S.B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT & RESEARCH, NAGPUR

Practical 1
Prelab

Aim: Installation of Linux Operating System.

Name:- Aaditya S Gudhniye

USN: CM24062

Semester / Year: IV Sem / II Year

Academic Session: 2025-26

Date of Performance: 13-01-26

Date of Submission: 20-01-26

❖ Aim: Installation of Linux Operating System.

❖ Objectives:

1. Understand the system requirements and compatibility for Linux OS installation.
2. Learn the step-by-step process to install and configure a Linux distribution.
3. Verify the installation and explore basic Linux commands for system setup.

❖ Requirements:

1. A bootable USB drive or DVD with the desired Linux distribution (e.g., Ubuntu, Fedora, Debian).
2. A computer system with minimum hardware requirements: 2 GB RAM, 20 GB free disk space, and a compatible processor.
3. Internet connection (optional, for updates during installation).
4. Software for creating a bootable USB, like Rufus or Etcher (if needed).
5. Basic knowledge of BIOS/UEFI settings for boot sequence configuration.

*****IN THIS PRACTICAL WE'LL BE INSTALLING UBUNTU*****

❖ Prerequisite:

Linux is an open-source operating system widely used for personal, professional, and server environments. Its flexibility, security, and community-driven development make it a popular choice for users. The installation of a Linux OS involves creating a bootable medium, setting up the system to boot from the medium, and following the installation wizard to partition the disk and configure system settings. Common distributions like Ubuntu, Fedora, and Debian offer user-friendly interfaces for easy installation. The process may include creating swap space, selecting a file system like ext4, and setting up user accounts. Post-installation tasks involve updating the system, installing necessary drivers, and customizing the environment. Understanding the installation process ensures better control over system performance and resource allocation, making Linux a powerful tool for both beginners and advanced users.

Steps to Make a Pendrive Bootable Using Rufus:

1. **Download and Open Rufus**: Download Rufus from its official website, install it, and launch the application.
2. **Insert Pendrive and Select ISO**: Connect the USB pendrive to your system. In Rufus, select your pendrive under "Device" and click "SELECT" to choose the Linux ISO file.
3. **Set Partition Scheme and File System**: Choose "GPT" for UEFI or "MBR" for BIOS under "Partition scheme," and ensure the file system is set to "FAT32."
4. **Start the Process**: Click "START," confirm the warning about data deletion, and wait for Rufus to create the bootable USB. Once done, eject the pendrive safely.

❖ Theory:

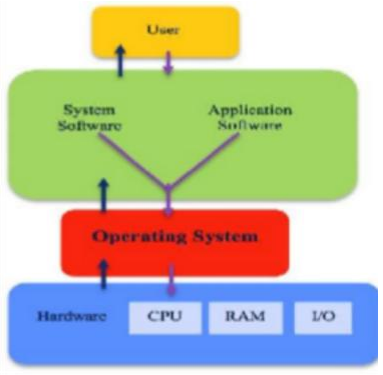
An operating system (OS) is system software that acts as an interface between computer hardware and the user. It manages hardware resources and provides essential services for executing application software. The OS is a vital component of a computer system, enabling users to interact with the system in a structured and efficient manner.

Functions of an Operating System:

1. **Process Management**: Handles the execution of multiple processes, ensuring smooth multitasking and optimal CPU usage.
2. **Memory Management**: Allocates and manages system memory, ensuring that each program gets the required memory without interfering with others.
3. **File System Management**: Organizes and manages data storage, allowing users to store, retrieve, and manipulate files.
4. **Device Management**: Coordinates and controls input/output devices such as keyboards, printers, and disk drives.
5. **Security and Access Control**: Protects data and system integrity by implementing user authentication and access restrictions.

Types of Operating Systems:

1. **Batch OS**: Executes tasks in batches without direct user interaction.
2. **Time-Sharing OS**: Allows multiple users to share system resources simultaneously.
3. **Distributed OS**: Manages a group of independent computers and makes them appear as a single system.
4. **Real-Time OS**: Designed for time-critical tasks where responses are needed within strict deadlines.
5. **Mobile OS**: Specialized for mobile devices, such as Android and iOS.

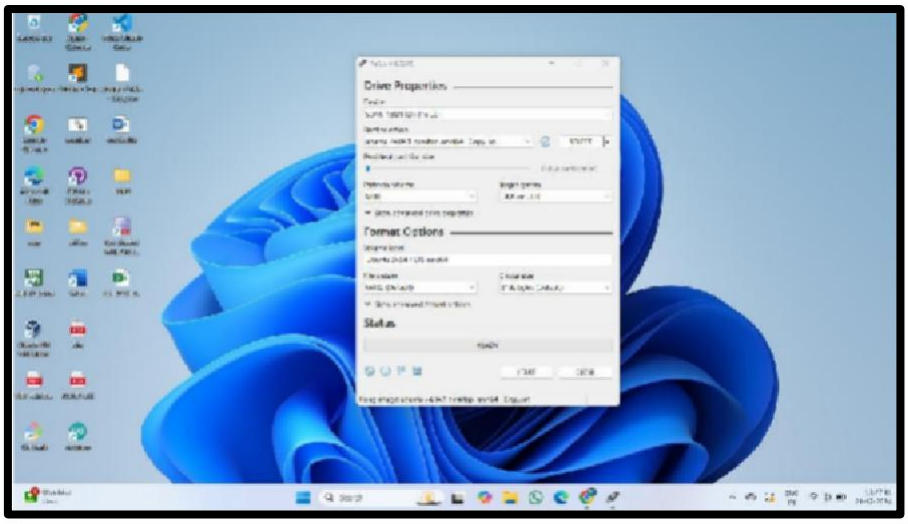


Operating systems like Windows, Linux, and macOS are widely used across various platforms, each catering to specific user needs. Linux, for example, is known for its open-source nature and flexibility, while Windows provides a user-friendly interface for personal and professional use.

In summary, an operating system is the backbone of computer functionality, ensuring efficient resource management, user interaction, and system reliability. It continues to evolve, adapting to technological advancements and diverse user requirements.

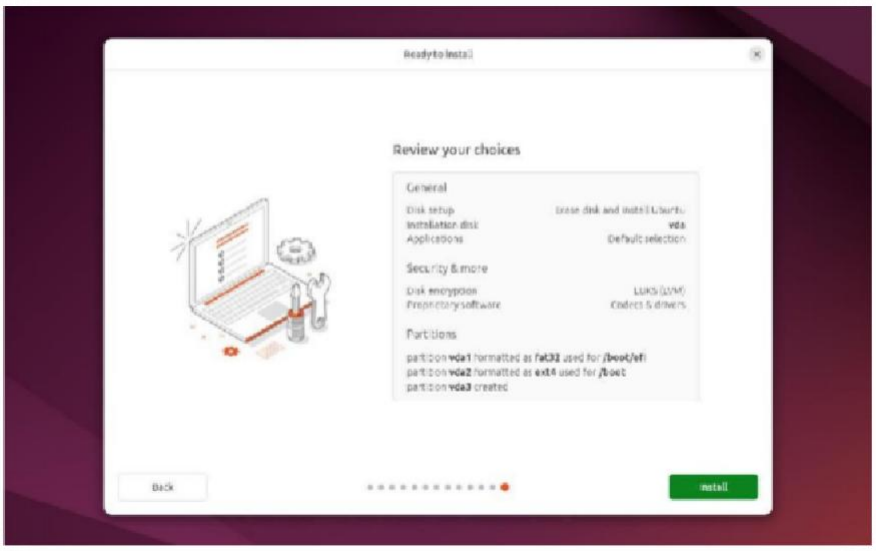
❖ Steps to Install Linux Operating System:

1. **Prepare Bootable Media**: Use a tool like Rufus to create a bootable USB drive or DVD with the Linux distribution ISO.



2. **Configure BIOS/UEFI Settings**: Restart your computer and access the BIOS/UEFI settings (usually by pressing a key like F2, F10, or DEL during boot). Set the boot priority to USB or DVD.

3. **Boot from Media**: Insert the bootable USB or DVD and restart the computer. The system will boot into the Linux installer.
4. **Choose Installation Option**: Select "Install Linux" or a similar option from the menu. Some distributions may allow you to try the OS before installation.
5. **Partition the Disk**:
   - Select the partition scheme (automatic or manual).
   - Create required partitions (e.g., root `/`, swap, and optionally `/home`).
6. **Set Up User Details**: Enter your username, password, and system name.
7. **Select Time Zone**: Choose your location to configure the correct time and date settings.
8. **Begin Installation**: Review the settings and click "Install." The process will take a few minutes to complete.



9. **Remove Bootable Media**: Once installation is finished, remove the USB or DVD when prompted and restart the system.

10. **Post-Installation Configuration**: Log in to your new Linux system, update packages, and install additional software if needed.

    Commands to update:

Command	Use
sudo apt update	Fetches the latest information about available packages and versions.
sudo apt upgrade	Installs the latest versions of all currently installed packages.
sudo apt full-upgrade	Upgrades packages, adding or removing dependencies as required.
sudo apt autoremove	Removes unnecessary packages no longer needed as dependencies.
sudo reboot	Restarts the system to apply critical updates if required.

❖ **Conclusion:** The installation of the Linux operating system provides a hands-on understanding of system setup and configuration. It enables efficient utilization of resources and customization to suit user needs. Mastering this process builds a solid foundation for exploring advanced system administration tasks.

❖ Discussion Questions:

1. **What is an operating system, and why is it important?**
2. **What is the purpose of creating a bootable USB, and how is it done?**
3. **Can you explain the difference between apt update and apt upgrade in Ubuntu?**
4. **Why is partitioning necessary during OS installation, and what are the common partitions used?**
5. **What steps should you follow after successfully installing a Linux OS?**

❖ References:

https://ubuntu.com/tutorials/install-ubuntu-desktop#1-overview
*https://youtu.be/wjbbl0TTMeo?si=32l6h8VbcmU-euD_*
https://answers.microsoft.com/
https://rufus.ie/en/

Date:20/01/2026

Signature
Course Coordinator
B.Tech CSE(AIML)
Sem: 4 / 2025-26

S.B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT & RESEARCH, NAGPUR

Practical 02

Aim: To understand and demonstrate the use of basic commands in different operating systems (Windows, Linux, and UNIX) for managing files, directories, permissions, and user interactions through a terminal or command-line interface.

Name: Aaditya S Gudhniye

USN: CM24062

Semester / Year: IV Sem / II Year

Academic Session: 2025-26

Date of Performance: 20-01-26

Date of Submission: 27-01-26

❖ **Aim:** To understand and demonstrate the use of basic commands in different operating systems (Windows, Linux, and UNIX) for managing files, directories, permissions, and user interactions through a terminal or command-line interface.

❖ Objectives:

1. To learn and practice fundamental command-line operations for file and directory management.
2. To explore and utilize user and permission management commands effectively.
3. To enhance system administration skills by working with commands across different operating systems.

❖ Requirements:

Hardware Requirements:

- **Processor**: Multi-core CPU, Intel Core i3 (3.0 GHz) or higher
- **RAM**: Minimum 4 GB (8 GB recommended for optimal performance)
- **Storage**: 100 GB HDD or SSD (Solid State Drive) for faster access
- **Network Interface**: Ethernet or Wi-Fi adapter for connectivity

Software Requirements:

- **Operating System**: Windows 10/11, Linux (Ubuntu 20.04/CentOS 8), UNIX-based OS
- **Command-line Interface**: PowerShell or Command Prompt (Windows), Terminal (Linux/UNIX)
- **Text Editor**: Nano, Vim, or Visual Studio Code for file editing
- **Administrative Privileges**: Superuser (Linux/UNIX) or Administrator (Windows) access

❖ Theory:

Command-line interfaces (CLI) are fundamental tools used in system administration to interact with operating systems efficiently. They enable users to execute tasks by entering text-based commands rather than relying on graphical interfaces. Using CLI commands, administrators can manage files and directories, modify permissions, monitor system performance, and configure system resources. Operating systems such as Windows, Linux, and UNIX offer numerous built-in commands to simplify administrative operations. Commands like `ls`, `cd`, `pwd`, `mkdir`, and `chmod` assist in navigating and organizing the file system. Understanding command structure and syntax ensures accurate execution of tasks. CLI usage helps in performing tasks faster and with greater control. It also supports automation through scripts, improving productivity and consistency. Gaining proficiency in basic commands enhances system management, security, and troubleshooting skills. This practical aims to build a strong foundation in using command-line tools effectively.

❖ **Commands:**

1. **Display User Manual of a Command**
   - Functionality: Shows the manual page with details about a command's usage, options, and arguments.
   - Syntax: man <command>
   - Example: man ls

2. **Change Current Working Directory**.
   - Functionality: Changes the terminal's current working directory.
   - Syntax: cd <directory-path>
   - Example: cd /home/user/Documents.

3. **List Contents of the Current Directory.**
   - Functionality: Lists all files and directories in the current location.
   - Syntax: ls
   - Example: ls

4. **Read/Modify/Concatenate Text Files.**
   - Functionality: Displays or manipulates file content.
   - Syntax:
     - Read: cat <filename>
     - Modify: 'nano <filename>
     - Concatenate: cat <file1> <file2> > <outputfile>

5. **Create a New Directory**.
   - Functionality: Creates a new directory at the specified path.
   - Syntax: mkdir <directory-name
   - Example: mkdir newdir

6. **Display Current Working Directory.**
   - Functionality: Prints the current directory path.
   - Syntax: pwd
   - Example: pwd

7. **Write Arguments to Standard Output.**
   - Functionality: Prints the provided string or variables.
   - Syntax: echo <arguments>
   - Example: echo Hello World

8. **Remove a File**.
   - Functionality: Deletes a specified file.
   - Syntax: rm <filename>
   - Example: rm file.txt

9. **Delete a Directory.**
   - Functionality: Removes an empty directory.
   - Syntax: rmdir <directory-name>
   - Example: rmdir olddir

10. **Copy a File or Directory.**
   - Functionality: Copies a file or directory to a destination.
   - Syntax: cp <source> <destination>
   - Example: cp file.txt backup/

11. **Switch to Root User.**
   - Functionality: Gains root privileges temporarily.
   - Syntax: sudo su
   - Example: sudo s

12. **Move Files or Directories.**
   - Functionality: Moves or renames files and directories.
   -Syntax: mv <source> <destination>
   - Example: mv file.txt newdir/

13. **Search for a String in a File.**
   - Functionality: Searches for a specific word or pattern in a file.
   - Syntax: grep "<string>" <file>
   - Example: grep "error" log.txt

14. **Print Top N Lines of a File.**
   - Functionality: Displays the first N lines of a file.
   - Syntax: head -n <N> <file>
   - Example: ‘head -n 10 file.txt’

15. **Print Last N Lines of a File.**
   - Functionality: Displays the last N lines of a file.
   - Syntax: tail -n <N> <file>
   - Example: ‘tail -n 10 file.txt

16. **Remove Read Permission from Owner.**
    - Functionality: Revokes the owner's read permission for a file.
    - Syntax: chmod u-r <filename>
    - Example: chmod u-r file.txt

17. **Change Specific Permissions.**
    - Functionality: Sets or removes specific file permissions.
    -Syntax: chmod u+r,w-x,g+w <filename>
    - Example: chmod u+r,w-x,g+w file.txt

18. **Add Write Permission to Owner, None to Others**.
    -Functionality: Allows write access for the owner only.
    - Syntax: chmod u+w,o-rwx <filename>
    - Example: chmod u+w,o-rwx file.txt

19. **Assign Permissions to Users.**
    - Functionality: Modifies file access for users, groups, and others.
    - Syntax: chmod u+rwx,g+rx,o+r <filename>
    - Example: 'chmod u+rwx,g+rx,o+r file.txt

20. **Assign R/W/X to Others.**
    - Functionality: Gives read, write, and execute permissions to others.
    - Syntax: chmod o+rwx <filename>
    - Example: chmod o+rwx file.txt

21. **Remove All Permissions from All Users.**
    - Functionality: Clears all permissions on a file.
    - Syntax: 'chmod a-rwx <filename>
    - Example: 'chmod a-rwx file.txt

22. **Remove Read Permission Using Absolute Mode.**
    - Functionality: Uses numeric mode to restrict read access.
    - Syntax: chmod 700 <filename>
    - Example: chmod 700 file.txt

23. **Set R/W for Owner, None for Group/Other.**
    - Functionality: Assigns permissions in numeric mode.
    - Syntax: chmod 600 <filename>
    - Example: chmod 600 file.txt'

24. **Add Execute for Owner, Read for Group/Others.**
    - Functionality: Adds execution and read access.
    - Syntax: chmod u+x,g+r,o+r <filename>

- Example: chmod u+x,g+r,o+r file.txt

25. **Add Execute Permission to All Users.**
    - Functionality: Enables execution by everyone.
    - Syntax: chmod a+x <filename>
    - Example: chmod a+x script.sh

❖ **Conclusion:** In conclusion, understanding and using essential operating system commands like **'ls', 'cd', 'cp', 'mv', and 'chmod'** enables efficient file management, navigation, and permission control. Tools like **'grep', 'head', and 'tail'** enhance data processing. Mastery of these commands improves system administration, task automation, and overall system security and performance.

❖ **Discussion Questions:**

1. **What is the significance of the pwd command in a Linux environment?**
2. **-Explain the function of the cp command and its common options.**
3. **How does chmod 700 affect file permissions, and what does each digit represent?**
4. **Describe the difference between head and tail commands in Linux.**
5. **What is the purpose of the grep command, and how is it used with regular expressions?**

❖ **References**:

https://ubuntu.com/tutorials/command-line-for-beginners#1-overview
https://www.geeksforgeeks.org/25-basic-ubuntu-commands/

Date: 27/01/2026

Signature
Course Coordinator
B.Tech CSE(AIML)
Sem: 4 / 2025-26