**Batch:A-3 Roll No.: 16010122104**

**Experiment No. 01**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **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:

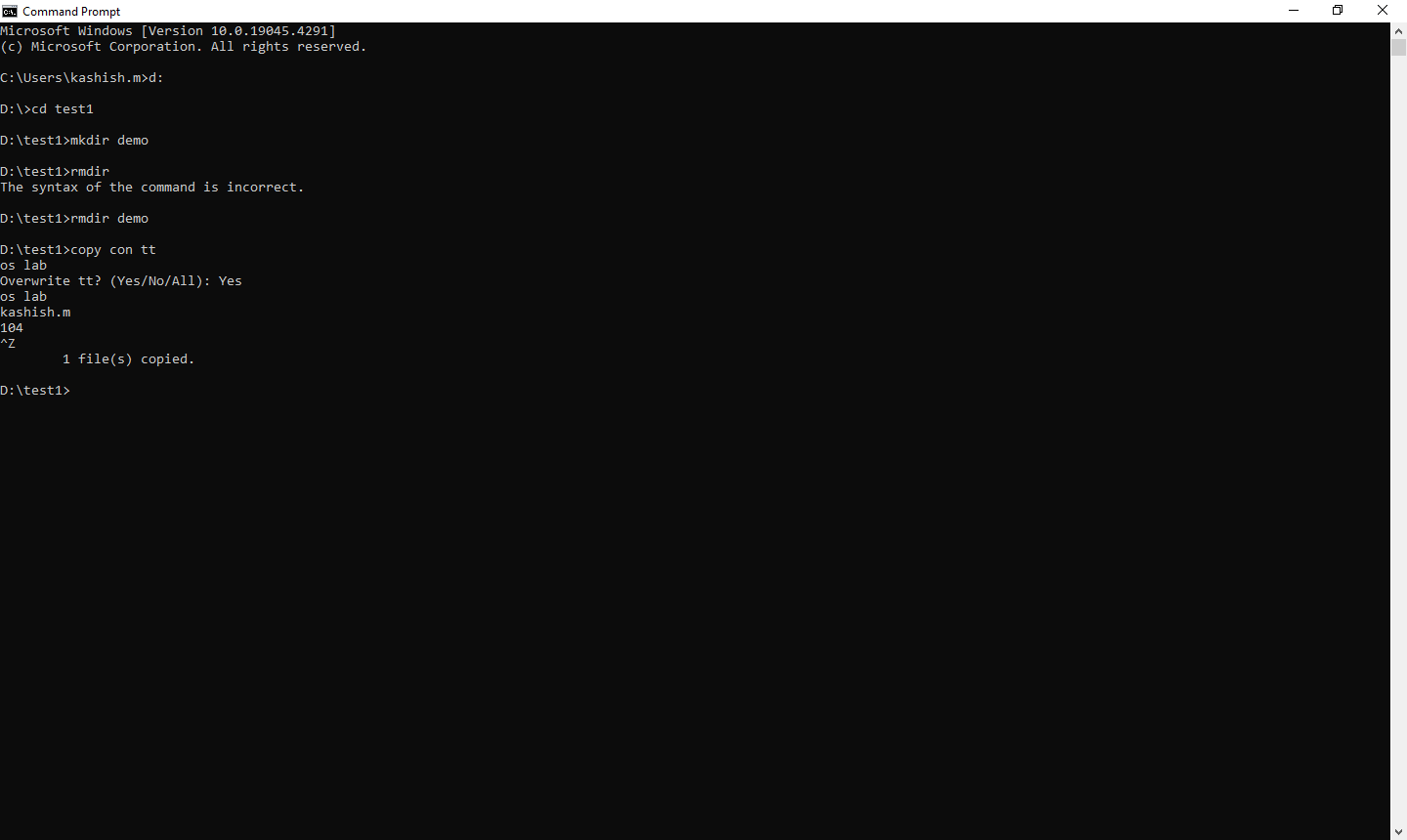
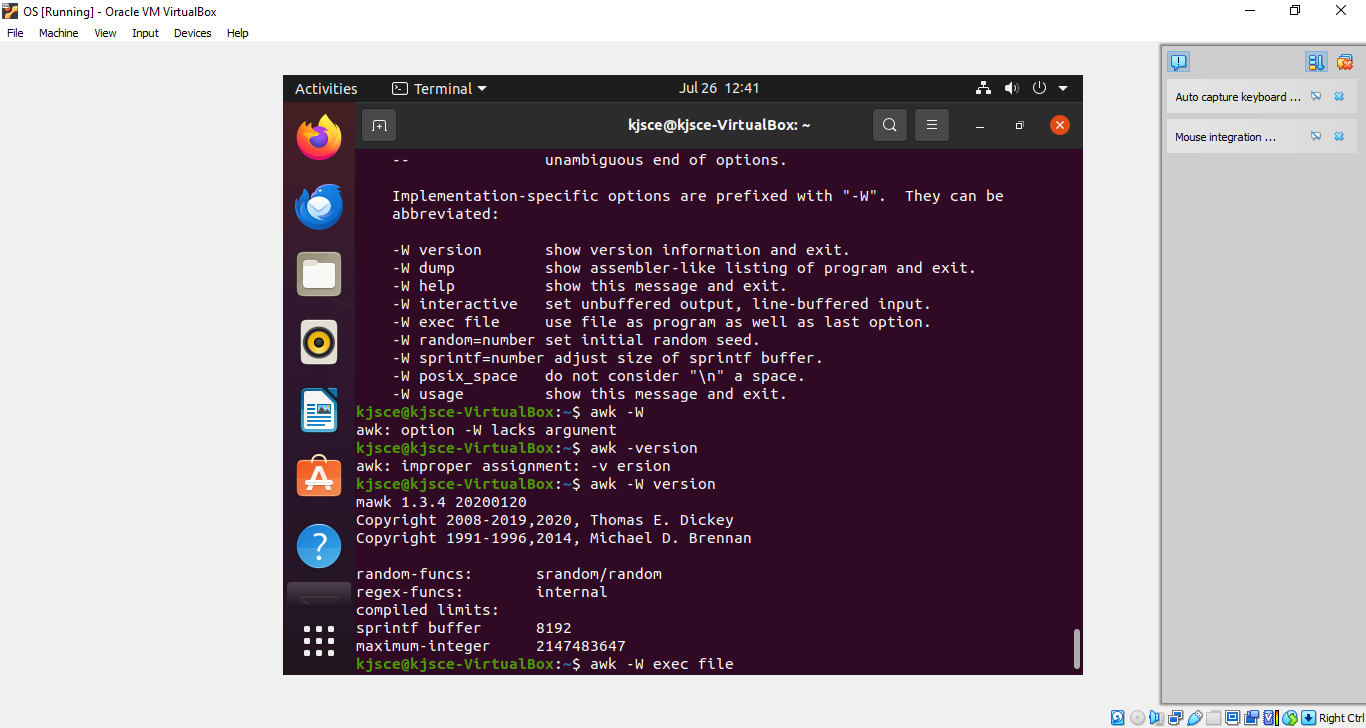
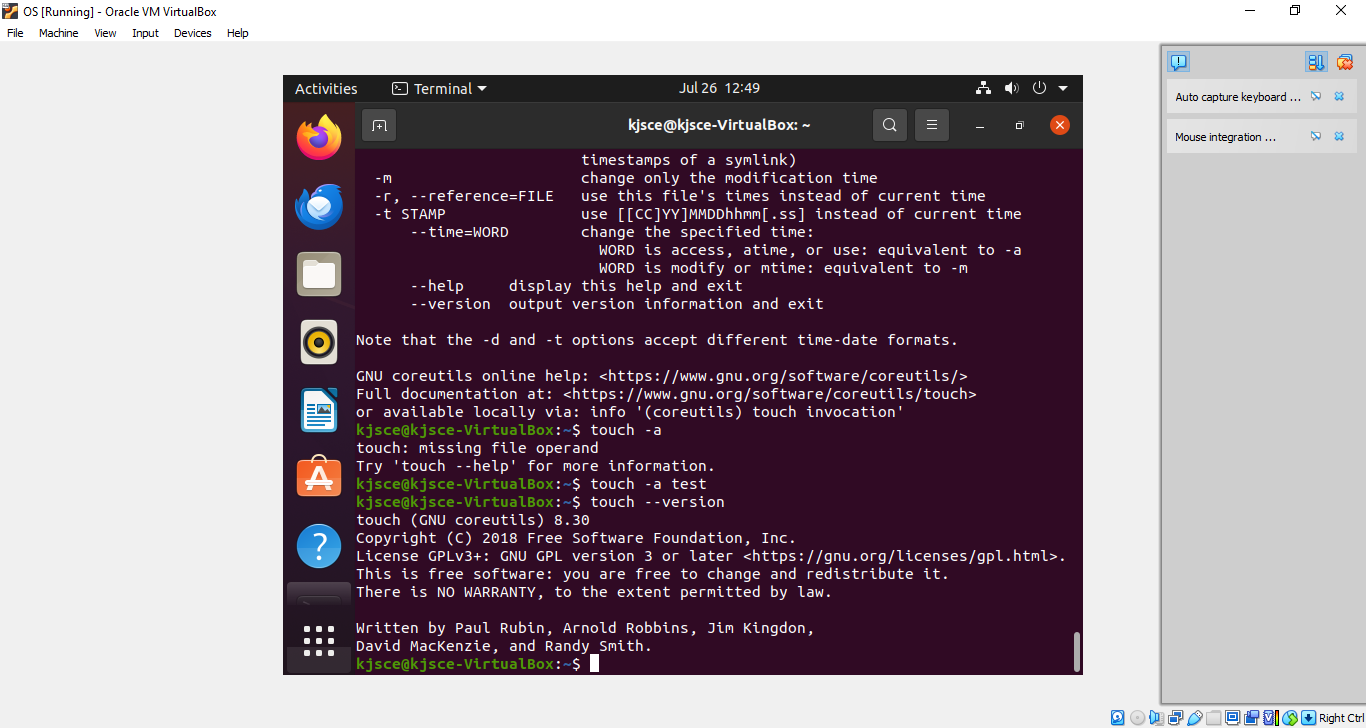
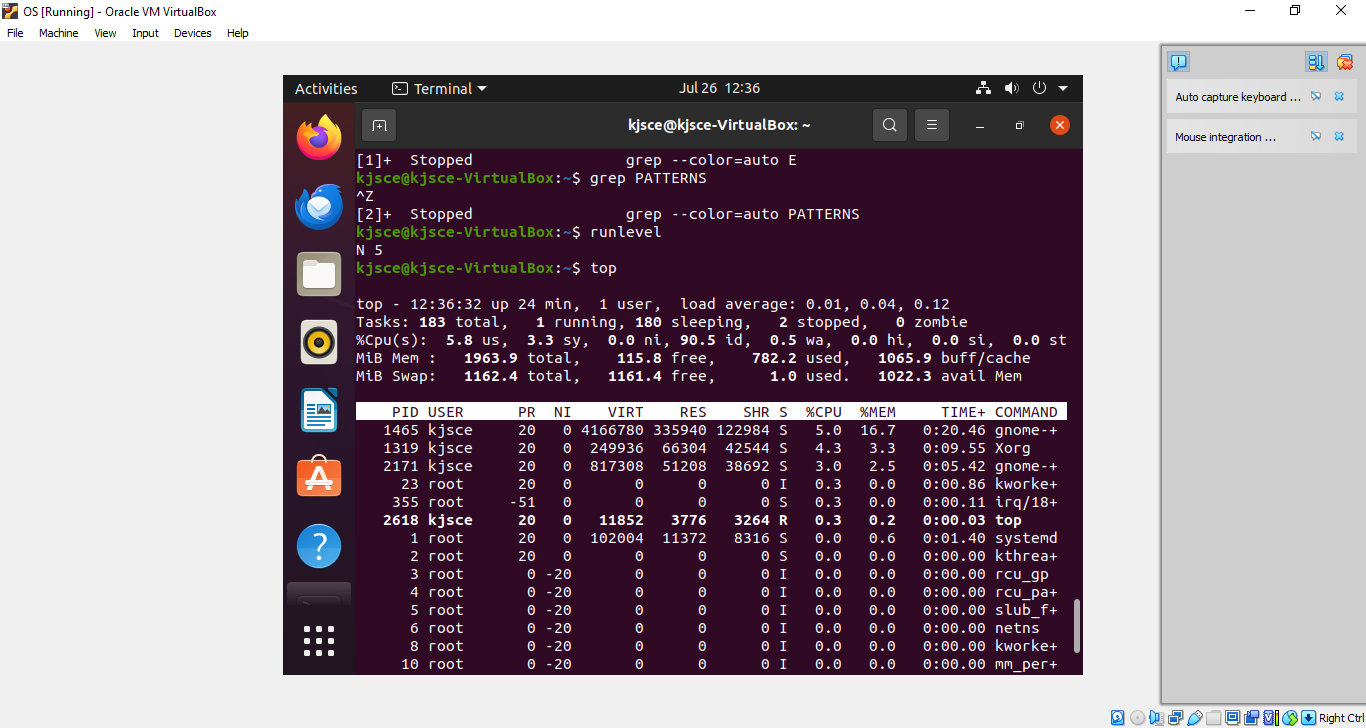
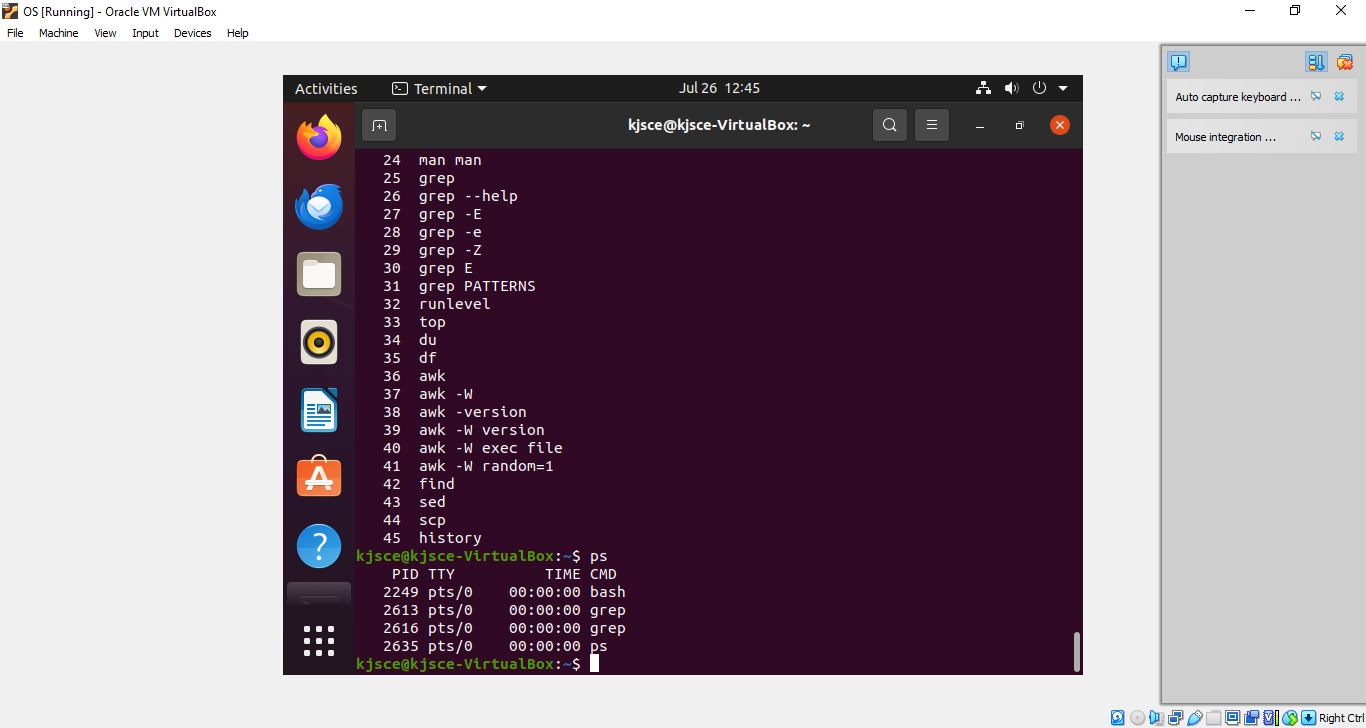
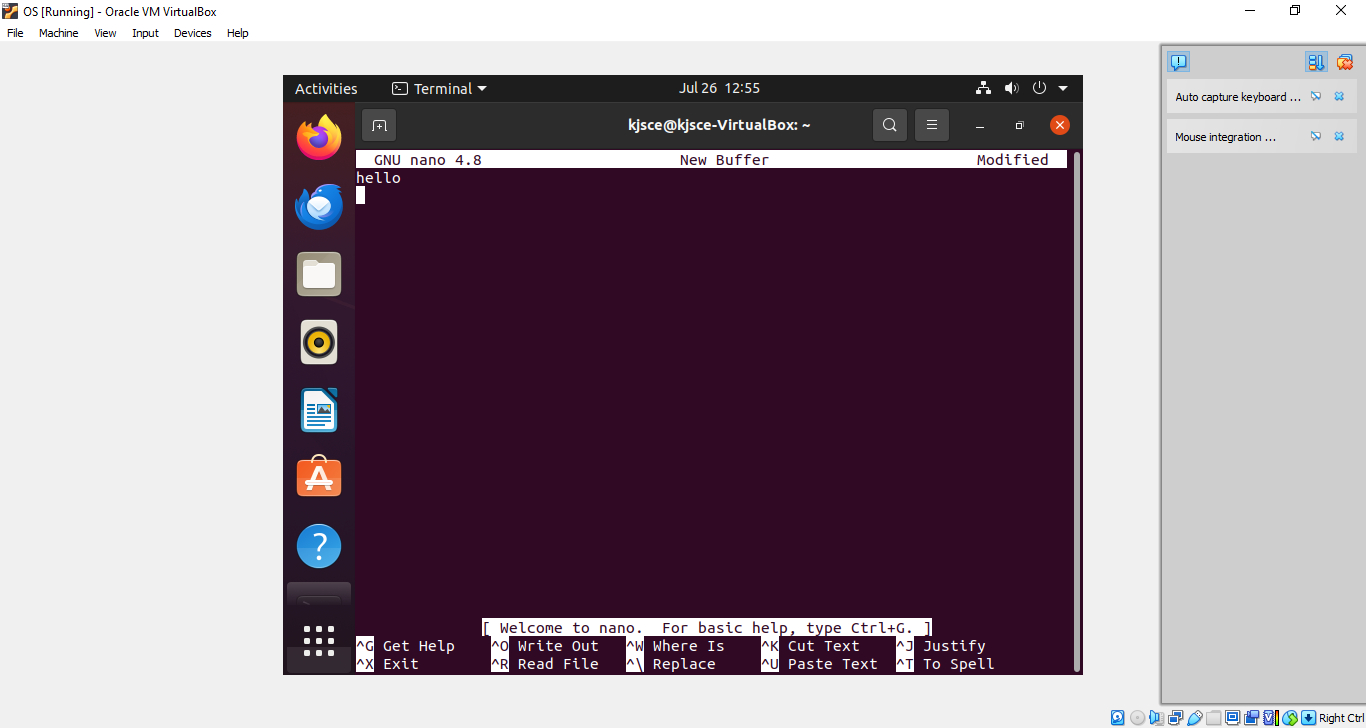
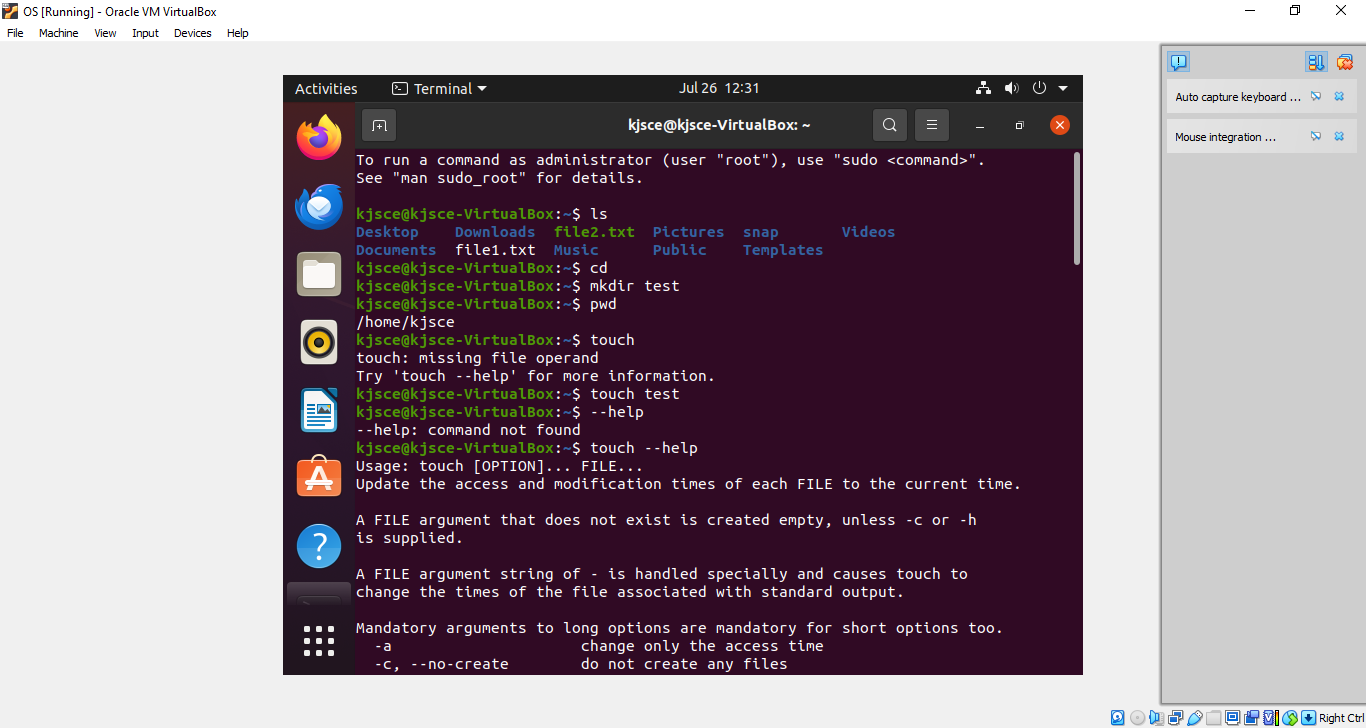
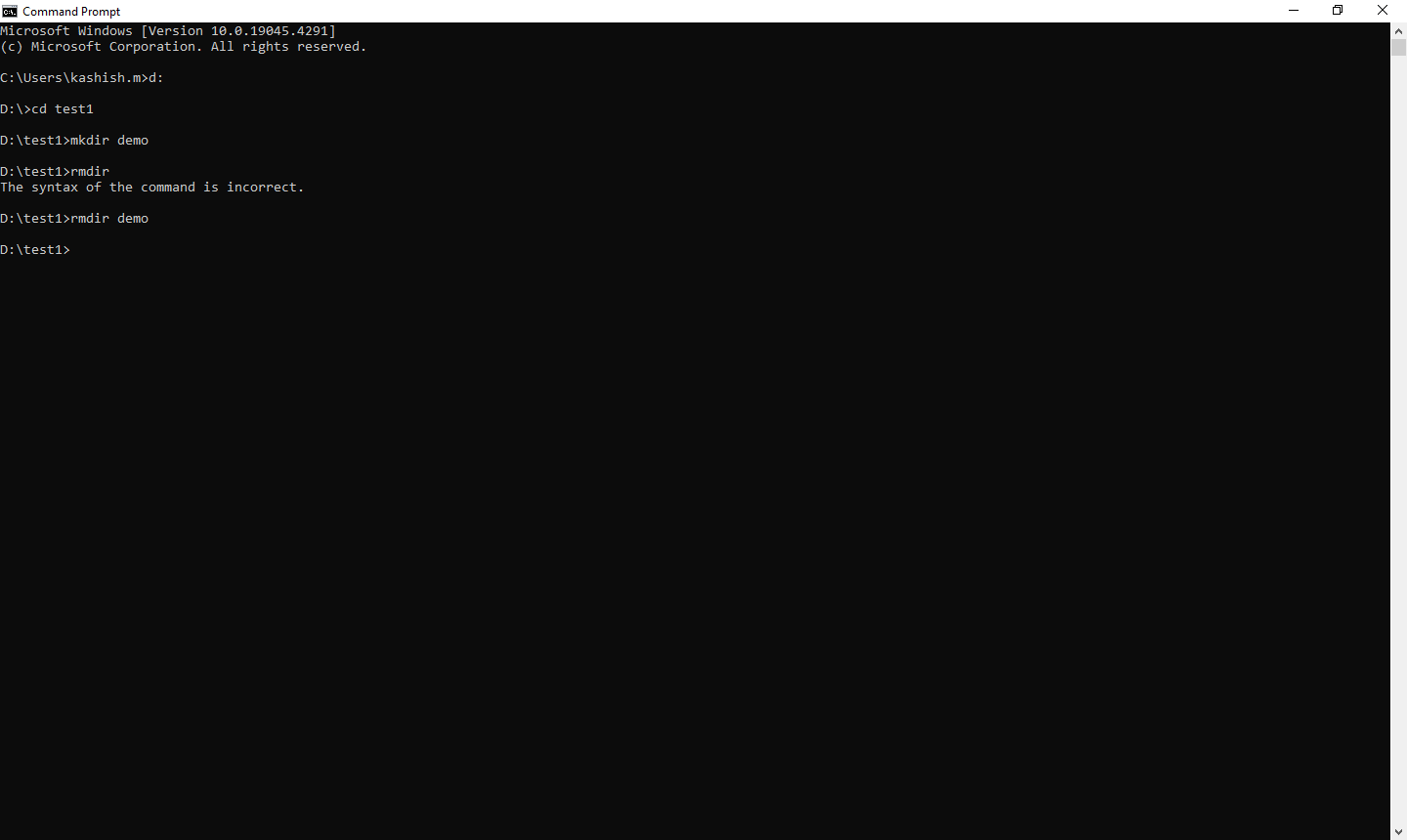
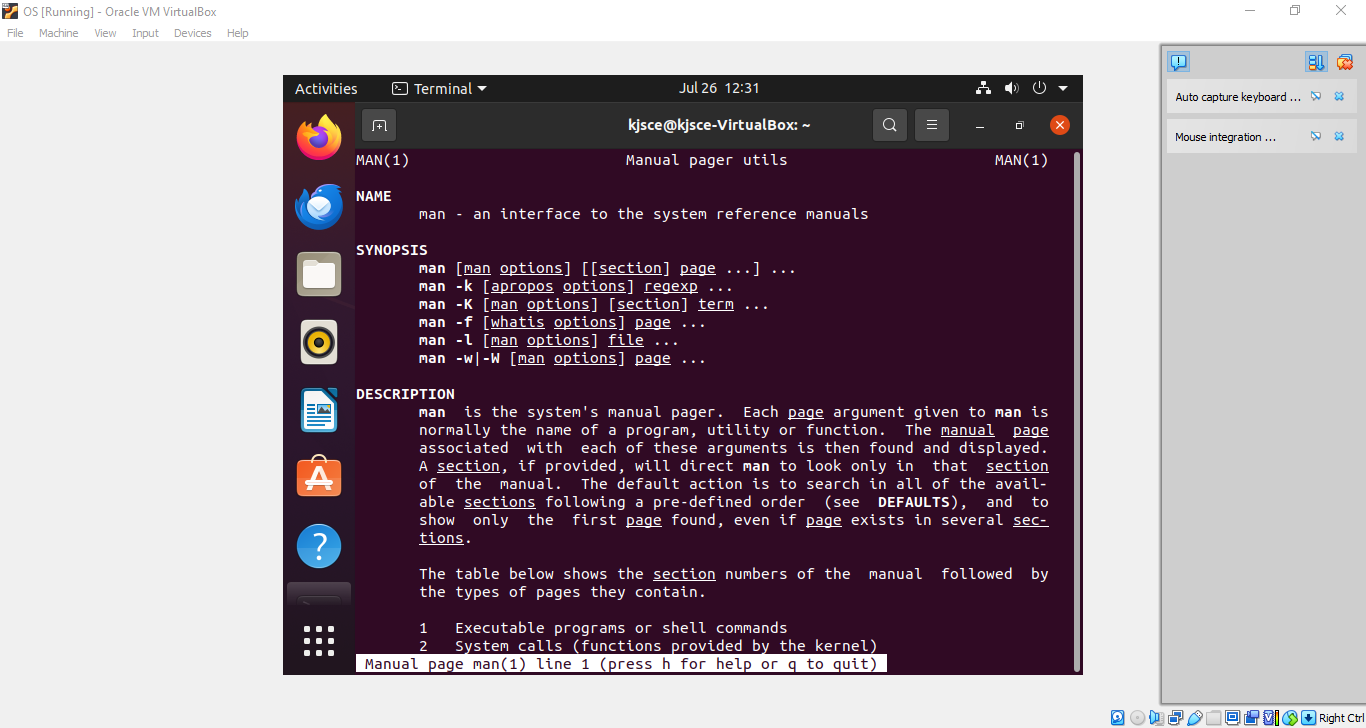
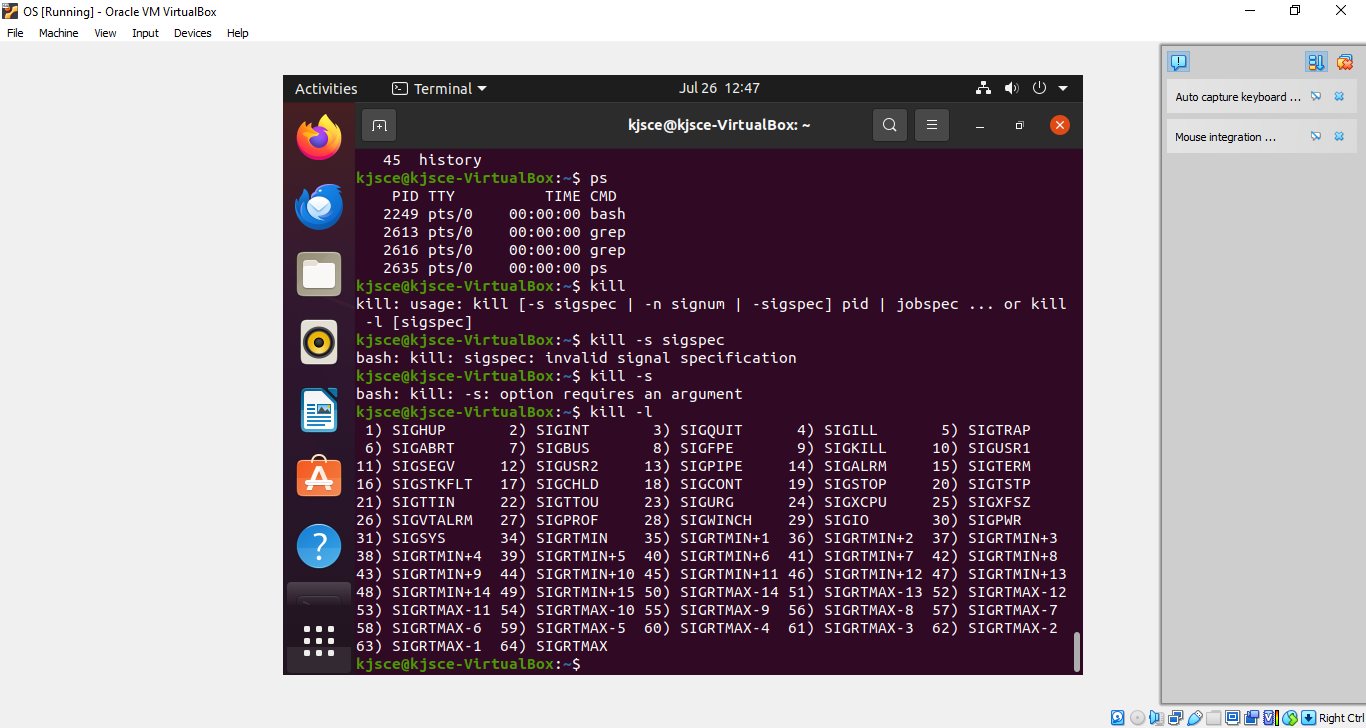
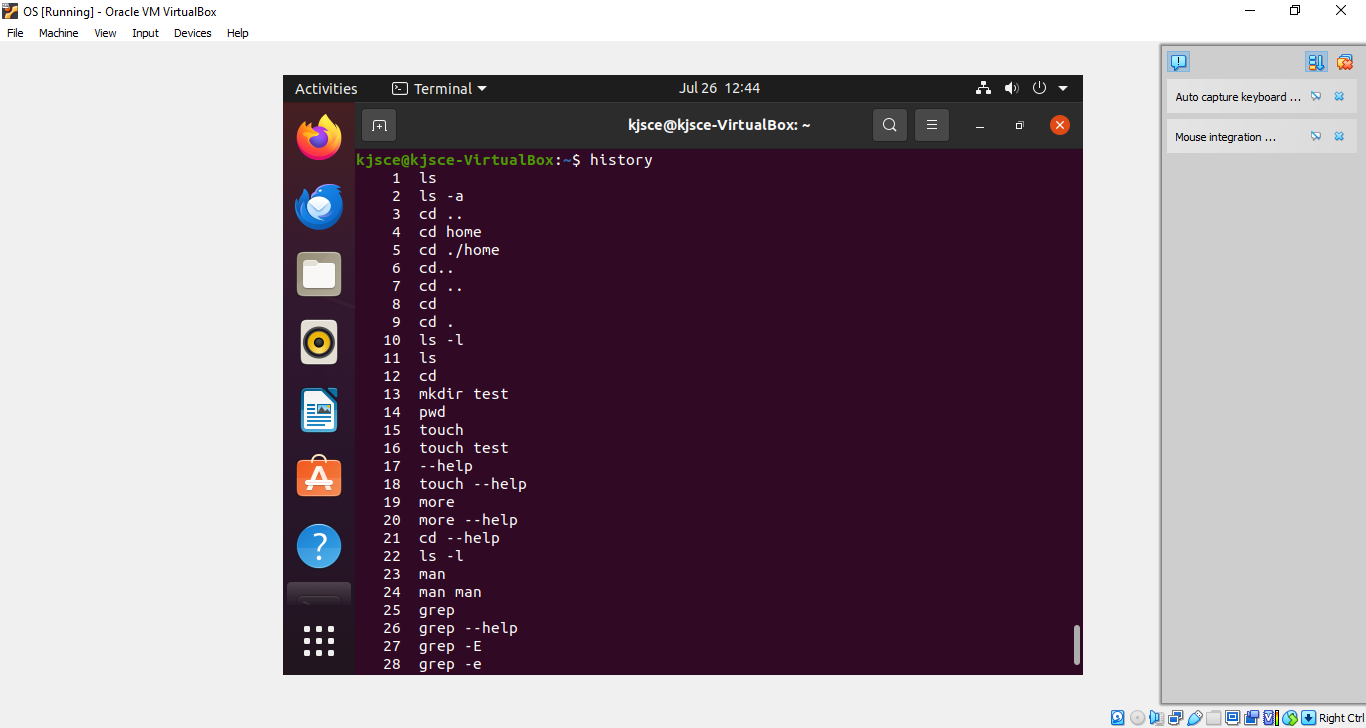
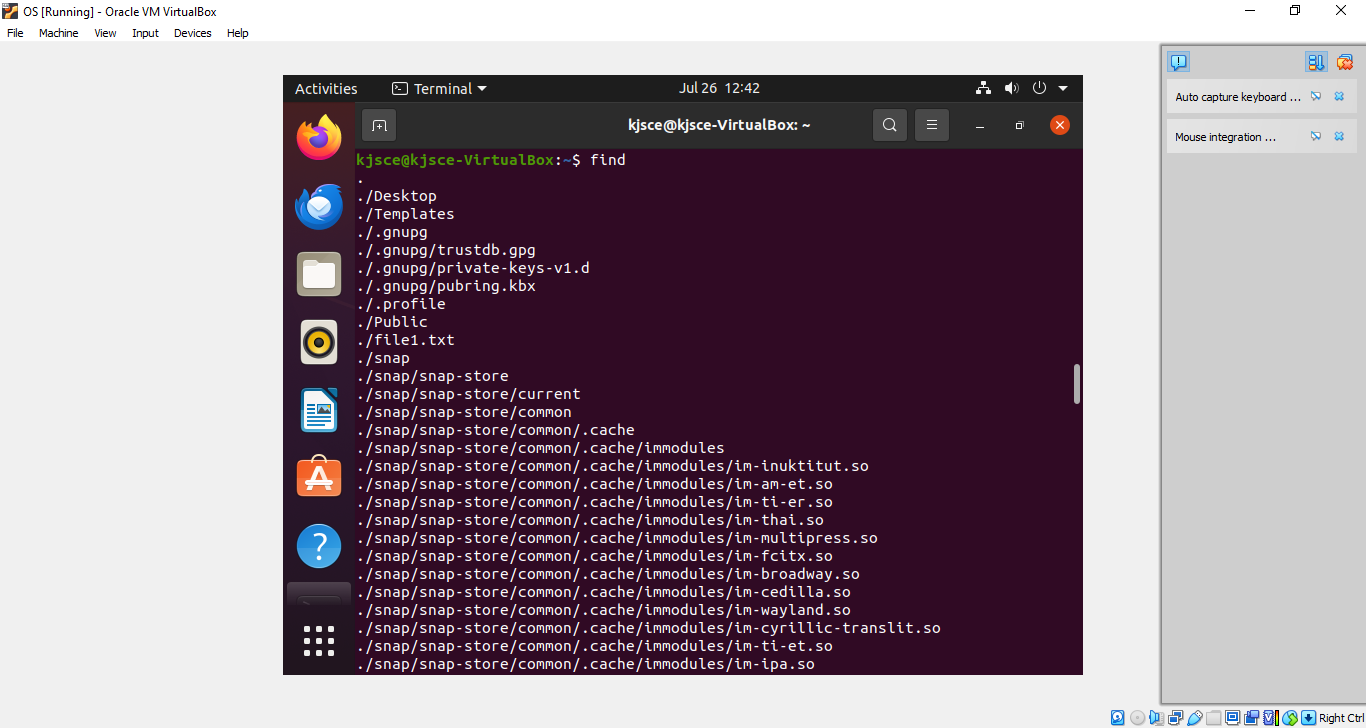
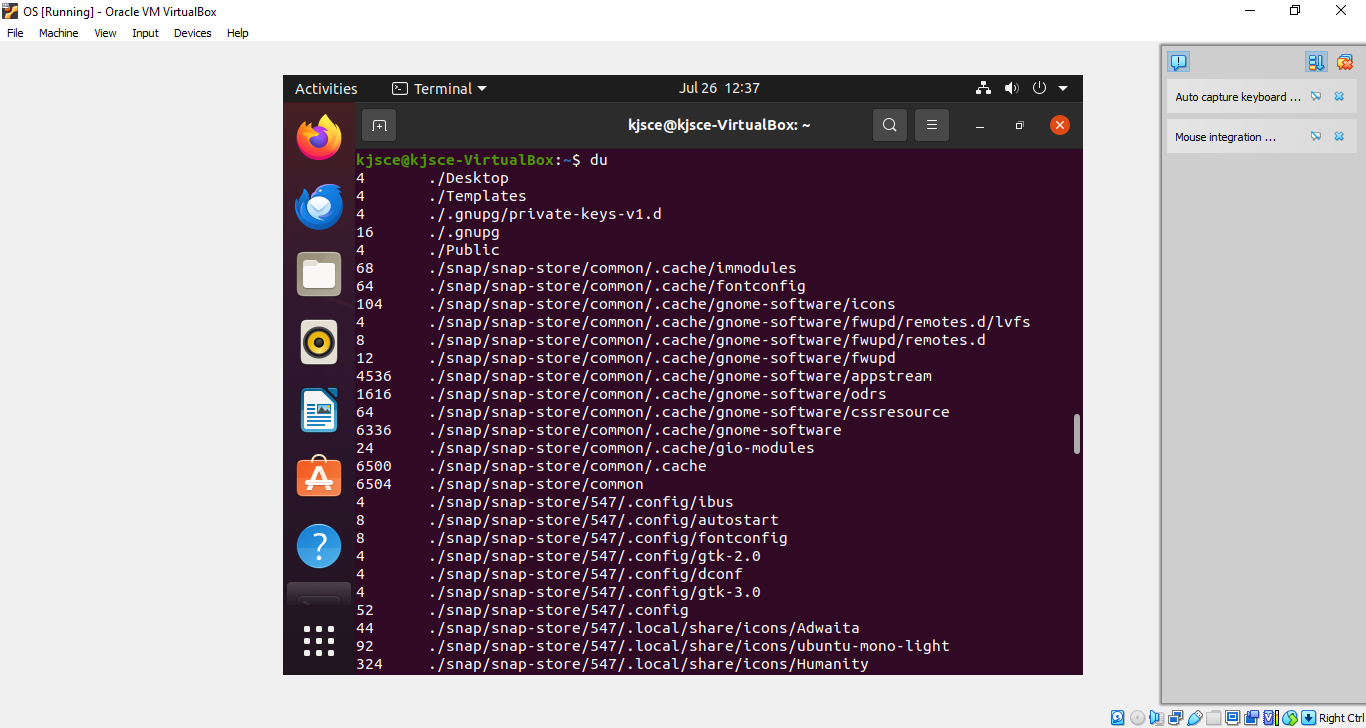
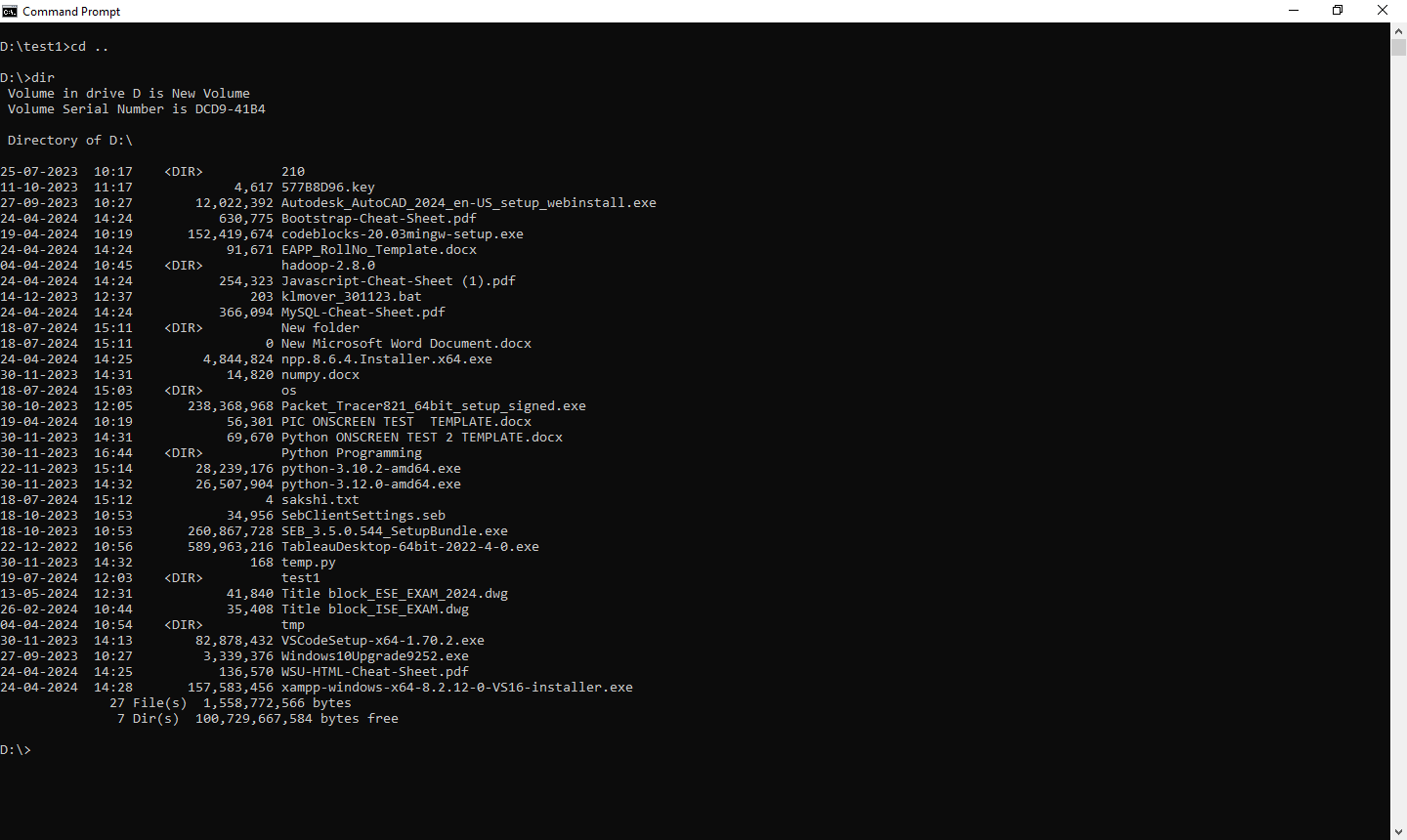
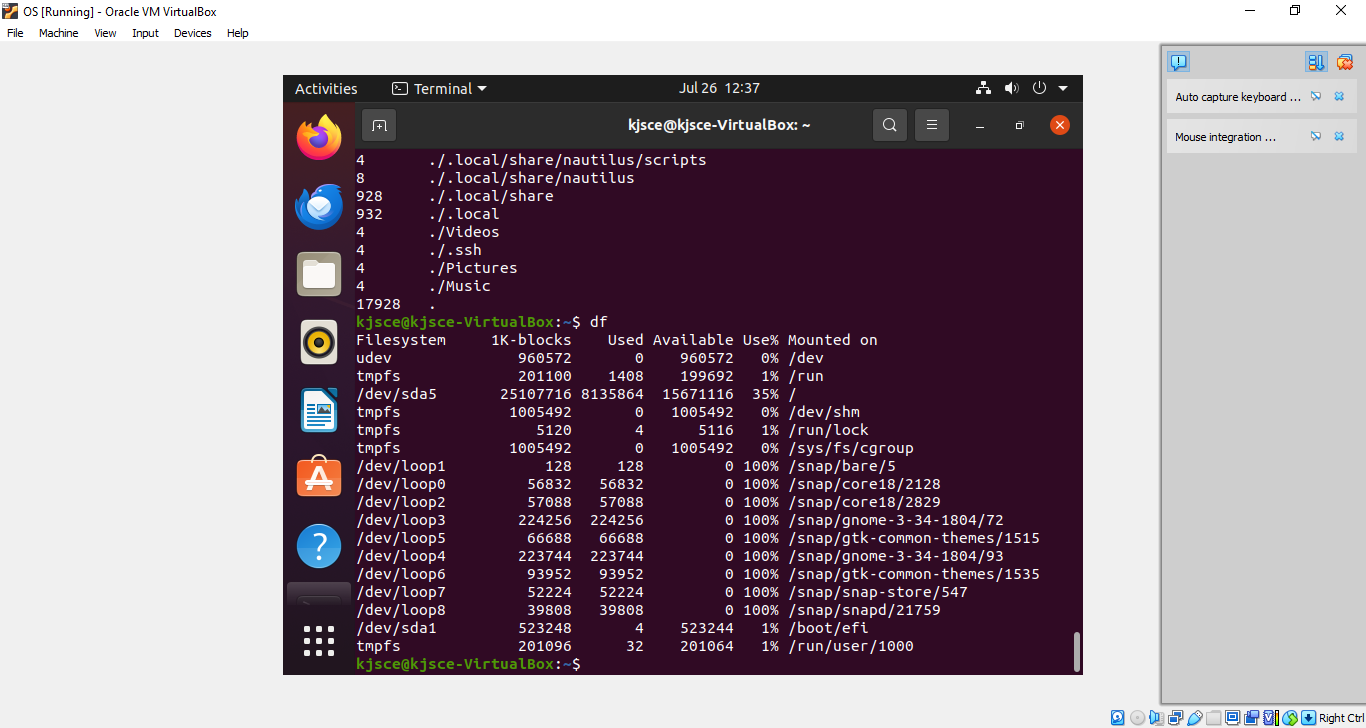
**DOS commands:** Attrib, dir, at, chkdsk, shutdown, tree, create a batch file, output and input redirection

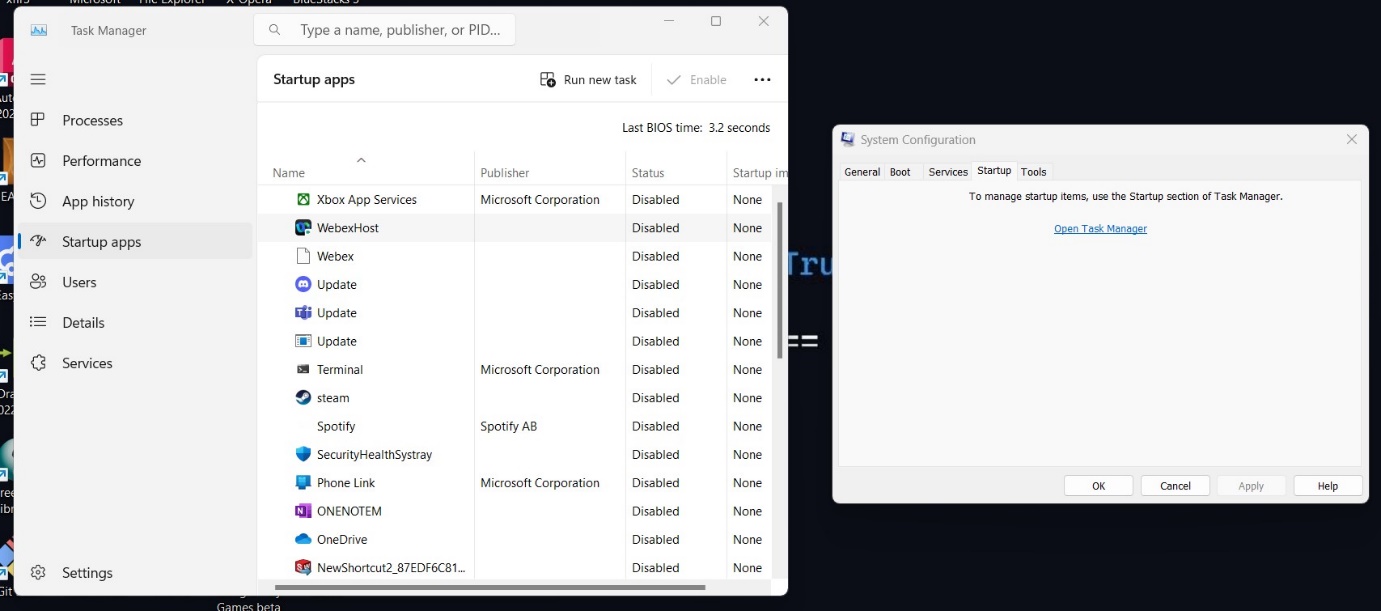
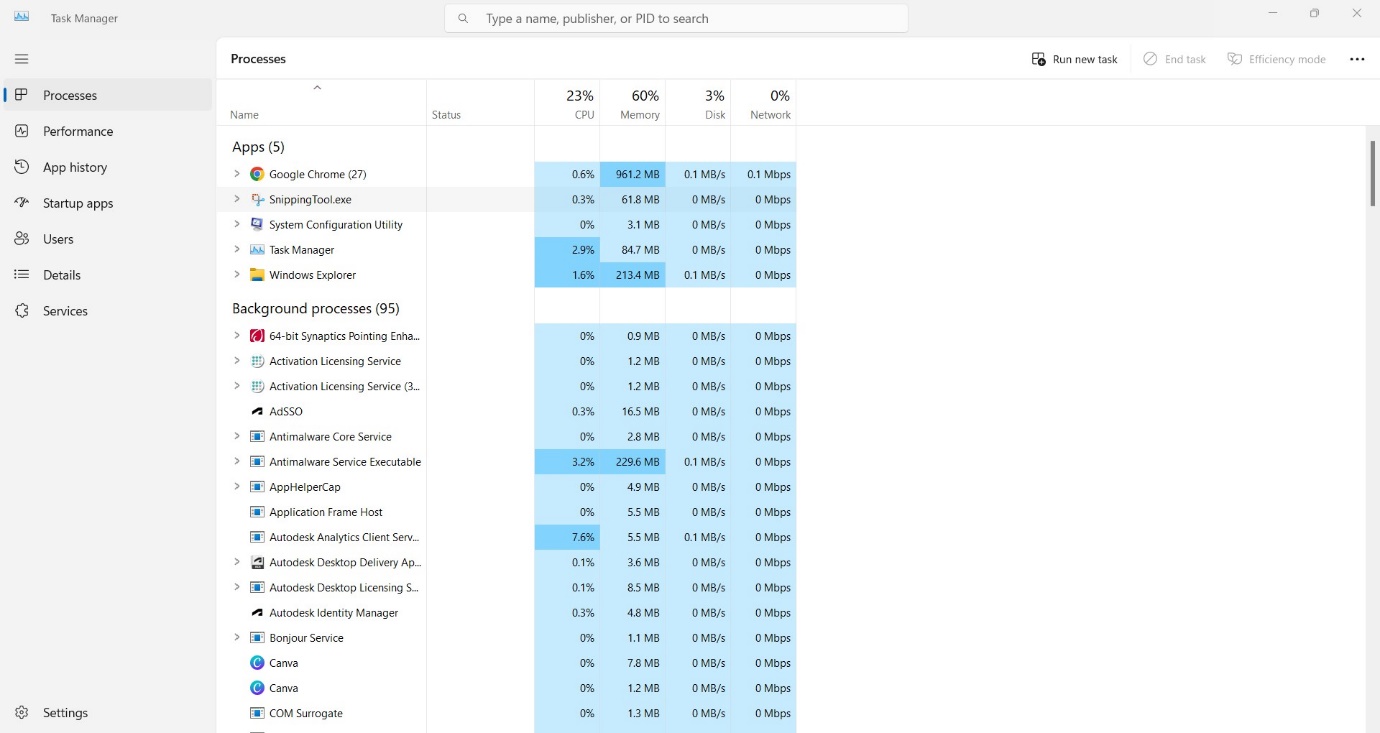
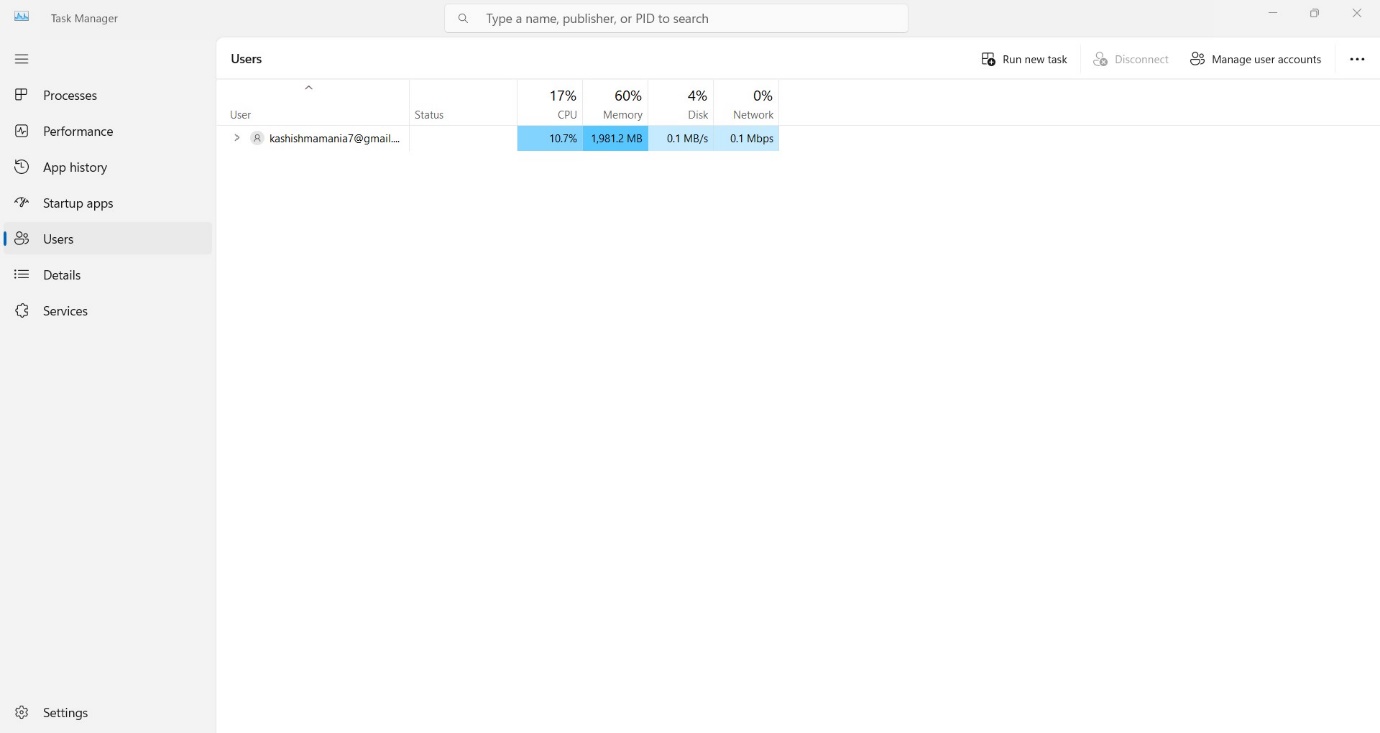
**Windows utilities**: msconfig, defragmenter, performance monitor, task manager, registry editor, event viewer, process explorer

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:**





**Conclusion:**

**Explored the basic Commands of UNIX: Shell, Processes, Files. Used trial and error method to implement various commands and see what changes were made in the OS.**

**Post Lab Descriptive Questions**

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

When reading and interpreting the syntax of an operating system (OS) command, you generally consider the following components:

* **Command**: The name of the program or utility being executed. For example, ls in Unix/Linux or dir in DOS/Windows.
* **Options/Flags**: These modify the behavior of the command. They are typically preceded by a dash (-) in Unix/Linux (e.g., -l for a long listing format) or a slash (/) in DOS/Windows (e.g., /p to pause output).
* **Arguments**: These are additional inputs provided to the command, often specifying what the command should operate on. For example, a file or directory name.
* **Pipes and Redirection**: Special symbols like |, >, and < are used to chain commands together or redirect input/output.

To interpret a command, start from left to right:

* Identify the command.
* Check for any options/flags and their meaning.
* Look for arguments specifying targets or parameters.
* Note any pipes or redirection symbols that change the flow of data.

1. Explain different functions of the operating systems.

Operating systems have several key functions:

1. **Process Management**: Handles the creation, scheduling, and termination of processes. It manages process synchronization, communication, and deadlock handling.
2. **Memory Management**: Manages the system's memory, including the allocation and deallocation of memory spaces as needed by various applications.
3. **File System Management**: Controls how data is stored, retrieved, and organized on storage devices. It provides a way to create, delete, read, write, and manage files and directories.
4. **Device Management**: Manages hardware devices through device drivers, providing a standard interface between the OS and hardware components.
5. **Security and Access Control**: Ensures system security and data integrity by managing user permissions and enforcing access controls.
6. **User Interface**: Provides an interface (either command-line or graphical) for users to interact with the system.
7. **Networking**: Manages network connections and communication, including network protocols and data transmission.
8. What are the default permissions assigned by Unix for Directory.

In Unix, the default permissions for a newly created directory are typically 755. This is represented as:

* **Owner**: Read (r), write (w), execute (x) — rwx (7)
* **Group**: Read (r), execute (x) — r-x (5)
* **Others**: Read (r), execute (x) — r-x (5)

The notation 755 means:

* Owner has full permissions (7).
* Group and others have read and execute permissions (5).

1. Give the difference between DOS and WINDOWS.

**DOS (Disk Operating System)**:

* Command-line based.
* Single-tasking (can run one program at a time).
* Limited memory management.
* No graphical user interface (GUI).
* Used primarily in the 1980s and early 1990s.

**Windows**:

* GUI-based, with support for a command line (Command Prompt or PowerShell).
* Multi-tasking (can run multiple applications simultaneously).
* Advanced memory management (including virtual memory).
* Supports a wide range of hardware and software.
* Continuously updated with modern features and security.

1. Explain Booting Process.

The booting process refers to the sequence of operations that the computer performs when it is powered on, leading to the loading of the operating system. The steps generally include:

1. **Power-On Self Test (POST)**: The system checks the hardware components (e.g., RAM, CPU, keyboard) to ensure they are working correctly.
2. **Loading the Bootloader**: The BIOS/UEFI firmware locates and loads the bootloader from the storage device.
3. **Bootloader Execution**: The bootloader, such as GRUB or Windows Boot Manager, loads the operating system kernel into memory.
4. **Kernel Initialization**: The kernel initializes the hardware and software components, setting up necessary drivers and services.
5. **System Services and User Space**: System services and background processes start, and the system enters user space, providing the user with a login prompt or GUI interface.

**Date: 31/07/20244 Signature of faculty in-charge**