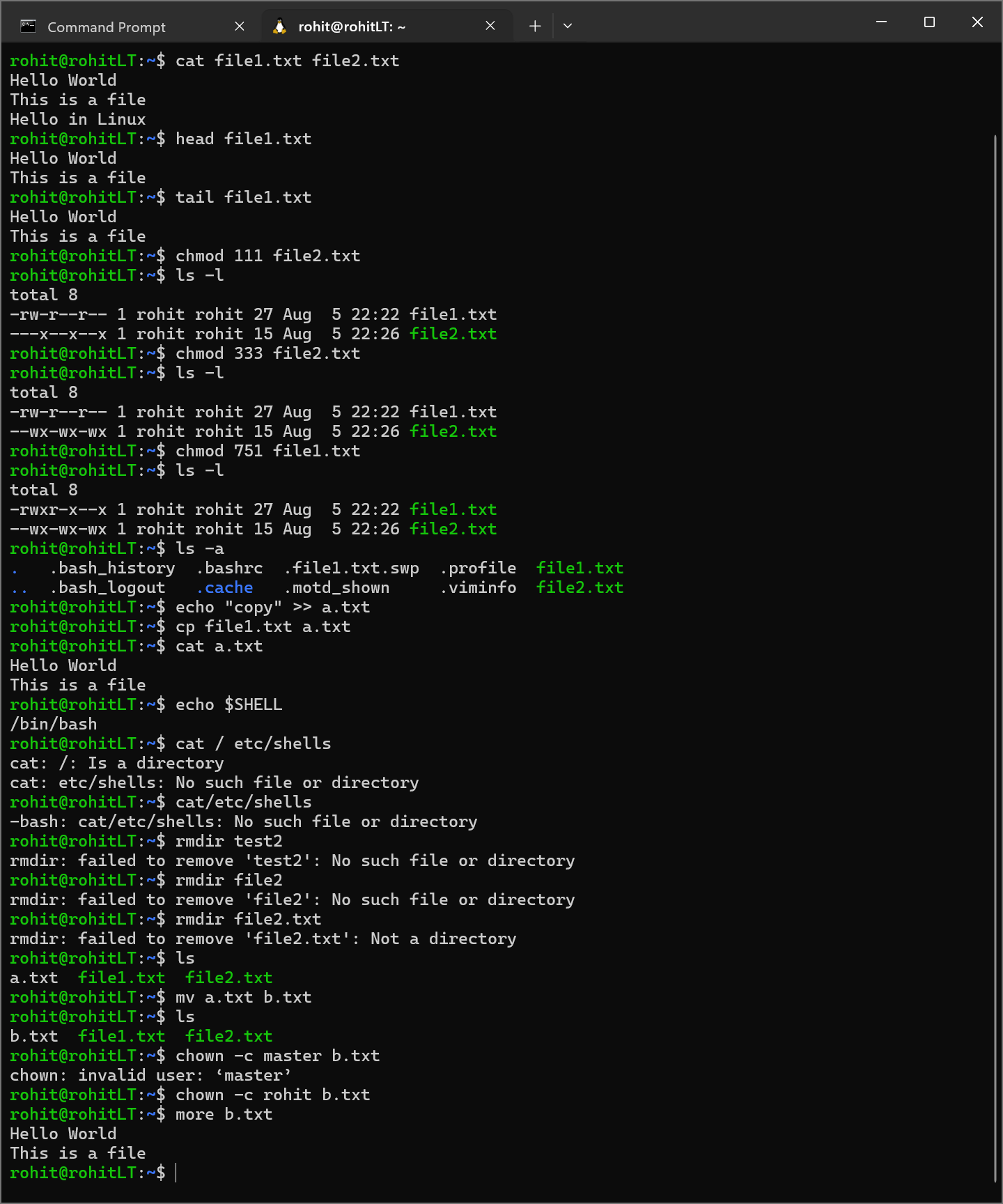
groups – show which groups user belongs to

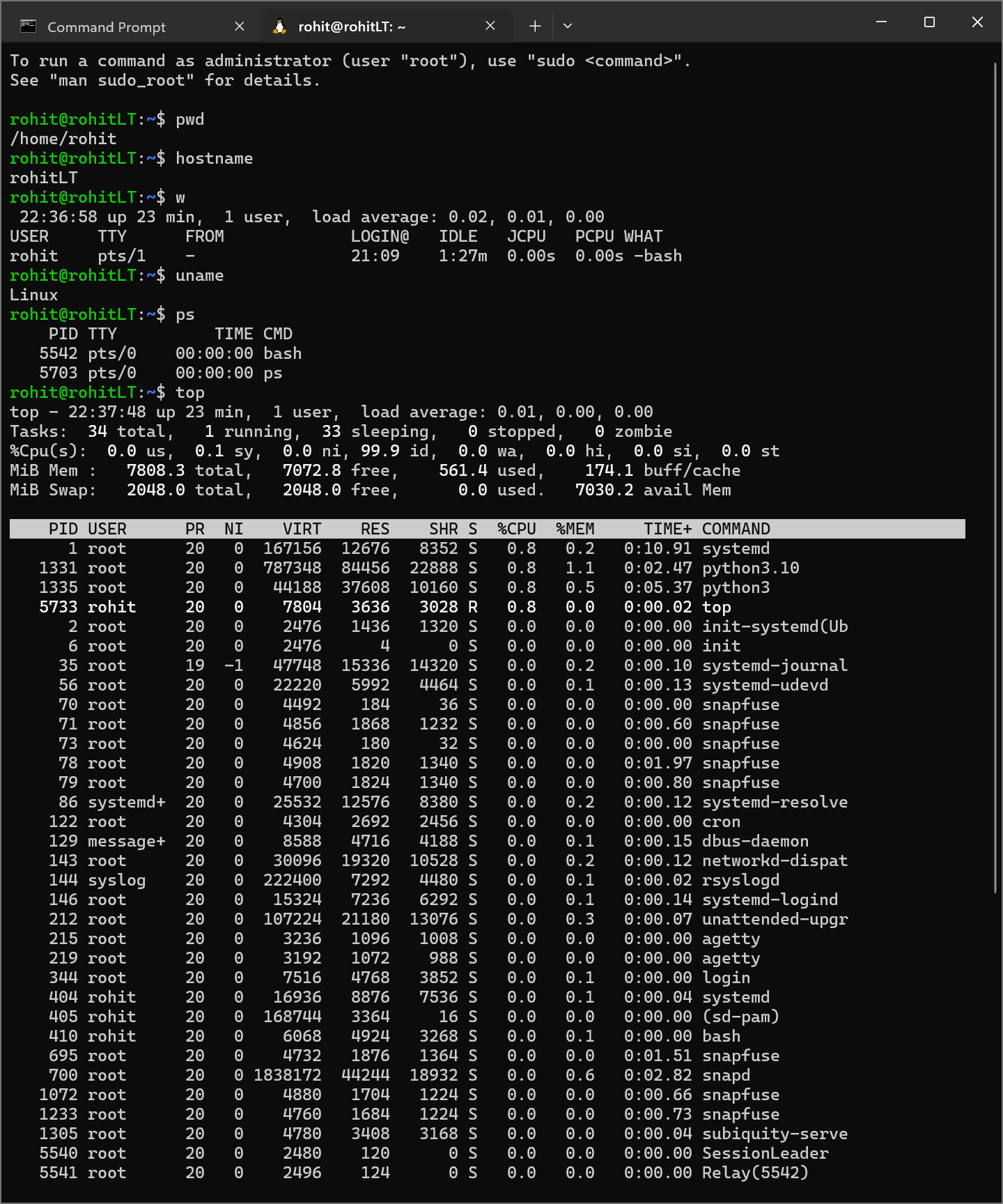
passwd – change user password

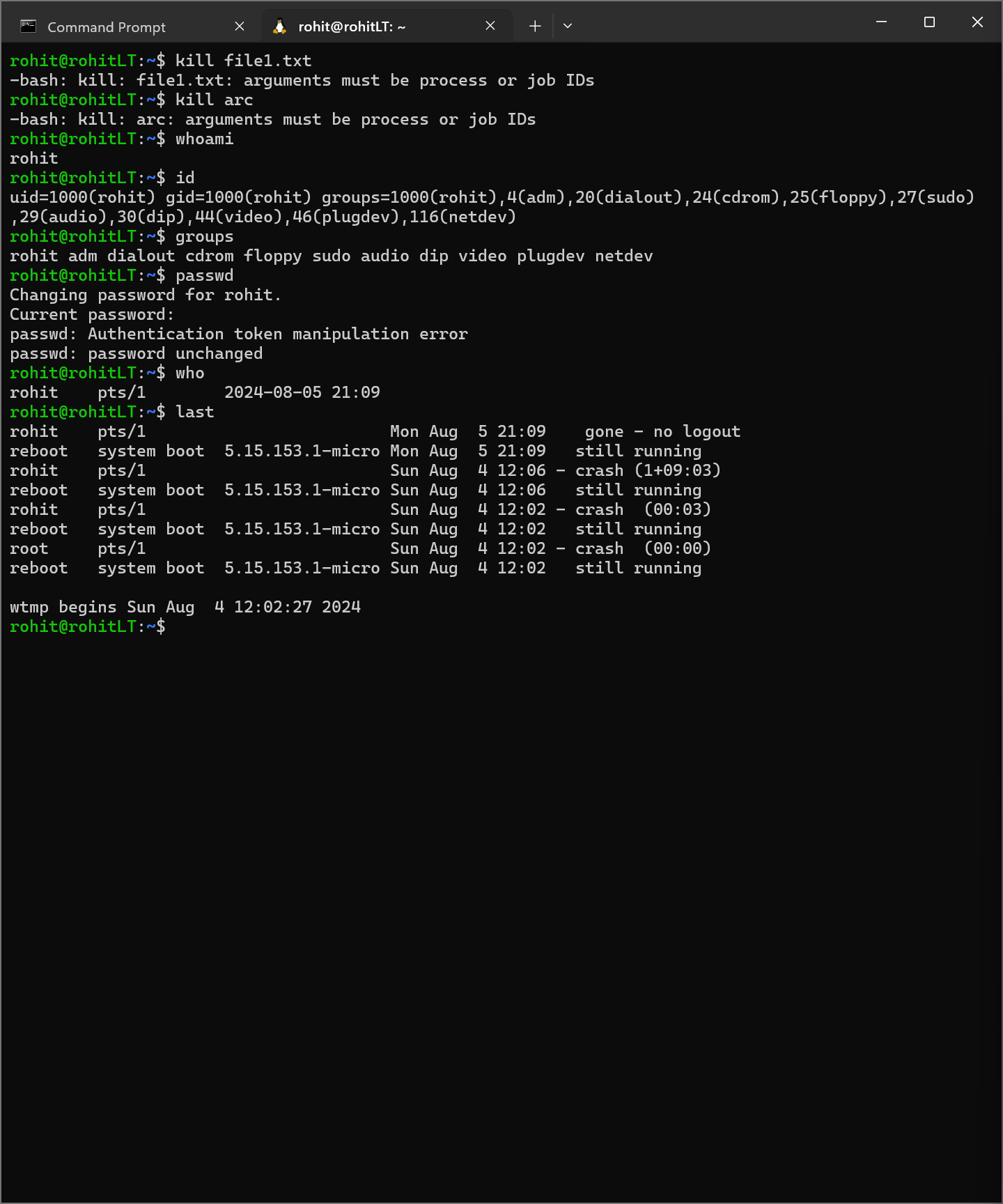
who – find out who is logged into the system

last – show history of logins into the system









**Conclusion:** Explored basic commands for handling File system under Unix/Linux using shell scripts.

**Post Lab Descriptive Questions**

**1.** **Explain how do you read and interpret syntax of any OS command.**

To read and interpret the syntax of any OS command, start by identifying the command itself, which usually appears first. Next, look for any options or flags, typically indicated by a dash (-) or double dash (--), which modify the command's behavior. Finally, identify any arguments or parameters that the command may require, such as file names or other data inputs. Understanding the context of the command and its typical usage can also provide insights into its function.

**2.** **Explain different functions of the operating systems.**

Operating systems perform several key functions, including managing hardware resources (CPU, memory, disk space), providing a user interface (command-line or graphical), and handling system security and access controls. They also facilitate multitasking, allowing multiple applications to run concurrently, manage files and directories, and coordinate communication between software and hardware components.

**3.** **What are the default permissions assigned by Unix for Directory.**

In Unix, the default permissions for a directory are typically set to 755. This means that the owner of the directory has read, write, and execute permissions (7), while the group and others have read and execute permissions (5 each). The execute permission for a directory allows users to enter the directory and access its contents.

**4.** **Give the difference between DOS and WINDOWS.**

The primary difference between DOS and Windows lies in their design and functionality. DOS (Disk Operating System) is a command-line interface operating system with limited multitasking capabilities and reliance on text-based commands. In contrast, Windows is a graphical user interface-based operating system that supports multitasking, enhanced user interaction, and a wide range of applications, making it more user-friendly and versatile.

**5.** **Explain Booting Process.**

The booting process is the sequence of events that occurs when a computer is powered on or restarted. It begins with the BIOS or UEFI firmware initializing hardware components and performing a Power-On Self-Test (POST) to ensure everything is functioning correctly. Once POST is complete, the system searches for a bootable device (hard drive, SSD, etc.), loads the bootloader, and transfers control to the operating system, which then completes the loading process and presents the user interface.

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_ Signature of faculty in-charge**