



**MVPS's
RAJARSHI SHAHU MAHARAJ POLYTECHNIC,
NASHIK.**

DEPARTMENT OF COMPUTER TECHNOLOGY.

ACADEMIC YEAR 2021-22

OPERATING SYSTEM (22516)

**MICRO-PROJECT
ON**

“Prepare Help Guide Using Shell Script for All Major Linux Commands”

SUBMITTED BY

SR. NO	ENROLLMENT NO	EXAM SEAT NO	STUDENT NAME
1	1910020362		Savant Omkar Vitthal
2	1910020363		Sawant Diksha Pravin
3	1910020364		Shimpi Om Vilas



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

CERTIFICATE

This is to certify that Ms./Mr. Savant Omkar Vitthal

Roll No. 46 of 5th semester of Diploma in Computer Technology of

Institute MVPS's Rajarshi Shahu Maharaj Polytechnic, Nashik (Code: 1002)

has successfully completed micro-project in OPERATING SYSTEM (22516)

for academic year 2021-22 as prescribed in curriculum of MSBTE, Mumbai.

Place: Nashik

Enrollment No.: 1910020362

Date:

Exam Seat No.:

Mr. R. S. Derle
Course Teacher/Guide

Mr. P. D. Boraste
H.O.D

Prof. Dr. D. B. Uphade
Principal





**MVPS's RAJARSHI SHAHU MAHARAJ
POLYTECHNIC, NASIK**
Institute Code: 1002
COMPUTER TECHNOLOGY DEPARTMENT
Log Book for Micro Project

Academic Year : 2021-22

Semester : V

Name of Course : Operating System (OSY)

Scheme: I

Class: TYCM

Course Code: 22516

Title of the project: Prepare Help Guide Using Shell Script for All Major Linux Commands

Group Members:

Sr. No.	Roll No.	Enrollment Number	Exam Seat No.	Name of the Student	Signature of student
1	46	1910020362		Savant Omkar Vitthal	
2	47	1910020363		Sawant Diksha Pravin	
3	48	1910020364		Shimpi Om Vilas	

Project Reporting:

Sr. No.	Date	Discussion & details	Group members present	Teacher's comment/remark	Signature of teacher
1		Formation of groups			
2		Discussion on concept of Micro Project			
3		Topic selection for the Micro Project			
4		Preliminary discussion with guide			
5		Submission of Micro Project proposal			
6		Information Gathered			
7		Literature survey (Introduction)			
8		Discussion with guide			
9		Design of GUI and output			
10		Code generation of modules			
11		Error Evaluation			
12		Merging of all individual modules into one single module			
13		Draft copy of report.			
14		Final report writing			
15		Presentation & oral			
16		Final submission			



**MVPS's RAJARSHI SHAHU MAHARAJ
POLYTECHNIC, NASIK**
Institute Code: 1002
COMPUTER TECHNOLOGY DEPARTMENT

Rubrics for Evaluation of Micro Project

Academic Year: 2021-22

Semester: V

Name of Course: Operating System (OSY)

Scheme: I

Class: TYCM

Course Code: 22516

Group Members:

Sr. No.	Roll No.	Enrollment Number	Exam Seat No.	Name of the Student	Signature of student
1	46	1910020362		Savant Omkar Vitthal	
2	47	1910020363		Sawant Diksha Pravin	
3	48	1910020364		Shimpi Om Vilas	

Sr. No.	Criteria	Indicators of different levels of performance					Marks obtained
		Poor (01)	Satisfactory (02)	Good (03)	Very Good (04)	Excellent (05)	
1	Selection Of Application (Programming Elements)						
2	Concept/ Content/ Function Descriptions						
3	Coding						
4	Error solving						
5	Timely Submission						
Total marks Out of 25							
Marks Out of 6							

Mr. R. S. Derle
Name & Signature of Course Teacher/Guide



**MVPS's RAJARSHI SHAHU MAHARAJ
POLYTECHNIC, NASIK**
Institute Code: 1002
COMPUTER TECHNOLOGY DEPARTMENT
ANNEXTURE II

Academic Year: 2021-22

Name of Faculty: Mr. R. S. Derle

Course: Operating System (OSY)

Course code: 22516 Semester: V

Title of the project: Prepare Help Guide Using Shell Script for All Major Linux Commands

COs addressed by the Micro Project:

CO-516.1 Install Operating System and configure it

CO-516.2 Use Operating system tools to perform various functions

CO-516.3 Execute process commands for performing process management operations

CO-516.4 Apply scheduling algorithms to calculate turnaround time and average waiting time

CO-516.5 Calculate efficiency of different memory management Techniques

CO-516.6 Apply File management techniques.

Major learning Outcomes achieved by students by doing the Project:

a) Practical Outcomes

.....
.....

b) Unit outcomes in cognitive domain

.....
.....

c) Outcomes in affective domain

.....
.....

Comment/Suggestions about team work/leadership/inter-personal communication (if any)

.....
.....

Roll. No.	Enrollment No.	Exam Seat No.	Student Name	Marks out of 6 for performance in group activity	Marks out of 4 for performance oral/ presentation	Total out of 10
46	1910020362		Savant Omkar Vitthal			
47	1910020363		Sawant Diksha Pravin			
48	1910020364		Shimpi Om Vilas			

Mr. R. S. Derle
Course Teacher/Guide

Mr. P. D. Boraste
H.O.D

ABSTRACT

This paper is scrutinizes the use of different concepts of Commands of Linux in Operating System subject, enabling viewer to get the complete concept of different aspects of JAVA programming. Linux is an open source operating system. Various commands are used in it to customize and gain control of the operating system. To satisfy this we created a help guide using shell script for all major Linux Commands

INDEX

Sr. No.	Contents	Page No.
1.	Introduction	3
2.	History	7
3.	Commands	10
4.	Conclusion	14
5.	Reference	15

Chapter-1

INTRODUCTION

1. Linux:

Linux is a family of open-source Unix-like operating systems based on the Linux kernel, an operating system kernel first released on September 17, 1991, by Linus Torvalds. Linux is typically packaged in a Linux distribution. Popular Linux distributions include Debian, Fedora, and Ubuntu. Commercial distributions include Red Hat Enterprise Linux and SUSE Linux Enterprise Server. Desktop Linux distributions include a windowing system such as X11 or Wayland, and a desktop environment such as GNOME or KDE Plasma. Distributions intended for servers may omit graphics altogether, or include a solution stack such as LAMP. Linux was originally developed for personal computers based on the Intel x86 architecture, but has since been ported to more platforms than any other operating system. Because of the dominance of the Linux-based Android on smartphones, Linux also has the largest installed base of all general-purpose operating systems. Although it is used by only around 2.3 percent of desktop computers, the Chromebook, which runs the Linux kernel-based Chrome OS, dominates the US K–12 education market and represents nearly 20 percent of sub-\$300 notebook sales in the US. Linux is the leading operating system on servers. Because Linux is freely redistributable, anyone may create a distribution for any purpose. Linux is one of the most prominent examples of free and open-source software collaboration. The source code may be used, modified and distributed commercially or non-commercially by anyone under the terms of its respective licenses, such as the GNU General Public License.

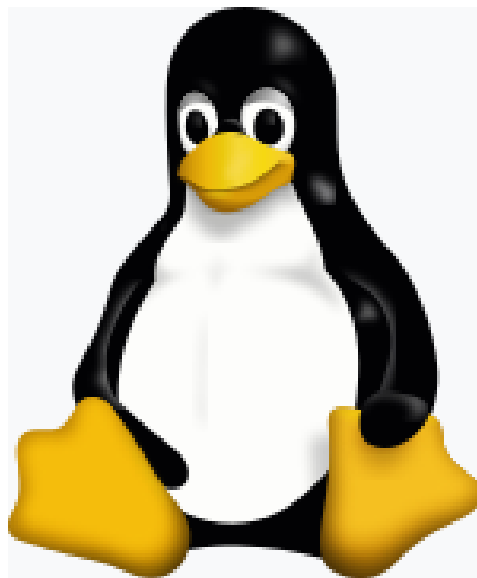


Fig.1. Linux

2. Shell:

A Unix shell is a command-line interpreter or shell that provides a command line user interface for Unix-like operating systems. The shell is both an interactive command language and a scripting language, and is used by the operating system to control the execution of the system using shell scripts. Users typically interact with a Unix shell using a terminal emulator; however, direct operation via serial hardware connections or Secure Shell are common for server systems. All Unix shells provide filename wildcarding, piping, here documents, command substitution, variables and control structures for condition-testing and iteration. The most generic sense of the term shell means any program that users employ to type commands. A shell hides the details of the underlying operating system and manages the technical details of the operating system kernel interface, which is the lowest-level, or "inner-most" component of most operating systems. In Unix-like operating systems, users typically have many choices of command-line interpreters for interactive sessions. When a user logs into the system interactively, a shell program is automatically executed for the duration of the session. On hosts with a windowing system, like macOS, some users may never use the shell directly. On Unix systems, the shell has historically been the implementation language of system startup scripts, including the program that starts a windowing system, configures networking, and many other essential functions. However, some system vendors have replaced the traditional shell-based startup system (init) with different approaches, such as systemd.

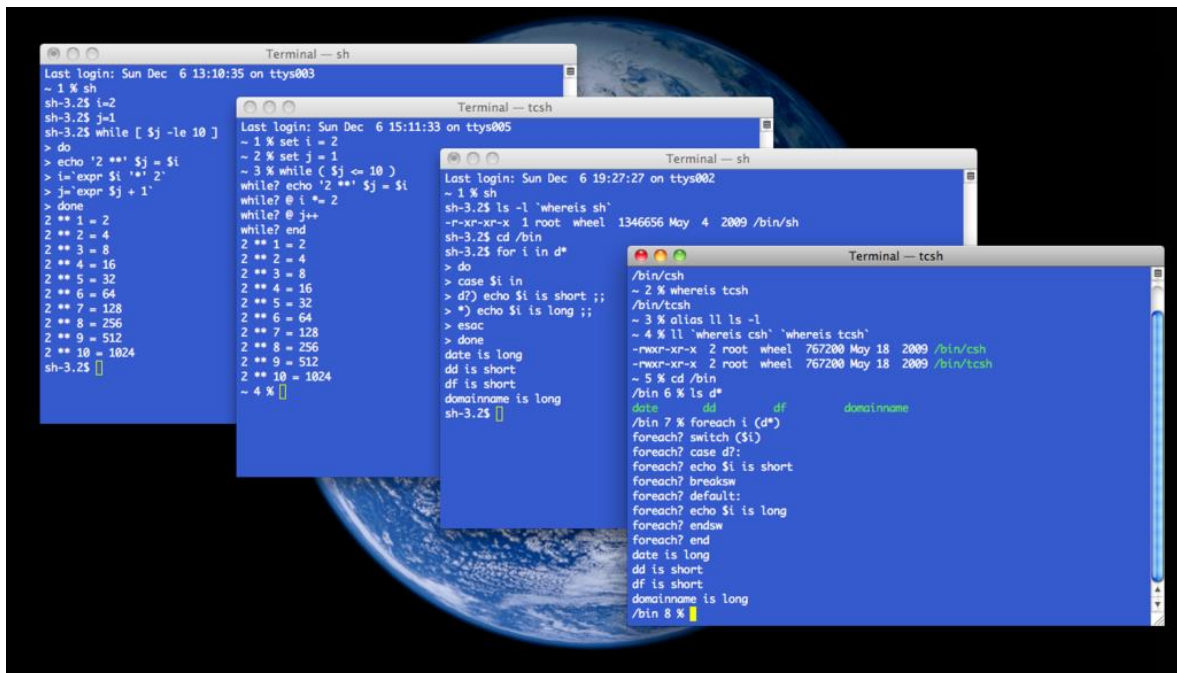


Fig.2. tcsh and sh shell windows on a Mac OS X

3. Terminal:

A computer terminal is an electronic or electromechanical hardware device that is used for entering data into, and displaying data from, a computer or a computing system. Early terminals were inexpensive devices but very slow compared to punched cards or paper tape for input, but as the technology improved and video displays were introduced, terminals pushed these older forms of interaction from the industry. A related development was timesharing systems, which evolved in parallel and made up for any inefficiencies of the user's typing ability with the ability to support multiple users on the same machine, each at their own terminal.

The function of a terminal is confined to display and input of data; a device with significant local programmable data processing capability may be called a "smart terminal" or fat client. A terminal that depends on the host computer for its processing power is called a "dumb terminal" or thin client. A personal computer can run terminal emulator software that replicates the function of a terminal, sometimes allowing concurrent use of local programs and access to a distant terminal host system.

Early user terminals connected to computers were electromechanical teleprinters/teletypewriters (TeleTYpewriter, TTY), such as the Teletype Model 33 ASR, originally used for telegraphy or the Friden Flexowriter.

```

07/01/2019 01:00 PM          440 CODE_OF_CONDUCT.md
07/01/2019 01:00 PM          568 common.openconsole.props
11/01/2019 11:41 AM          530 consolegit2gitfilters.json
11/19/2019 10:06 AM       9,000 contributing.md
01/10/2020 10:09 AM          544 custom.props
07/12/2019 02:54 PM      <DIR>      Debug
07/19/2019 09:16 AM      <DIR>      dep
07/01/2019 01:00 PM          21 dirs
01/13/2020 11:06 AM      <DIR>      doc
07/01/2019 01:00 PM       1,116 LICENSE
12/20/2019 00:18 AM       3,078 NOTICE.md
07/19/2019 09:16 AM       1,120 NuGet.Config
10/15/2019 00:57 AM      <DIR>      obj
01/13/2020 11:06 AM     111,391 OpenConsole.sln
01/13/2020 11:07 AM      <DIR>      packages
07/01/2019 01:00 PM      <DIR>      pkg
01/03/2020 09:00 AM       12,978 README.md
07/14/2019 05:05 PM      <DIR>      Release
12/20/2019 00:18 AM      <DIR>      res
07/01/2019 01:00 PM      <DIR>      samples
11/01/2019 11:41 AM       2,766 SECURITY.md
01/13/2020 11:06 AM      <DIR>      src
01/13/2020 11:06 AM      <DIR>      tools
07/19/2019 01:46 PM     103,484 UpgradeLog.htm
07/14/2019 05:06 PM      <DIR>      x64

19 File(s)      260,046 bytes
20 Dir(s)      226,350,678,016 bytes free

C:\Users\cinnamon\GitHub\WindowsTerminal>

```


Fig.3. Terminal

4. Commands:

In computing, a command is a directive to a computer program to perform a specific task. It may be issued via a command-line interface, such as a shell, or as input to a network service as part of a network protocol, or as an event in a graphical user interface triggered by the user selecting an option in a menu.

Specifically, the term command is used in imperative computer languages. The name arises because statements in these languages are usually written in a manner similar to the imperative mood used in many natural languages. If one views a statement in an imperative language as being like a sentence in a natural language, then a command is generally like a verb in such a language.

Many programs allow specially formatted arguments, known as flags or options, which modify the default behaviour of the program, while further arguments may provide objects, such as files, to act on. Comparing to a natural language: the flags are adverbs, while the other arguments are objects.



```
Starting DeviceLogics DR-DOS...  
  
DeviceLogics DR-DOS 8.0  
Copyright (c) 1976, 2004 DeviceLogics, LLC. All rights reserved.  
  
Date: Sun 4-26-2020  
Enter date (mm-dd-yy):  
Time: 15:00:00.00  
Enter time:  
  
C:\>_
```

Fig.4. Date Command

Chapter 2

History

1. Linux:

The Unix operating system was conceived and implemented in 1969, at AT&T's Bell Labs, in the United States by Ken Thompson, Dennis Ritchie, Douglas McIlroy, and Joe Ossanna. First released in 1971, Unix was written entirely in assembly language, as was common practice at the time. In 1973 in a key, pioneering approach, it was rewritten in the C programming language by Dennis Ritchie (with the exception of some hardware and I/O routines). The availability of a high-level language implementation of Unix made its porting to different computer platforms easier.

In 1991, while attending the University of Helsinki, Torvalds became curious about operating systems. Frustrated by the licensing of MINIX, which at the time limited it to educational use only, he began to work on his own operating system kernel, which eventually became the Linux kernel.

Torvalds began the development of the Linux kernel on MINIX and applications written for MINIX were also used on Linux. Later, Linux matured and further Linux kernel development took place on Linux systems. GNU applications also replaced all MINIX components, because it was advantageous to use the freely available code from the GNU Project with the fledgling operating system; code licensed under the GNU GPL can be reused in other computer programs as long as they also are released under the same or a compatible license. Torvalds initiated a switch from his original license, which prohibited commercial redistribution, to the GNU GPL. Developers worked to integrate GNU components with the Linux kernel, making a fully functional and free operating system.



Fig.4. Linus Torvalds, principal author of the Linux kernel

2. Shell:

The first Unix shell was the Thompson shell, sh, written by Ken Thompson at Bell Labs and distributed with Versions 1 through 6 of Unix, from 1971 to 1975. Though rudimentary by modern standards, it introduced many of the basic features common to all later Unix shells, including piping, simple control structures using if and goto, and filename wildcarding. Though not in current use, it is still available as part of some Ancient UNIX systems.

It was modeled after the Multics shell, developed in 1965 by American software engineer Glenda Schroeder. Schroeder's Multics shell was itself modeled after the RUNCOM program Louis Pouzin showed to the Multics Team. The "rc" suffix on some Unix configuration files (for example, ".vimrc"), is a remnant of the RUNCOM ancestry of Unix shells.

The PWB shell or Mashey shell, sh, was an upward-compatible version of the Thompson shell, augmented by John Mashey and others and distributed with the Programmer's Workbench UNIX, circa 1975–1977. It focused on making shell programming practical, especially in large shared computing centers. It added shell variables (precursors of environment variables, including the search path mechanism that evolved into \$PATH), user-executable shell scripts, and interrupt-handling. Control structures were extended from if/goto to if/then/else/endif, switch/breaksw/endsw, and while/end/break/continue. As shell programming became widespread, these external commands were incorporated into the shell itself for performance.

But the most widely distributed and influential of the early Unix shells were the Bourne shell and the C shell. Both shells have been used as the coding base and model for many derivative and work-alike shells with extended feature sets.



Fig.5. Ken Thompson

3. Terminal:

The console of Konrad Zuse's Z3 had a keyboard in 1941, as did the Z4 in 1942-45. But these consoles could only be used to enter numeric inputs and were thus analogous to those of calculating machines; programs, commands, and other data were entered via paper tape. Both machines had a row of display lamps for results.

In 1955, the Whirlwind Mark I computer was the first computer equipped with a keyboard-printer combination with which to support direct input of data and commands and output of results. The device was a Friden Flexowriter, which would continue to serve this purpose on many other early computers well into the 1960s.

An "intelligent" terminal does its own processing, usually implying a microprocessor is built in, but not all terminals with microprocessors did any real processing of input: the main computer to which it was attached would have to respond quickly to each keystroke. The term "intelligent" in this context dates from 1969.

Notable examples include the IBM 2250, predecessor to the IBM 3250 and IBM 5080, and IBM 2260, predecessor to the IBM 3270, introduced with System/360 in 1964.

IBM 2250 Model 4, including light pen and programmed function keyboard

Most terminals were connected to minicomputers or mainframe computers and often had a green or amber screen. Typically terminals communicate with the computer via a serial port via a null modem cable, often using an EIA RS-232 or RS-422 or RS-423 or a current loop serial interface. In fact, the instruction design for the Intel 8008 was originally conceived at Computer Terminal Corporation as the processor for the Datapoint 2200.



Fig.6. IBM 2250 Model 4, including light pen and programmed function keyboard

Chapter-3

COMMANDS

1. pwd:

When you first open the terminal, you are in the home directory of your user. To know which directory you are in, you can use the “pwd” command. It gives us the absolute path, which means the path that starts from the root. The root is the base of the Linux file system. It is denoted by a forward slash(/). The user directory is usually something like “/home/username”.

```
nayso@Alok-Aspire:~$ pwd
/home/nayso
```

2. ls:

Use the "ls" command to know what files are in the directory you are in. You can see all the hidden files by using the command “ls -a”.

```
nayso@Alok-Aspire:~$ ls
Desktop          itsuserguide.desktop  reset-settings        VCD_Copy
Documents        Music                  School_Resources     Videos
Downloads        Pictures               Students_Works_10
examples.desktop Public                 Templates
GplatesProject   Qgis Projects         TuxPaint-Pictures
```

3. cd:

Use the "cd" command to go to a directory. For example, if you are in the home folder, and you want to go to the downloads folder, then you can type in “cd Downloads”.

```
nayso@Alok-Aspire:~$ cd Downloads
nayso@Alok-Aspire:~/Downloads$ cd
nayso@Alok-Aspire:~$ cd Raspberry\ Pi
nayso@Alok-Aspire:~/Raspberry Pi$ cd ..
nayso@Alok-Aspire:~$
```


4. mkdir & rmdir:

Use the mkdir command when you need to create a folder or a directory. For example, if you want to make a directory called “DIY”, then you can type “mkdir DIY”. Use rmdir to delete a directory. But rmdir can only be used to delete an empty directory. To delete a directory containing files, use rm.

```
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$ mkdir DIY
nayso@Alok-Aspire:~/Desktop$ ls
DIY
nayso@Alok-Aspire:~/Desktop$ rmdir DIY
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$
```

5. rm:

Use the rm command to delete files and directories. Use "rm -r" to delete just the directory. It deletes both the folder and the files it contains when using only the rm command.

```
nayso@Alok-Aspire:~/Desktop$ ls
newer.py  New Folder
nayso@Alok-Aspire:~/Desktop$ rm newer.py
nayso@Alok-Aspire:~/Desktop$ ls
New Folder
nayso@Alok-Aspire:~/Desktop$ rm -r New\ Folder
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$
```

6. touch:

The touch command is used to create a file. It can be anything, from an empty txt file to an empty zip file. For example, “touch new.txt”.

```
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$ touch new.txt
nayso@Alok-Aspire:~/Desktop$ ls
new.txt
```


7. man & --help:

To know more about a command and how to use it, use the man command. It shows the manual pages of the command. For example, “man cd” shows the manual pages of the cd command. Typing in the command name and the argument helps it show which ways the command can be used (e.g., cd – help).

```
TOUCH(1)                                User Commands                                TOUCH(1)

NAME
    touch - change file timestamps

SYNOPSIS
    touch [OPTION]... FILE...

DESCRIPTION
    Update the access and modification times of each FILE to the current
    time.

    A FILE argument that does not exist is created empty, unless -c or -h
    is supplied.

    A FILE argument string of - is handled specially and causes touch to
    change the times of the file associated with standard output.

    Mandatory arguments to long options are mandatory for short options
    too.

    -a      change only the access time

Manual page touch(1) line 1 (press h for help or q to quit)
```

8. cp:

Use the cp command to copy files through the command line. It takes two arguments: The first is the location of the file to be copied, the second is where to copy.

```
nayso@Alok-Aspire:~/Desktop$ ls /home/nayso/Music/
nayso@Alok-Aspire:~/Desktop$ cp new.txt /home/nayso/Music/
nayso@Alok-Aspire:~/Desktop$ ls /home/nayso/Music/
new.txt
```

9. mv:

Use the mv command to move files through the command line. We can also use the mv command to rename a file. For example, if we want to rename the file “text” to “new”, we can use “mv text new”. It takes the two arguments, just like the cp command.

```
nayso@Alok-Aspire:~/Desktop$ ls
new.txt
nayso@Alok-Aspire:~/Desktop$ mv new.txt newer.txt
nayso@Alok-Aspire:~/Desktop$ ls
newer.txt
```

10. locate:

The locate command is used to locate a file in a Linux system, just like the search command in Windows. This command is useful when you don't know where a file is saved or the actual name of the file. Using the -i argument with the command helps to ignore the case (it doesn't matter if it is uppercase or lowercase). So, if you want a file that has the word “hello”, it gives the list of all the files in your Linux system containing the word "hello" when you type in “locate -i hello”. If you remember two words, you can separate them using an asterisk (*). For example, to locate a file containing the words "hello" and "this", you can use the command “locate -i *hello*this”.

```
nayso@Alok-Aspire:~$ locate newer.txt
/home/nayso/Desktop/newer.txt
nayso@Alok-Aspire:~$ locate *DIY*Hacking*
/home/nayso/DIY Hacking
```

Chapter-4

CONCLUSION

Linux is an operating system's kernel. You might have heard of UNIX. Well, Linux is a UNIX clone. But it was actually created by Linus Torvalds from Scratch. Linux is free and open-source, that means that you can simply change anything in Linux and redistribute it in your own name. There are several Linux Distributions, commonly called “distros”.

Ubuntu Linux

Red Hat Enterprise Linux

Linux Mint

Debian

Fedora

Linux is Mainly used in servers. About 90% of the internet is powered by Linux servers. This is because Linux is fast, secure, and free! The main problem of using Windows servers are their cost. This is solved by using Linux servers. The OS that runs in about 80% of the smartphones in the world, Android, is also made from the Linux kernel. Most of the viruses in the world run on Windows, but not on Linux!

Linux Shell or “Terminal”

So, basically, a shell is a program that receives commands from the user and gives it to the OS to process, and it shows the output. Linux's shell is its main part. Its distros come in GUI (graphical user interface), but basically, Linux has a CLI (command line interface).

To open the terminal, press Ctrl+Alt+T in Ubuntu, or press Alt+F2, type in gnome-terminal, and press enter. In Raspberry Pi, type in lxterminal. There is also a GUI way of taking it, but this is better.

Chapter - 5

REFERENCES

- 1.** <https://maker.pro/linux/tutorial/basic-linux-commands-for-beginners>
- 2.** [https://en.wikipedia.org/wiki/Command_\(computing\)](https://en.wikipedia.org/wiki/Command_(computing))
- 3.** https://en.wikipedia.org/wiki/Computer_terminal
- 4.** https://en.wikipedia.org/wiki/Unix_shell
- 5.** <https://en.wikipedia.org/wiki/Linux>