**Linux Commands**

**Linux** is an open-source operating system (OS) based on the Unix architecture. It was created by **Linus Torvalds** in 1991 and has since become one of the most widely used operating systems in the world. Linux is known for its stability, security, and flexibility, making it a popular choice for everything from personal computers to servers, supercomputers, and embedded systems.

**Key Features of Linux:**

1. **Open Source**: Linux is distributed under open-source licenses (like the GNU General Public License), meaning its source code is freely available for anyone to view, modify, and distribute.
2. **Customizability**: Users can modify the OS to suit their needs, from the kernel (core of the OS) to the user interface.
3. **Stability and Reliability**: Linux is known for its ability to run for long periods without crashing, making it ideal for servers and critical systems.
4. **Security**: Linux has a strong security model, with built-in features like user permissions, firewalls, and regular updates to address vulnerabilities.
5. **Multi-User and Multi-Tasking**: Linux supports multiple users and allows them to run multiple processes simultaneously.
6. **Wide Hardware Support**: Linux runs on a variety of hardware, from smartphones (Android is Linux-based) to supercomputers.
7. **Large Community Support**: A global community of developers and users contributes to its development, ensuring continuous improvement and support.

**Why Linux is Important:**

1. **Powering the Internet**: Most web servers, cloud infrastructure, and data centers run on Linux. Popular platforms like Google, Amazon, and Facebook rely on Linux for their backend operations.
2. **Cost-Effective**: Being free and open-source, Linux reduces software licensing costs, making it accessible to individuals, businesses, and governments.
3. **Versatility**: Linux is used in diverse environments, from embedded systems (like IoT devices) to supercomputers (90% of the world’s top supercomputers run Linux).
4. **Privacy and Control**: Unlike proprietary operating systems, Linux gives users full control over their system, ensuring transparency and reducing the risk of unwanted data collection.
5. **Innovation**: Linux fosters innovation by allowing developers to experiment and build new tools, applications, and systems without restrictions.
6. **Educational Value**: Linux is a great platform for learning about operating systems, programming, and system administration.
7. **Supports Development**: Many programming languages, frameworks, and tools are natively supported on Linux, making it a favorite among developers.

**Popular Linux Distributions (Distros):**

* **Ubuntu**: User-friendly and widely used for desktops and servers.
* **Fedora**: Known for cutting-edge features and Red Hat support.
* **Debian**: Stable and versatile, serving as the base for many other distros.
* **CentOS/RHEL**: Popular in enterprise environments.
* **Arch Linux**: Highly customizable for advanced users.
* **Linux Mint**: Beginner-friendly with a focus on simplicity.

**Conclusion:**

Linux is a cornerstone of modern computing, powering everything from personal devices to global infrastructure. Its open-source nature, combined with its robustness and flexibility, makes it a critical tool for innovation, security, and efficiency in the digital age. Whether you're a developer, sysadmin, or casual user, Linux offers something for everyone.

**What is a Command Line Interpreter (CLI)?**

A **Command Line Interpreter (CLI)** is a program that takes commands from the keyboard and executes them in an operating system. It allows users to interact with the system by typing commands instead of using a graphical interface.

**Examples of Command Line Interpreters:**

* **Bash** (Bourne Again Shell ) – Default in Linux/macOS
* **Cmd (Command Prompt)** – Default in Windows
* **PowerShell** – Advanced Windows shell
* **Zsh (Z Shell)** – An improved version of Bash
* **sh (Bourne Shell)** – Older Unix shell

**How It Works:**

1. **User Inputs a Command** – e.g., ls (to list files).
2. **CLI Processes the Command** – Checks if it's a built-in command or an external program.
3. **Executes & Displays Output** – Shows the result in the terminal.

**Example in Bash:**

echo "Hello, World!"

➡ Output: Hello, World!

**What is Bash?**

**Bash (Bourne Again Shell)** is a command-line interpreter and scripting language used in Linux and Unix systems. It allows users to run commands, automate tasks, and write scripts for system administration.

**Key Features of Bash:**

✅ **Command Execution** – Run Linux commands like ls, cd, cat, etc.  
✅ **Scripting** – Automate tasks using .sh scripts.  
✅ **Variables** – Store and manipulate data (name="Anbu").  
✅ **Loops & Conditions** – Use if, for, while for logic control.  
✅ **Pipelines & Redirection** – Combine commands (|, >, >>).

**Example Bash Script:**

#!/bin/bash

echo "Hello, what is your name?"

read name

echo "Welcome, $name!"

Save this as **greet.sh**, then run:

bash greet.sh

**Why Do We Use #!/bin/bash in Shell Scripts?**

The line #!/bin/bash at the beginning of a shell script is called a **shebang** (#!). It tells the system which interpreter should be used to execute the script.

**Purpose of #!/bin/bash:**

✅ **Specifies the Interpreter** – Ensures the script runs in Bash, even if the user’s default shell is different.  
✅ **Avoids Compatibility Issues** – Prevents errors if the script uses Bash-specific features.  
✅ **Allows Direct Execution** – Without this line, the script must be run using bash script.sh instead of ./script.sh.

**Example Script (script.sh)**

#!/bin/bash

echo "Hello, this is a Bash script!"

**How to Run the Script:**

1. **Make it executable:**

chmod +x script.sh

1. **Run it:**

./script.sh

💡 If you use #!/bin/sh, it may use a different shell like Dash (on Debian-based systems), which lacks some Bash-specific features.

**Commands:**

* sudo -i //switch to root user for administrator privilege.
* mkdir learnLinux // Make new directory/folder
* cd learnLinux // change directory /move to the specified folder/directory, Also type some char and then use TAB key
* cd .. // Go back to the preview folder/directory ie one step back
* cd ../.. // Go to two step back
* cd // Go back to the home directory/folder
* touch one.txt // To create an empty file.
* Touch one.txt two.txt three.txt four.txt // To create a multiple empty files
* ls // displays list of the files and directory
* ls -l // displays files and folder with extension and permission

The -l option stands for "long listing format."

It provides detailed information about each file and directory, including:

* File type and permissions
* Number of hard links
* Owner name
* Group name
* File size
* Modification date and time
* File/directory name
* ls -la
  + The -l option provides the long listing format (as explained above).
  + The -a option stands for "all," which includes hidden files (files starting with .)
  + This means ls -la lists **all files, including hidden ones**.

**Key Differences:**

| **Command** | **Shows Hidden Files?** | **Long Listing Format?** |
| --- | --- | --- |
| ls -l | ❌ No | ✅ Yes |
| ls -la | ✅ Yes | ✅ Yes |

* nano new.txt // It is a simple and user-friendly text editor for the terminal. It is used to create, edit, and view text files.
  + Welcome guys. // Press ctrl+x and y & Press ENTER kay

**Common Nano Commands**

| **Command** | **Description** |
| --- | --- |
| nano file.txt | Open (or create) a file in Nano |
| Ctrl + O | Save the file |
| Ctrl + X | Exit Nano |
| Ctrl + W | Search for text in the file |
| Ctrl + K | Cut a line |
| Ctrl + U | Paste the cut line |
| Ctrl + G | Show help menu |

**When to Use Nano?**

* When you need a quick and simple text editor.
* If you find vi or vim difficult to use.
* For small configuration file edits.

For larger and more complex editing, editors like vim or emacs may be better choices.

* cat one.txt // (short for **concatenate**) This command is used to read, display, and concatenate files in Linux.
  + Ctrl + C **cancels a running process**.
  + Ctrl + D **signals end-of-input (EOF)**, commonly used to save and exit file operations like cat >> file.txt.

| **Command** | **Description** |
| --- | --- |
| cat file.txt | Display file content |
| cat > file.txt | Create a new file |
| cat >> file.txt | Append content to a file |
| cat file1.txt file2.txt > merged.txt | Merge multiple files |
| cat -n file.txt | Show file content with line numbers |
|  |  |

* echo “Welcome” // to display text or variables in the terminal. It is commonly used in shell scripting and printing messages.

**Common Uses of echo**

1. **Print Text**

echo "Welcome to Linux!"

**Output:**

Welcome to Linux!

1. **Print Environment Variables**

echo $HOME

**Output:**

/home/username

(Displays the home directory of the user.)

1. **Print a Variable Value**

name="Elakkiya"

echo "Hello, $name!"

**Output:**

Hello, Elakkiya!

1. **Enable Escape Characters (-e option)**

echo -e "Line 1\nLine 2\nLine 3"

**Output:**

Line 1

Line 2

Line 3

(\n creates a new line.)

1. **Write to a File**

echo "This is a test" > file.txt

(Creates file.txt and writes "This is a test" to it.)

1. **Append to a File**

echo "Another line" >> file.txt

(Adds "Another line" to file.txt without deleting existing content.)

1. **Suppress New Line (-n option)**

echo -n "Hello, World!"

(Prints text **without a newline** at the end.)

1. **Display Colors in Terminal**

echo -e "\e[32mThis is green text\e[0m"

(Displays **green-colored** text.)

**Summary Table**

| **Command** | **Description** |
| --- | --- |
| echo "Hello" | Print text |
| echo $HOME | Print environment variable value |
| echo -e "Line1\nLine2" | Enable escape sequences (new line) |
| echo "Text" > file.txt | Write text to a file |
| echo "Text" >> file.txt | Append text to a file |
| echo -n "Hello" | Print text without a newline |

* rm one.txt // To remove particular file
* rm \* // To remove all the file
* rm -r // To remove all the files and folder with content.
* cp test.txt mfolder/ // To copy file
* mv myfile.txt mfolder // To move file or rename file
* mkdir test && cd test // Run multiple commands in a single line
* mkdir test;cd test // Run multiple commands in a single line
* mkdir test|cd test // Run multiple commands in a single line
* rmdir test // To remove directory/folder. If you want to remove a folder, It should be empty ie. No file inside that.
* man rm // To know more about the command, It’s a manual of the command. We can check any command using man
  + Press q to quit
* clear // To clear the screen
* exit // come out from the remote server to local terminal account
* history // To see all the previously used commands
* ~ // Means home directory/folder
* sudo su // switch to root user
* pwd // To know the present working directory
* which docker // To know where the docker has installed (path)
* vi file.txt

Network:

* ping google.com // Press CTRL+C to come out
* nslookup google.com
* curl <https://www.google.com> // Given source code of the page
* man curl // To know more about the command
* wget [https://www.google.com/images/branding/googlelogo/2x/googlelogo\_color\_272x92dp.png //](https://www.google.com/images/branding/googlelogo/2x/googlelogo_color_272x92dp.png%20//) To download the file

**Add Two Numbers using Shell Scripting:**

>nano add.sh

#!/bin/bash

echo “Enter 2 numbers:”

read num1

read num2

echo “Sum of 2 numbers: $((num1+num2))” //ctrl+x, y then Enter

>ls -la // to all the files

>chmod u+x add.sh // add permission to execute the file

>./add.sh // To run the script file

Enter 2 numbers:

10

10

Sum of 2 numbers:20

>

**Package:**