**PRACTICAL FILE**

**OPERATING SYSTEMS**

January 2024-June 2024

BE (CSE)4th SEM



DEPARTMENT COMPUTER SCIENCE ENGINEERING

UNIVERSITY INSTITUTE OF ENGINEERING AND TECHNOLOGY

PUNJAB UNIVERSITY, CHANDIGARH

Under the guidance of: Submitted By : Amanpreet Kaur

Dr Mamta Juneja UEM223117

Professor,Dept of Computer Science and Engineering Group 3

UIET Chandigarh BE CSE 4th Sem

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **PRACTICAL** | **DATE** | **PAGE NO** |
| 1 | 1. Introduction to Operating System,    Types of Operating System, Difference between Linux,Unix and Windows 2. Installation of Ubuntu | 18/01/2024 |  |
| 2 | To implement the commands:  cat,man,echo,touch,ls,mkdir,cd,cp, pwd,tty,who,wc,mv, rmdir,rm,script | 25/01/2024 |  |
| 3 | To implement the commands:  whatis , whereis,find,type,bc,expr, diff,cmp,comm,sort,tee,cut,tr, grep,head,tail,free,df,du,ulimit, cal,ncal,pipe(|) | 01/02/2024 |  |
| 4 | To implement the commands:  umask , chmod, adduser,su, deluser,gzip/gunzip,tar, split,sleep,shutdown,chown,semicolon(;) | 08/02/2024 |  |
| 5 | To implement the commands:  ps,nohup,kill,nice,batch, crontab,wall,write | 15/02/2024 |  |
| 6 | Shell scripting  → Introduction to vim editor  → Syntax of control structure | 22/02/2024 |  |
| 7 | Programs:     * Addition of two numbers * Check whether the number entered is odd/even * Check whether the year is leap or not * Calculate the average salaries of employees * Calculate the average marks of students * To find circumference and area of circle * To find the sum of digits of a number. * Find the factorial of a number * Find the fibonacci series * Check whether a number is prime or not * Check whether the number is palindrome or not * Reverse of a given number * To create a binary calculator using switch case * Linear Search * Binary search * Bubble sort | 29/02/2024 |  |
| 8. | To implement the following String Operations programs in vim editor:  i) Find length of string  ii) Compare two strings  iii) Reverse of a string  iv) String is palindrome or not  v) Occurrence of a pattern  vi)Concatenate strings  vii) Convert lowercase letters into uppercase. | 07/03/2024 |  |
| 9. | To implement the following File Operations programs in vim editor :  i) To see permissions of a file after checking whether it is file or directory  ii) Rename file  iii) Take backup of a directory  iv) Compare directories and delete duplicate files  v) To check the number of files through command  line arguments  vi) If number of files is even, then copy first argument in second and so on, else display odd  vii) Number of arguments.  viii) Convert lowercase letters into uppercase.  ix) To calculate length of a file | 14/03/2024 |  |
| 10 | Write a C++ program to implement the following CPU scheduling algorithms :     * First Come First Serve(FCFS) * Shortest Job First(SJF) (P, NP) * Priority Scheduling (P, NP) * Round Robin(RR) | 21/03/2024 |  |
| 11 | C++ programming to implement Banker’s Algorithm:     * Safety Algorithm- * Request and Resource Allocation |  |  |
| 12 | Write a C++ program to implement the following Process  synchronisation algorithms with semaphores :     * Producer Consumer Problem * Reader Writers Problem * Dining Philosopher Problem |  |  |
| 13 | C++ programming to implement Page Faults:     * FIFO page replacement * Optimal Page Replacement (OPR) * Least Recently Used(LRU) * Most Frequently Used(MFU) * Least Frequently Used(LFU) |  |  |
| 14 | Write C++ programs to implement Disk Scheduling Algorithm:     * FCFS * SSTF * SCAN * C-SCAN * LOOK * C-LOOK |  |  |
| 15 | Write C++ programs to implement the following Memory management algorithms (MVT) :     * First Fit (FF) * Best Fit (BF) * Worst Fit (WF) |  |  |

Introduction

**Operating System** lies in the category of system software. It basically manages all the resources of the computer. An operating system acts as an interface between the software and different parts of the computer or the computer hardware. The operating system is designed in such a way that it can manage the overall resources and operations of the computer.

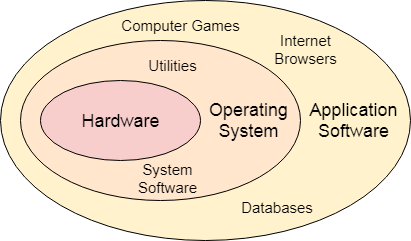
**What is an operating system?**

Operating System is a fully integrated set of specialized programs that handle all the operations of the computer. It controls and monitors the execution of all other programs that reside in the computer, which also includes application programs and other system software of the computer. Examples of Operating Systems are Windows, Linux, Mac OS, etc.

An Operating System (OS) is a collection of software that manages computer hardware resources and provides common services for computer programs. The operating system is the most important type of system software in a computer system.

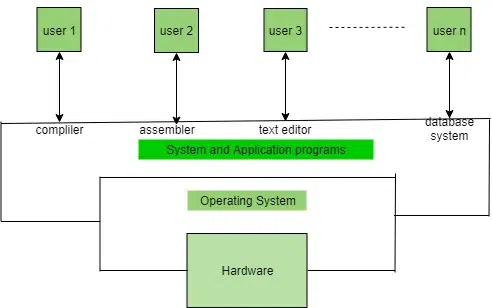
An **Operating System** can be defined as an **interface between user and hardware**. It is responsible for the execution of all the processes, Resource Allocation, [CPU](https://www.javatpoint.com/cpu-full-form) management, File Management and many other tasks.

The purpose of an operating system is to provide an environment in which a user can execute programs in convenient and efficient manner.



## **What is an Operating System Used for?**

The operating system helps in improving the computer software as well as hardware. Without OS, it became very difficult for any application to be user-friendly. The Operating System provides a user with an interface that makes any application attractive and user-friendly. The operating System comes with a large number of device drivers that make OS services reachable to the hardware environment. Each and every application present in the system requires the Operating System. The operating system works as a communication channel between system hardware and system software. The operating system helps an application with the hardware part without knowing about the actual hardware configuration. It is one of the most important parts of the system and hence it is present in every device, whether large or small device.



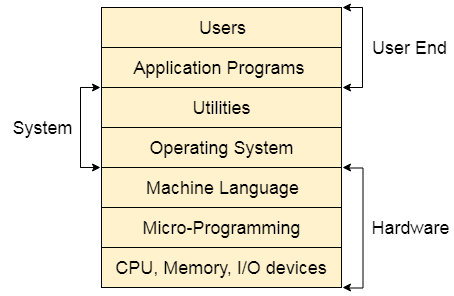
## **Functions of the Operating System**

* **Resource Management:**The operating system manages and allocates memory, CPU time, and other hardware resources among the various programs and processes running on the computer.
* **Process Management:**The operating system is responsible for starting, stopping, and managing processes and programs. It also controls the scheduling of processes and allocates resources to them.
* **Memory Management:** The operating system manages the computer’s primary memory and provides mechanisms for optimizing memory usage.
* **Security:**The operating system provides a secure environment for the user, applications, and data by implementing security policies and mechanisms such as access controls and encryption.
* **Job Accounting:**It keeps track of time and resources used by various jobs or users.
* **File Management:**The operating system is responsible for organizing and managing the file system, including the creation, deletion, and manipulation of files and directories.
* **Device Management:** The operating system manages input/output devices such as printers, keyboards, mice, and displays. It provides the necessary drivers and interfaces to enable communication between the devices and the computer.
* **Networking:** The operating system provides networking capabilities such as establishing and managing network connections, handling network protocols, and sharing resources such as printers and files over a network.
* **User Interface**: The operating system provides a user interface that enables users to interact with the computer system. or a combination of both.
* **Backup and Recovery:** The operating system provides mechanisms for backing up data and recovering it in case of system failures, errors, or disasters.
* **Virtualization:**The operating system provides virtualization capabilities that allow multiple operating systems or applications to run on a single physical machine. This can enable efficient use of resources and flexibility in managing workloads.
* **Performance Monitoring:** The operating system provides tools for monitoring and optimizing system performance, including identifying bottlenecks, optimizing resource usage, and analyzing system logs and metrics.
* **Time-Sharing:** The operating system enables multiple users to share a computer system and its resources simultaneously by providing time-sharing mechanisms that allocate resources fairly and efficiently.
* **System Calls:** The operating system provides a set of system calls that enable applications to interact with the operating system and access its resources. System calls provide a standardized interface between applications and the operating system, enabling portability and compatibility across different hardware and software platforms.
* **Error-detecting Aids:**These contain methods that include the production of dumps, traces, error messages, and other debugging and error-detecting methods.

## **Structure of a Computer System**

A Computer System consists of:

* Users (people who are using the computer)
* Application Programs (Compilers, Databases, Games, Video player, Browsers, etc.)
* System Programs (Shells, Editors, Compilers, etc.)
* Operating System ( A special program which acts as an interface between user and hardware )
* Hardware ( CPU, Disks, Memory, etc)



## **Objectives of Operating Systems**

* **Convenient to use:** One of the objectives is to make the computer system more convenient to use in an efficient manner.
* **User Friendly:** To make the computer system more interactive with a more convenient interface for the users.
* **Easy Access:** To provide easy access to users for using resources by acting as an intermediary between the hardware and its users**.**
* **Management of Resources:**For managing the resources of a computer in a better and faster way.
* **Controls and Monitoring:** By keeping track of who is using which resource, granting resource requests, and mediating conflicting requests from different programs and users.
* **Fair Sharing of Resources:** Providing efficient and fair sharing of resources between the users and programs.

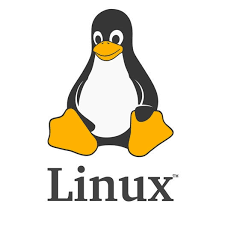
## **Types of Operating Systems**

* **Batch Operating System:**A [Batch Operating System](https://www.geeksforgeeks.org/batch-processing-operating-system/) is a type of operating system that does not interact with the computer directly. There is an operator who takes similar jobs having the same requirements and groups them into batches.
* **Time-sharing Operating System:**[Time-sharing Operating System](https://www.geeksforgeeks.org/time-sharing-operating-system/) is a type of operating system that allows many users to share computer resources (maximum utilization of the resources).
* **Distributed Operating System:**[Distributed Operating System](https://www.geeksforgeeks.org/what-is-a-distributed-system/) is a type of operating system that manages a group of different computers and makes appear to be a single computer. These operating systems are designed to operate on a network of computers. They allow multiple users to access shared resources and communicate with each other over the network. Examples include Microsoft Windows Server and various distributions of Linux designed for servers.
* **Network Operating System:**[Network Operating System](https://www.geeksforgeeks.org/what-is-a-network-operating-system/) is a type of operating system that runs on a server and provides the capability to manage data, users, groups, security, applications, and other networking functions.
* **Multiprocessing Operating System:**[Multiprocessor Operating Systems](https://www.geeksforgeeks.org/introduction-of-multiprocessor-and-multicomputer/) are used in operating systems to boost the performance of multiple CPUs within a single computer system. Multiple CPUs are linked together so that a job can be divided and executed more quickly.
* **Single-User Operating Systems:**[Single-User Operating Systems](https://www.geeksforgeeks.org/single-user-operating-system/) are designed to support a single user at a time. Examples include Microsoft Windows for personal computers and Apple macOS.
* **Multi-User Operating Systems:**[Multi-User Operating Systems](https://www.geeksforgeeks.org/multi-user-operating-system/) are designed to support multiple users simultaneously. Examples include Linux and Unix.
* **Embedded Operating Systems:**[Embedded Operating Systems](https://www.geeksforgeeks.org/embedded-real-time-system/)are designed to run on devices with limited resources, such as smartphones, wearable devices, and household appliances. Examples include Google’s Android and Apple’s iOS.
* **Cluster Operating Systems:** Cluster Operating Systems are designed to run on a group of computers, or a cluster, to work together as a single system. They are used for high-performance computing and for applications that require high availability and reliability. Examples include Rocks Cluster Distribution and OpenMPI.
* **Real-time Operating System:**[Real-time Operating System](https://www.geeksforgeeks.org/real-time-operating-system-rtos/) is a type of operating system that serves a real-time system and the time interval required to process and respond to inputs is very small. These operating systems are designed to respond to events in real time. They are used in applications that require quick and deterministic responses, such as embedded systems, industrial control systems, and robotics

## **Examples of Operating Systems**

* **Windows**(GUI-based, PC)
* **GNU/Linux** (Personal, Workstations, ISP, File, and print server, Three-tier client/Server)
* **macOS** (Macintosh), used for Apple’s personal computers and workstations (MacBook, iMac).
* **Android** (Google’s Operating System for smartphones/tablets/smartwatches)
* **iOS** (Apple’s OS for iPhone, iPad, and iPod Touch)

Linux



**Linux** ([/ˈlɪnʊks/](https://en.wikipedia.org/wiki/Help:IPA/English) [*LIN-uuks*](https://en.wikipedia.org/wiki/Help:Pronunciation_respelling_key)) is a family of [open-source](https://en.wikipedia.org/wiki/Free_and_open-source_software) [Unix-like](https://en.wikipedia.org/wiki/Unix-like) [operating systems](https://en.wikipedia.org/wiki/Operating_system) based on the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel), an [operating system kernel](https://en.wikipedia.org/wiki/Kernel_(operating_system)) first released on September 17, 1991, by [Linus Torvalds](https://en.wikipedia.org/wiki/Linus_Torvalds). Linux is typically [packaged](https://en.wikipedia.org/wiki/Package_manager) as a [Linux distribution](https://en.wikipedia.org/wiki/Linux_distribution) (distro), which includes the kernel and supporting [system software](https://en.wikipedia.org/wiki/System_software) and [libraries](https://en.wikipedia.org/wiki/Library_(computing)), many of which are provided by the [GNU Project](https://en.wikipedia.org/wiki/GNU_Project). Many Linux distributions use the word "Linux" in their name, but the [Free Software Foundation](https://en.wikipedia.org/wiki/Free_Software_Foundation) uses and recommends the name "**GNU/Linux**" to emphasize the use and importance of [GNU](https://en.wikipedia.org/wiki/GNU) software in many distributions, [causing some controversy](https://en.wikipedia.org/wiki/GNU/Linux_naming_controversy).

Linux was originally developed for [personal computers](https://en.wikipedia.org/wiki/Personal_computer) based on the [Intel x86](https://en.wikipedia.org/wiki/Intel_x86) architecture, but has since been [ported](https://en.wikipedia.org/wiki/Porting) to more [platforms](https://en.wikipedia.org/wiki/Computer_hardware_platforms) than any other operating system. Because of the dominance of Linux-based [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) on [smartphones](https://en.wikipedia.org/wiki/Smartphone), Linux, including Android, has the [largest](https://en.wikipedia.org/wiki/Usage_share_of_operating_systems) installed base of all [general-purpose operating systems](https://en.wikipedia.org/wiki/General-purpose_operating_system) as of May 2022. Although Linux is, as of November 2022, used by only around 2.6 percent of [desktop computers](https://en.wikipedia.org/wiki/Desktop_computer), the [Chromebook](https://en.wikipedia.org/wiki/Chromebook), which runs the Linux kernel-based [ChromeOS](https://en.wikipedia.org/wiki/ChromeOS" \o "ChromeOS), dominates the US [K–12](https://en.wikipedia.org/wiki/K%E2%80%9312) education market and represents nearly 20 percent of sub-$300 [notebook](https://en.wikipedia.org/wiki/Laptop) sales in the US.[[27]](https://en.wikipedia.org/wiki/Linux#cite_note-30) Linux is the leading operating system on servers (over 96.4% of the top one million web servers' operating systems are Linux),[[28]](https://en.wikipedia.org/wiki/Linux#cite_note-31) leads other [big iron](https://en.wikipedia.org/wiki/Big_iron_(computing)) systems such as [mainframe computers](https://en.wikipedia.org/wiki/Mainframe_computer), and is used on all of the [world's 500 fastest supercomputers](https://en.wikipedia.org/wiki/TOP500)[[d]](https://en.wikipedia.org/wiki/Linux#cite_note-32) (as of November 2017, having gradually displaced all competitors)

## **How is the Linux operating system used?**

Every version of the Linux OS manages hardware resources, launches and handles applications, and provides some form of user interface. The enormous community for developers and wide range of distributions means that a Linux version is available for almost any task, and Linux has penetrated many areas of computing.

For example, Linux has emerged as a popular OS for [web servers](https://www.techtarget.com/whatis/definition/Web-server) such as Apache, as well as for network operations, scientific computing tasks that require huge compute clusters, running databases, desktop and endpoint computing, and running mobile devices with OS versions like Android.

The Linux OS can be found in many different settings, supporting many different use cases. Linux is used in the following ways:

* **Server OS** for web servers, database servers, file servers, email servers and any other type of shared server. Designed to support high-volume and multithreading applications, Linux is well-suited for [all types of server applications](https://www.techtarget.com/searchdatacenter/tip/Choosing-the-best-server-OS-Linux-vs-Windows-comparisons).
* **Desktop OS** for personal productivity computing. Linux is an open source and freely available [desktop environment](https://www.techtarget.com/searchenterprisedesktop/tip/Top-5-enterprise-Linux-distributions-to-consider-adopting) for users who prefer it to commercial OSes.
* **Headless server OS** for systems that do not require a graphical user interface (GUI) or directly connected terminal and keyboard. Headless systems are often used for remotely managed networking server and other devices.
* **Embedded device or appliance OS** for systems that require limited computing function. Linux is used as an embedded OS for a variety of applications, including household appliances, automotive entertainment systems and network file system appliances.
* **Network OS** for routers, switches, [domain name system](https://www.techtarget.com/searchnetworking/definition/domain-name-system) servers, home networking devices and more. For example, Cisco offers a version of the Cisco Internetwork Operating System ([IOS](https://www.techtarget.com/searchnetworking/definition/Cisco-IOS-Cisco-Internetwork-Operating-System)) that uses the Linux [kernel](https://www.techtarget.com/searchdatacenter/definition/kernel).
* **Software development OS** for enterprise software development. Although many development tools have been ported to Windows or other OSes, Linux is home to some of the most widely used [open source software development tools](https://www.techtarget.com/searchapparchitecture/tip/Explore-10-popular-open-source-development-tools). For example, git for distributed source control; vim and emacs for source code editing; and compilers and interpreters for almost every programming language.
* **Cloud OS** for cloud instances. Major [cloud computing](https://www.techtarget.com/searchcloudcomputing/definition/cloud-computing) providers offer access to cloud computing instances running Linux for cloud servers, desktops and other services.

Linux is highly configurable and depends on a modular design that enables users to customize their own versions of Linux. Depending on the application, Linux can be optimized for different purposes such as:

* networking performance;
* computation performance;
* deployment on specific hardware platforms; and
* deployment on systems with limited memory, storage or computing resources.

Users can choose different Linux distributions for specific applications or adapt a specific distribution to incorporate custom kernel configurations.

## **Linux distributions**

Since its initial development, Linux has adopted the [copyleft](https://www.techtarget.com/whatis/definition/copyleft) stipulations of the Free Software Foundation which originated the [GNU GPL](https://www.techtarget.com/searchdatacenter/definition/GNU-General-Public-License-GNU-GPL-or-simply-GPL). The GPL says that anything taken for free and modified must be distributed for free. In practice, if Linux or other GNU-licensed components are developed or modified to create a new version of Linux, that new version must be distributed for free. This prevents a developer or other groups from profiting unfairly from the freely available work of others.

LINUX

Tux, the Linux mascot.

Hundreds of different Linux versions, also known as distributions or distros, are available. Distributions usually distinguish themselves from the pack by addressing a specific goal, philosophy, function or target market.

There are distributions tailored for specific target functions, such as servers, desktops, gaming, security, or embedded devices including [Raspberry Pi](https://www.techtarget.com/whatis/definition/Raspberry-Pi-35-computer) systems. Most modern distributions are precompiled and ready to use, while others like Gentoo Linux consist of source code that a user can compile locally during initial installation to optimize their system configuration. Knoppix Linux is one of many distros used to recover damaged hard drives and perform other technical support tasks. Information security professionals use [Kali Linux for penetration testing](https://www.techtarget.com/searchsecurity/feature/Top-tips-for-using-the-Kali-Linux-pen-testing-distribution) and other security-related tasks.

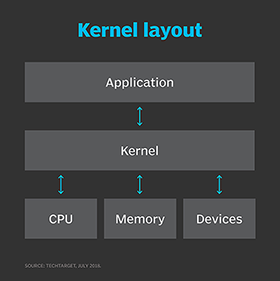
Linux has become an important component of automaker entertainment systems. Many automakers have joined [Automotive Grade Linux](https://www.automotivelinux.org/) (AGL), an open source project hosted by the Linux Foundation. For example, Toyota and Lexus vehicles use AGL for their infotainment systems.

Linux distributions may be community-developed, like [Debian](https://www.techtarget.com/searchdatacenter/definition/Debian), Slackware and Gentoo. Other distributions are commercial and intended for enterprise use, including Red Hat Enterprise Linux and [SUSE](https://www.techtarget.com/searchdatacenter/definition/SuSE) Linux Enterprise Server. Many distributions use a combination of community- and corporate-supported development, such as Red Hat's Fedora, openSUSE from SUSE and [Ubuntu](https://www.techtarget.com/searchdatacenter/definition/Ubuntu) from Canonical.

## **Linux components and terminology**

The Linux OS system incorporates several different components, including:

* [**Bootloader**](https://www.techtarget.com/searchdatacenter/definition/boot-loader-boot-manager)**.**A bootloader is responsible for managing the boot process of the computer and for starting the Linux kernel. It can also be used to manage systems that boot more than one OS.
* **Kernel.** The core of the Linux system, the kernel handles network access, schedules processes or applications, manages basic peripheral devices and oversees all file system services. The Linux kernel is the software that interfaces directly with the computer hardware.
* **Init system.**The first process to run once the kernel is loaded. A process is an instance of a program running on a computer, and the init system initializes the system to enable it to run other processes. Init is a daemon program that acts as the parent process to all other processes running on the system. Init can be configured to start specific processes at system initialization. For example, when the system will be running a web server the init system can be configured to load all necessary web server software.
* **Daemons.**This is a program that runs in the background, handling requests for a service. A web server running on a Linux server depends on a daemon, usually named httpd, to listen for web server requests.
* **Graphical server.**This is the software that controls how graphics are displayed on a computer. Without a graphical server, users can only interact with the Linux system through a command-line interface. The [X Window System](https://www.techtarget.com/whatis/definition/X-Window-System-X-or-XWindows), also known as X11 or X, is the most common graphical server for Linux, though not the only one. X runs as a server daemon on the system and is called upon by applications when graphical output is required.
* **Desktop environment.**This is the collection of applications and user interface controls with which users interact when using Linux as a desktop platform. Access to the desktop environment is usually controlled through the X Window System or another graphical system. Each desktop environment defines its own look, including the way graphical elements like windows, pull-down menus and files are displayed and manipulated. The desktop environment will also include a set of default applications for managing files and folders, text editing, running a command-line session and other common tasks.
* **Applications.**This is the software that is installed during and after the initial Linux installation. Most Linux distributions include thousands of different applications, including both for a networked server and for desktop use.



The Linux kernel mediates interaction between applications and system hardware such as CPU, memory and devices like storage or printers.

While these components are included in most Linux distributions, they are not necessarily part of every deployed Linux system. For example, a Linux-based server may not require a graphical server, desktop environment or applications.

But it is really the many outside developers and GNU projects that offer high-level functions to the Linux kernel to provide a fully realized OS. For example, there are modules to provide a command-line interface, implement a GUI, manage security, and offer video input or audio services -- each of which can be modified and optimized to form unique distributions for specific tasks.

Desktop environments can also vary widely with different approaches to GUI design and default applications. The two most popular desktop environments are:

* The [GNOME](https://www.techtarget.com/searchdatacenter/definition/GNOME-GNU-Network-Object-Model-Environment) desktop environment, which is included in most popular Linux distributions, is the default desktop for many. Designed to be easy to use and reliable, GNOME spawned other desktop environment projects, including MATE, Cinnamon and Unity.
* The KDE desktop environment is the primary alternative to GNOME. KDE is also designed to be easy to use and reliable. It has spawned other projects, including the Trinity Desktop Environment.

Package manager software typically adds, updates or removes software components under the Linux OS. Package managers enable users to install additional software not included with their distributions. Examples of software package managers include [RPM Package Manager](https://www.techtarget.com/searchdatacenter/definition/RPM-Package-Manager-Red-hat-Package-Manager), dpkg, OpenPKG and Zero Install.

## **How the Linux operating system works**

The Linux OS follows a modular design that is the key to its many variations and distributions. All Linux distributions are based on the Linux kernel, but they can differ depending on factors such as:

* **Kernel version.**Distributions can be configured with more recent versions to incorporate newer features or with older versions to be more stable.
* **Kernel modules.** This is software that can be loaded and unloaded into the kernel to extend functionality without rebooting. Kernel modules are often used to support:
  + [device drivers](https://www.techtarget.com/searchenterprisedesktop/definition/device-driver), which use code that controls how attached devices operate;
  + file system drivers, which use code that controls how the kernel works with different file systems; and
  + system calls, which use code that controls how programs request services from the kernel.
* **Configuration options.**Linux kernels compiled with configuration options set to include only device or file system drivers are used for some specialized distributions; for example, compiling a kernel for a wireless device without any wired network device drivers.

The Linux kernel is the one thing that all systems running Linux have in common. Linux works by:

* Loading and booting a Linux kernel.
* Once booted, the kernel manages all system input and output. The system is initialized, and processes can be started.
* As system processes are started, the system can be used for processes that include network server functions, commands entered interactively via command line, desktop applications or any application or program.

While the kernel may be almost identical -- with some divergence for configuration and compilation differences -- the user experience can vary widely, depending on how the Linux system is being used. For example, some Linux use cases with widely different user experiences include:

* **Desktop productivity** systems, such as those used by software developers or other professionals. Software development workstations may be optimized for performance, while desktops for administrative professionals may be optimized for use of desktop productivity tools.
* **Network servers** may not even include a terminal for direct access. These [headless servers](https://www.techtarget.com/whatis/definition/headless-server) are managed remotely through network terminal or Windows sessions. Servers may be used by many but should be directly accessed only by authorized system admins.
* **Thin clients** enable users to access a rich desktop environment from a lightweight device. This includes Raspberry Pi single-card computers and Google Chromebooks.

**Advantages**

* **Open source software.**The Linux kernel is released under the GNU GPL open source software license. Most distros include hundreds of applications, with many options in almost every category. Many distributions also include proprietary software, such as device drivers provided by manufacturers, to support their hardware.
* **Licensing costs.**Unlike Microsoft Windows or Apple macOS, Linux has no explicit licensing fees. While system support is available for a fee from many Linux vendors, the OS itself is free to copy and use. Some IT organizations have increased their savings by switching their server software from a commercial OS to Linux.
* **Reliability.**Linux is considered a reliable OS and is well-supported with security patches. Linux is also considered to be stable, meaning it can run in most circumstances. Linux also copes with errors when running software and unexpected input.
* **Backward compatibility.**Linux and other open source software tend to be updated frequently for security and functional patches, while retaining core functionality. Configurations and shell scripts are likely to work unchanged even when software updates are applied. Unlike commercial software vendors that roll out new versions of their OSes along with new ways to work, Linux and open source applications generally don't change their modes of operation with new releases.
* **Many choices.**Between the hundreds of available distributions, thousands of applications and almost infinite options for configuring, compiling and running Linux on almost any hardware platform, it is possible to optimize Linux for almost any application.

**Disadvantages**

* **Lack of established standard.**There is no standard version of Linux, which may be good for optimizing Linux for particular applications, but less so for deploying standardized server or desktop images. The wide range of options can complicate support as a result.
* **Support costs.**While an organization can acquire Linux freely without licensing fees, support is not free. Most enterprise Linux distributors like SUSE and Red Hat offer support contracts. Depending on the circumstances, these license fees can reduce savings significantly.
* **Proprietary software.** Desktop productivity software like Microsoft Office cannot be used on Linux desktops, and other proprietary software may be unavailable for Linux platforms.
* **Unsupported hardware.** While many hardware manufacturers make Linux device drivers available for their products, many do not.
* **Steep learning curve.** Many users struggle to learn to use the Linux desktop or Linux-based applications.

In some cases, the same Linux attribute can be either an advantage or disadvantage. For example, having many options for customizing the Linux OS is advantageous for manufacturers looking for an embedded OS, but it is a disadvantage for enterprises that want a desktop OS that can be used by a wide range of end users.

**WINDOWS**

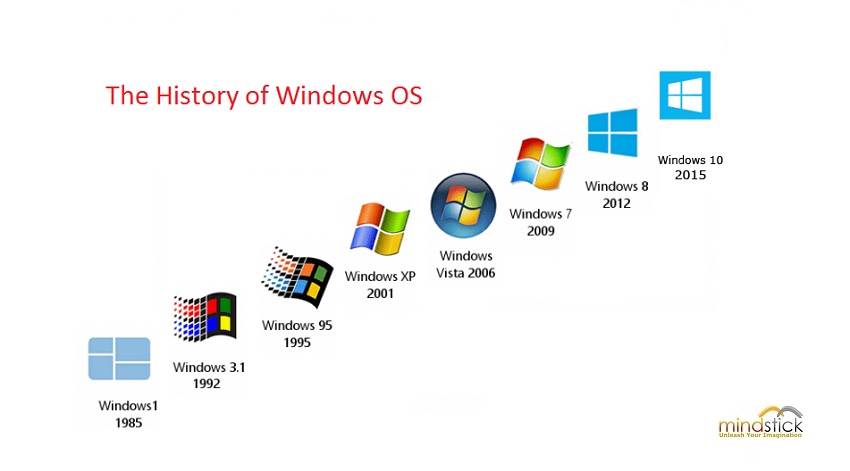
Windows is a **graphical operating system** developed by Microsoft. It allows users to view and store files, run the software, play games, watch videos, and provides a way to connect to the internet. It was released for both home computing and professional works.

Microsoft introduced the first version as 1.0

It was released for both home computing and professional functions of Windows on **10 November 1983**. Later, it was released on many versions of Windows as well as the current version, Windows 10.

In 1993, the first business-oriented version of Windows was released, which is known as **Windows NT 3.1**. Then it introduced the next versions, **Windows 3.5**, **4/0**, and **Windows 2000**. When the XP Windows was released by Microsoft in 2001, the company designed its various versions for a personal and business environment. It was designed based on standard x86 hardware, like **Intel** and **AMD processor**. Accordingly, it can run on different brands of hardware, such as HP, Dell, and Sony computers, including home-built PCs.

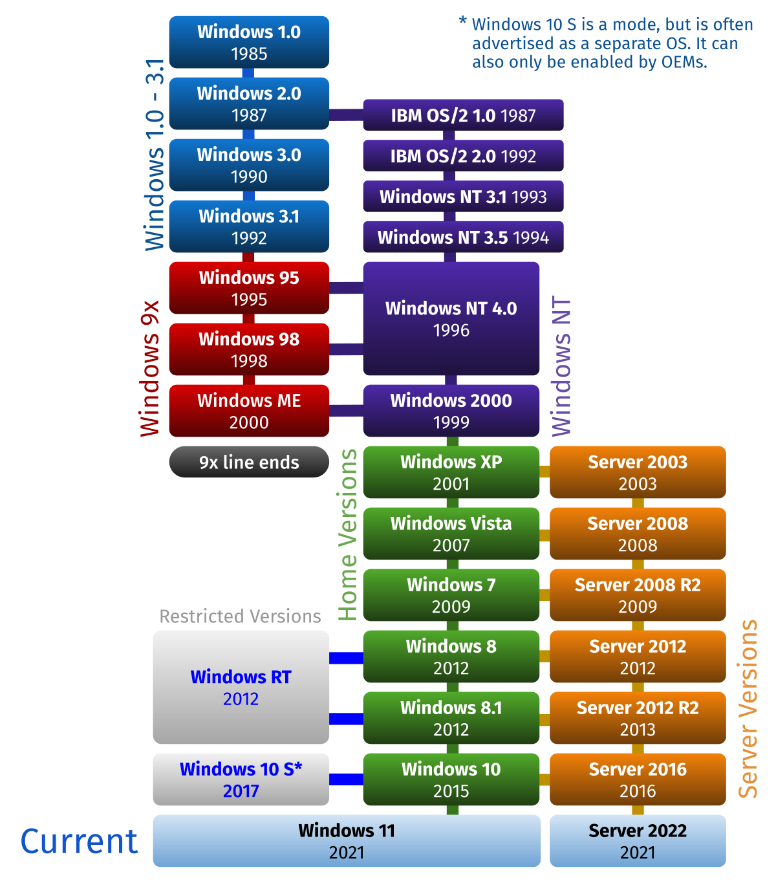
**HISTORY OF WINDOWS**



The first version of Windows, released in 1985, was simply a GUI offered as an [extension](https://www.britannica.com/technology/plug-in) of Microsoft’s existing disk operating system, or [MS-DOS](https://www.britannica.com/technology/MS-DOS). Based in part on licensed concepts that [Apple Inc.](https://www.britannica.com/topic/Apple-Inc) had used for its Macintosh System Software, Windows for the first time allowed DOS users to visually navigate a virtual desktop, opening graphical “windows” displaying the contents of electronic folders and files with the click of a [mouse](https://www.britannica.com/technology/mouse-computer-device) button, rather than typing commands and directory paths at a text prompt.

With the 2001 release of [Windows XP](https://www.britannica.com/technology/Windows-XP), Microsoft united its various Windows packages under a single banner, offering multiple editions for consumers, businesses, multimedia developers, and others. Windows XP abandoned the long-used Windows 95 kernel (core [software](https://www.britannica.com/technology/software) code) for a more powerful code base and offered a more practical interface and improved application and memory management. The highly successful XP standard was succeeded in late 2006 by [Windows Vista](https://www.britannica.com/technology/Windows-Vista), which experienced a troubled rollout and met with considerable marketplace resistance, quickly acquiring a reputation for being a large, slow, and resource-consuming system. Responding to Vista’s disappointing adoption rate, Microsoft in 2009 released [Windows 7](https://www.britannica.com/technology/Windows-7), an OS whose interface was similar to that of [Vista](https://www.britannica.com/dictionary/Vista) but was met with enthusiasm for its noticeable speed improvement and its modest system requirements.

[Windows 8](https://www.britannica.com/technology/Windows-8) in 2012 offered a start screen with applications appearing as tiles on a grid and the ability to synchronize settings so users could log on to another Windows 8 [machine](https://www.britannica.com/technology/machine) and use their preferred settings. In 2015 Microsoft released [Windows 10](https://www.britannica.com/technology/Windows-10), which came with [Cortana](https://www.britannica.com/topic/Cortana), a digital personal assistant like [Apple](https://www.britannica.com/topic/Apple-Inc)’s Siri, and the Web browser Microsoft Edge, which replaced Internet Explorer. Microsoft also announced that Windows 10 would be the last version of Windows, meaning that users would receive regular updates to the OS but that no more large-scale revisions would be done

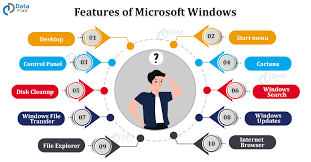


After Windows 95, Microsoft released several other versions of Windows, including Windows 98, Windows 2000, Windows XP, Windows Vista, Windows 7, Windows 8, and Windows 10. Each version of this OS offered different improvements and fixes, including performance improvements, enhanced interface, support for new hardware and software, and better security features.

The Windows operating system has undergone incredible evolution since it was first launched in 1985. Each version of Windows offers different improvements and fixes, and this operating system remains one of the most popular in the world to this day.

## **Functions of the Windows Operating System**

* The functions of the Windows operating system are crucial in running various applications and programs on computers or laptops. Some of the functions of the Windows operating system include:
* Managing Computer Resources: The primary function of the Windows operating system is to manage and organize computer resources such as CPU, RAM, and hard disk. The Windows operating system will complete various tasks such as opening applications, accessing the internet, and printing documents using these resources.



* Providing an Interface: The Windows operating system provides a [Graphical User Interface](https://it.telkomuniversity.ac.id/tag/graphical-user-interface/) (GUI) that allows users to access and use various applications and programs easily. This interface allows users to select menus, click icons, and navigate various applications easily.
* Providing Compatibility: The Windows operating system is designed to support various hardware and software devices, making it easier for users to install and use different applications and programs on their computers or laptops. Windows also provides the ability to run programs and applications designed for different versions of the Windows operating system.
* Facilitating Network Setup: Windows operating system provides various tools to facilitate network setup, such as network configuration, firewall configuration, and network access configuration. This allows users to connect to networks and share data with other devices in the network easily.
* Facilitating Security: The Windows operating system provides various security features such as anti-virus, anti-malware, and firewall that help protect computers or laptops from virus and malware attacks. Windows also provides tools to configure network security and access control to protect user’s important data.
* Managing File Management: Windows also facilitates file management such as data storage, access rights configuration, and file searching. Windows provides various tools to help users organize and store data and files on their computers or laptops.
* Facilitating System Setup: Windows provides various tools for system setup such as desktop appearance configuration, sound and visual effects configuration, and operating system updates. This makes it easy for users to configure and update the operating system according to their needs and preferences.

## **Advantages and Disadvantages of Windows Operating System**

As the most widely used operating system in the world, Windows has various advantages and disadvantages that need to be considered. Here are some of the advantages and disadvantages of the Windows operating system:

### Advantages:

* Wide Compatibility: Windows has extensive compatibility with various hardware and software devices, making it possible for users to use different types of devices with this operating system.
* Ease of Use: Windows is designed with a user-friendly and familiar user interface for many people, making it easy to learn and use even for new users.
* Rich Multimedia Features: Windows provides many multimedia features, such as Windows Media Player, that allow users to play various types of audio and video files easily.
* Large Developer Support: There are many developers who make applications and games for Windows, so users can choose from a variety of application and game options available.
* Long-term Support: Microsoft provides long-term support for each version of Windows, which means users can receive updates and security support for several years after its release.

### Disadvantages:

* Vulnerability to Malware: As the most widely used operating system in the world, Windows is also more vulnerable to virus and malware attacks, so users need to be more cautious in using it.
* Hardware Requirements: Windows requires relatively high hardware requirements, making it difficult for users with older computers or laptops to use the latest version of Windows.
* Cost: Windows is not a free operating system, so users need to purchase a license to use it, which can be expensive.
* Having too many features: In some cases, Windows can be too complicated with too many features, so users have to take the time to learn and understand these features.
* In conclusion, Windows OS has its strengths and weaknesses like any other operating system. However, its wide compatibility and large developer support make it the top choice for many users. Users should take note of these shortcomings and ensure to take necessary security measures to protect their devices from virus and malware attac

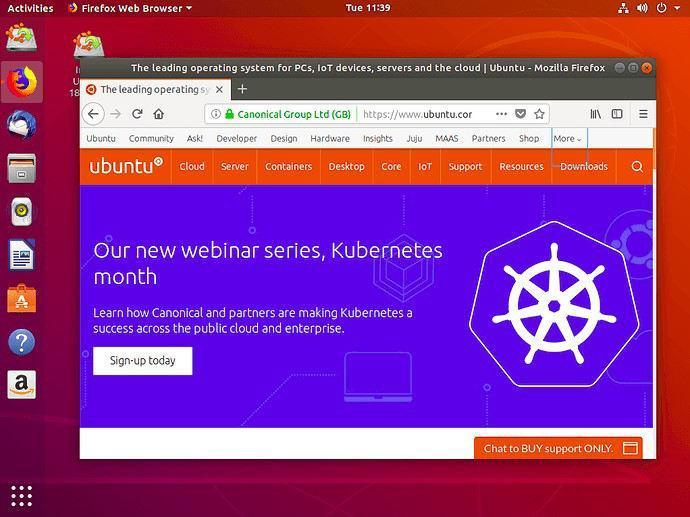
Windows vs Unix

|  |  |  |
| --- | --- | --- |
| Parameter | unix | windows |
| **Licensing** | UNIX was developed as an open-source OS using C and Assembly languages. Since being open source UNIX, and its various Linux distributions account for the most used OS in the world.  Unix and all its Linux distributions are available under the General Public License. | Windows Operating System is proprietary software owned by Microsoft, meaning its source code is not available to the public. |
| User Interface: | Unix operating systems are relatively harder to grasp and produce significant barriers for newcomers. However, some Linux distributions like Ubuntu are changing such perceptions by bringing in more GUI-based applications. | Windows Operating System on the other hand is designed with the outset of keeping the UI as simple and user-friendly as possible, so that non-IT people can easily use computers for their work. |
| Case Sensitivity | Unix is completely case-sensitive, meaning the files with names “GFG.txt” and “gfg.txt” can be considered two different files. | Windows, on the other hand, have optional case sensitivity. |
| Security | It is more secure as all changes to the system require explicit user permission. | It is less secure compared to UNIX. |
| **Reliability** | Unix and its distributions are well known for being very stable to run. | Although Windows has been stable in recent years,  it is still to match the stability provided by Unix systems |
| **Data Backup & Recovery** | It is tedious to create a backup and recovery system in UNIX | It has an integrated backup and recovery system that make it simpler to use |

**Installation of Ubuntu.**

Following are the steps to install ubuntU

**1. Overview**



The Ubuntu desktop is easy to use, easy to install and includes everything WE need to run our organisation, school, home or enterprise. It’s also open source, secure, accessible and free to download.

**2. Requirements**

Following points should be considered before starting the installation:

* Connect the laptop to a power source.

* Ensure that we must have at least 25 GB of free storage space, or 5 GB for a minimal installation.

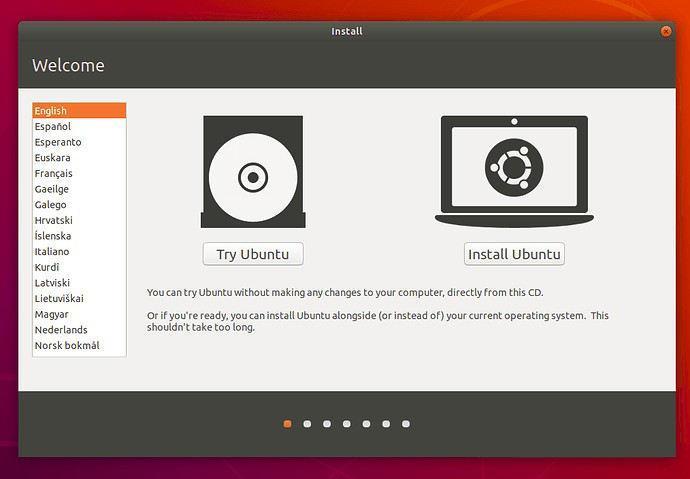
* Have access to a USB flash drive containing the version of Ubuntu we want to install.

* Make sure we have a recent backup of our data. While it’s unlikely that anything will go wrong, you can never be too prepared.

12

**3. Boot from USB flash drive**

Most computers will boot from USB automatically. Simply insert the USB flash drive and either power on the computer or restart it. We should see welcome window prompting us to choose our language and either install or try the Ubuntu desktop.



Depending on our computer’s configuration, we may instead see an alternative boot menu showing a large language selection pane. Use the mouse or cursor keys to select a language and we’ll be presented with a simple menu.



13

Select the second option, ‘Install Ubuntu’, and press return to launch the desktop installer automatically. Alternatively, select the first option, ‘Try Ubuntu without installing’, to test Ubuntu.

A few moments later, after the desktop has loaded, we’ll see the welcome window. From here, we can select our language from a list on the left and choose between either installing Ubuntu directly, or trying the desktop first.

If computer doesn’t automatically boot from USB, try holding F12 when the computer first starts. With most machines, this will allow us to select the USB device from a system-specific boot menu.

F12 is the most common key for bringing up our system’s boot menu, but Escape, F2 and F10 are common alternatives. If unsure, look for a brief message when the system starts - this will often inform us of which key to press to bring up the boot menu.

**4. Prepare to install Ubuntu**

We will first be asked to select our keyboard layout. If the installer doesn’t guess the default layout correctly, use the ‘Detect Keyboard Layout’ button to run through a brief configuration procedure.

After selecting *Continue* we’ll be asked *What apps would you like to install to start with?* The two options are ‘Normal installation’ and ‘Minimal installation’. The first is the equivalent to the old default bundle of utilities, applications, games and media players — a great launchpad for any Linux installation. The second takes considerably less storage space and allows us to install only what we need.

Beneath the installation-type question are two checkboxes; one to enable updates while installing and another to enable third-party software.

* It’s advised enabling both Download updates and Install third-party software.

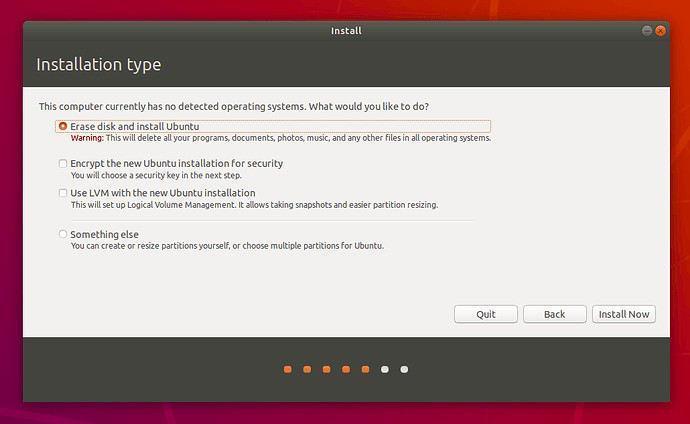
* Stay connected to the internet so that the latest updates could be known while we install Ubuntu.

* If not connected to the internet, we will be asked to select a wireless network, if available. It’s advised to connect during the installation so we can ensure that our machine is up to date.

1. **Allocate drive space**

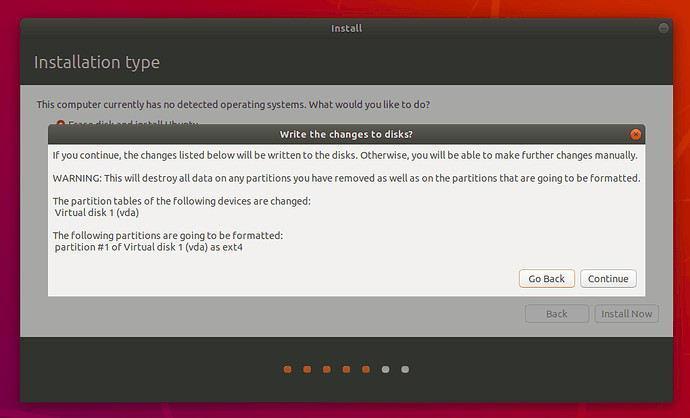
Use the checkboxes to choose whether we’d like to install Ubuntu alongside another operating system, delete our existing operating system and replace it with Ubuntu, or — if we’re an advanced user — choose the ’Something else’ option.

14



**6. Begin installation**

After configuring storage, click on the ‘Install Now’ button. A small pane will appear with an overview of the storage options we’ve chosen, with the chance to go back if the details are incorrect. Click Continue to fix those changes in place and start the installation process.

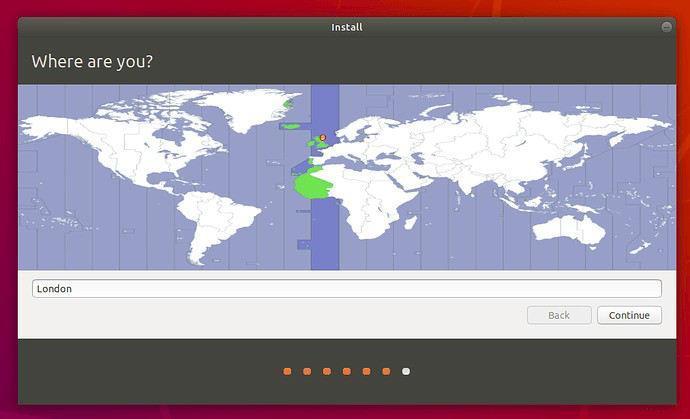


15

**7. Select your location**

If connected to the internet, our location will be detected automatically. Check if the location is correct and click ’Forward’ to proceed.

If unsure about the time zone, type the name of a local town or city or use the map to select the location.



If having problems while connecting to the Internet, use the menu in the top-right-hand corner to select a network.

**8. Login details**

Enter name and the installer will automatically suggest a computer name and username. These can easily be changed if we prefer. The computer name is how our computer will appear on the network, while our username will be our login and account name.

Next, enter a strong password. The installer will let us know if it’s too weak.

We can also choose to enable automatic login and home folder encryption. If our machine is portable, it’s recommend keeping automatic login disabled and enabling encryption. This should stop people from accessing our personal files if the machine is lost or stolen.

If we enable home folder encryption and forget our password, we won’t be able to retrieve any personal data stored in our home folder.

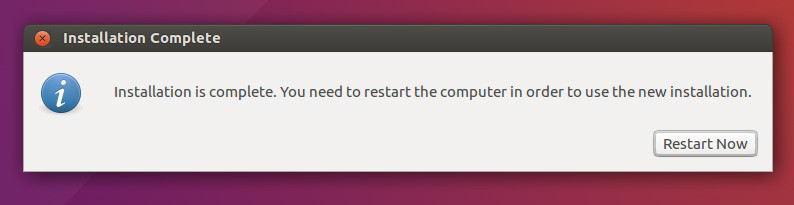
**9. Background installation**

The installer will now complete in the background while the installation window teaches us a little about how awesome Ubuntu is. Depending on the speed of our machine and network connection, installation should only take a few minutes.



**11. Installation complete**

After everything has been installed and configured, a small window will appear asking us to restart our machine. Click on Restart Now and remove the USB flash drive when prompted. If we initiated the installation while testing the desktop, we can also get the option to continue testing.

  
  
  
Linux has been successfully installed.

**PRACTICAL-2**

**OBJECTIVE:**  To implement the following Linux commands :

man, echo, cat, touch, ls, mkdir, cd, cp, pwd, tty, who, whoami,wc,mv, rm, rmdir

**LINUX**

**COMMANDS:**

1. **man**

**Description:**

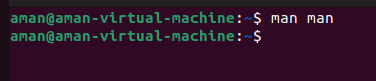
**man** is the system's manual pager. Each page argument given to man is normally the name of a program, utility or function. The manual page associated with each of these arguments is then found and displayed. A section, if provided, will direct man to look only in that section of the manual. The default action is to search in all of the available sections, following a predefined order and to show only the first page found, even if page exists in several sections.

**Syntax :**

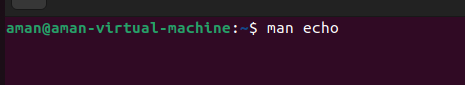
man [OPTION]... [COMMAND NAME]...

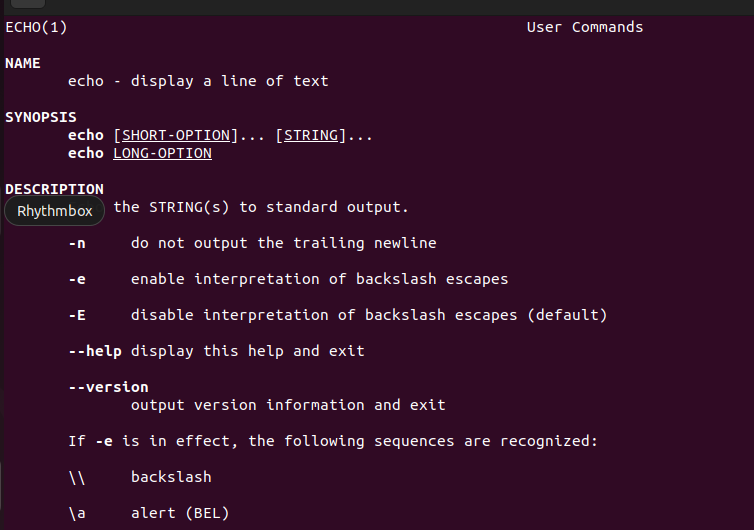
|  |  |  |
| --- | --- | --- |
| **Tag** | **Description** |  |
|  |  |  |
|  | It displays the whole manual of the command. |  |
| **No Option** | **Syntax :** |  |
|  | man [COMMAND NAME] |  |
|  |  |  |
|  | Since a manual is divided into multiple sections so this |  |
|  | option is used to display only a specific section of a |  |
| **Section-num:** | manual. |  |
|  | **Syntax :** |  |
|  | man [SECTION-NUM] [COMMAND NAME] |  |
|  |  |  |
|  | One may not be able to remember the sections in which a |  |
|  | command is present. So this option gives the section in |  |
| **-f option** | which the given command is present. |  |
|  | **Syntax :** |  |
|  | man -f [COMMAND NAME] |  |
|  |  |  |
|  | This option helps us to display all the available intro |  |
| **-a option:** | manual pages in succession. |  |
| **Syntax :** |  |
|  |  |
|  | man -a [COMMAND NAME] |  |
|  |  |  |
|  | This option searches the given command as a regular |  |
|  | expression in all the manuals and it returns the manual |  |
| **-k option:** | pages with the section number in which it is found. |  |
|  | **Syntax :** |  |
|  | man -k [COMMAND NAME] |  |
|  |  |  |
|  | This option returns the location in which the manual page |  |
| **-w option:** | of a given command is present. |  |
| **Syntax :** |  |
|  |  |
|  | man -w [COMMAND NAME] |  |
|  |  |  |
|  | It considers the command as case sensitive. |  |
| **-I option:** | **Syntax :** |  |
|  | man -I [COMMAND NAME] |  |
|  |  |  |

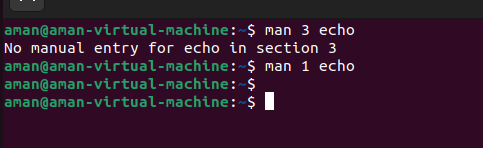
**IMPLEMENTATION:**

****

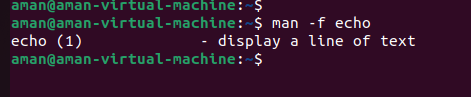


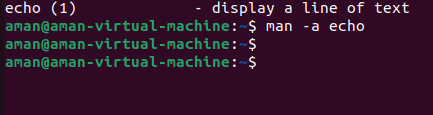


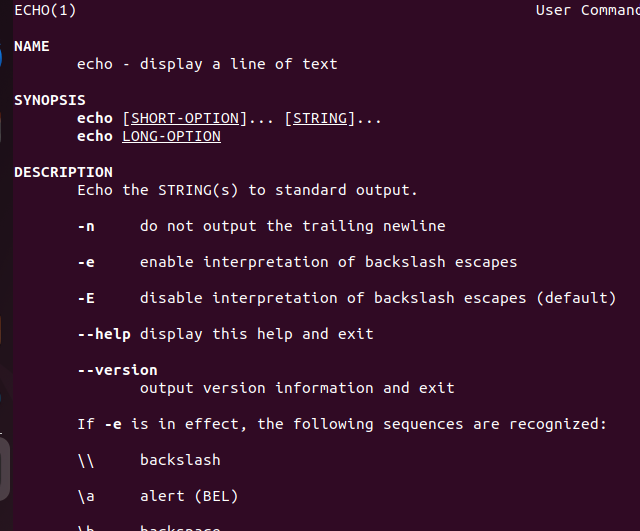


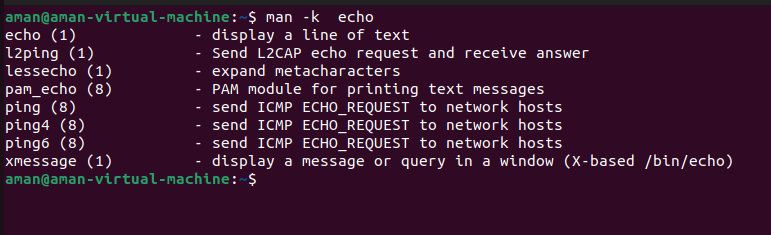


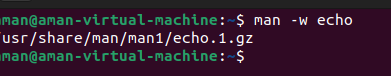




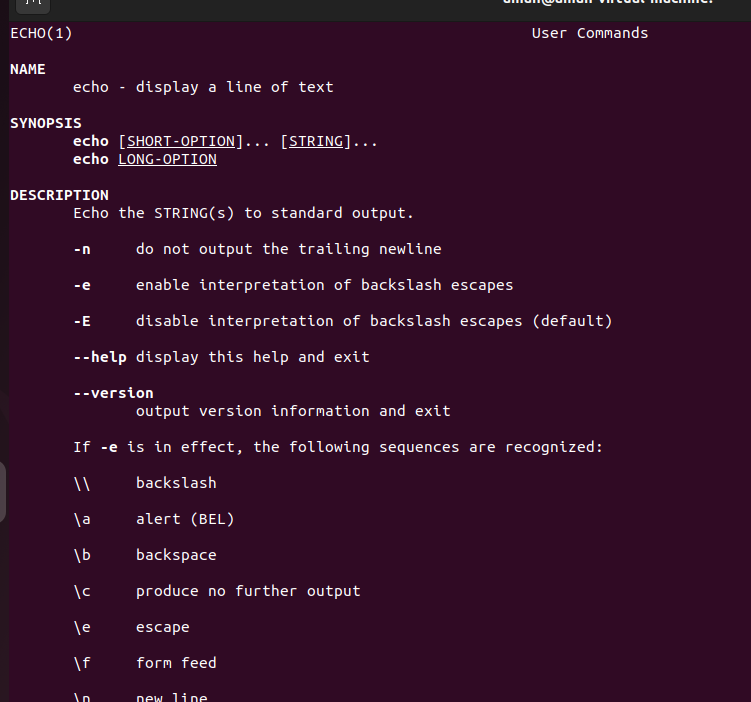












**2.echo**

**Description:**

echo is a fundamental command found in most operating systems that offer a command line. It is frequently used in scripts, batch files, and as part of individual commands; anywhere you may need to insert text.

**Syntax :** echo [SHORT-OPTION]... [STRING]...

**Tag** :

|  |  |  |
| --- | --- | --- |
| **-e** | Enable interpretation of backslash escape sequences (see below |  |
| for a list of these). |  |
|  |  |
|  |  |  |
| **\a** | An alert (The BELL character). |  |
|  |  |  |
| **\b** | Backspace |  |
|  |  |  |
| **\c** | Produce no further output after this. |  |
|  |  |  |
| **\e** | The escape character; equivalent to pressing the escape key. |  |
|  |  |  |
| **\f** | A form feed. |  |
|  |  |  |
| **\n** | A newline. |  |
|  |  |  |
| **\t** | A horizontal tab. |  |
|  |  |  |
| **\v** | A vertical tab. |  |
|  |  |  |

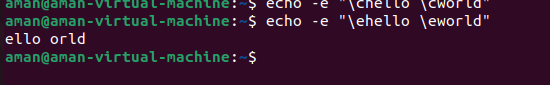
IMPLEMENTATION:

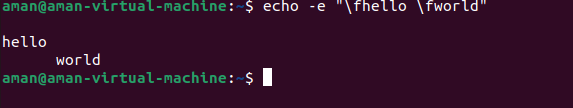


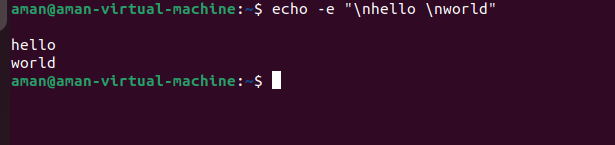


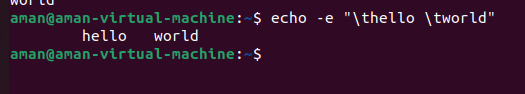


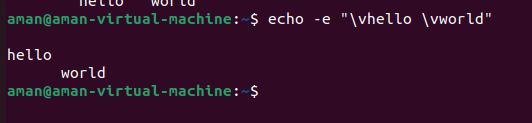












**3.Cat**

**Description:**

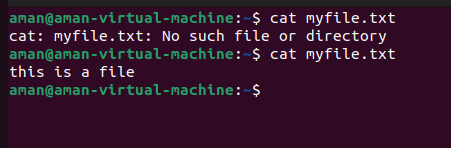
Cat(concatenate) command is very frequently used in Linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate files. So let us see some frequently used cat commands.

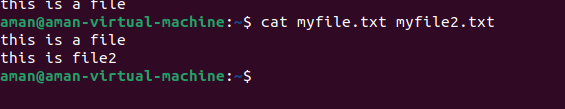
**Syntax :**

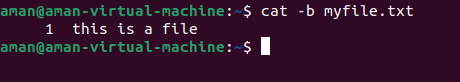
**v\**cat [Options] [File]...

|  |  |  |
| --- | --- | --- |
| **Tag** | **Description** |  |
| -b | Number non-blank output lines |  |
| -E | Display $ at end of each line |  |
| -n | Number all output lines |  |
| -s | Never more than one single blank line |  |
| -t | Equivalent to -vT |  |
| -T | Display TAB characters as ^I |  |
| -u | (Ignored) |  |
| -v | Use ^ and M- notation, except for LFD and TAB. display this |  |
| help and exit |  |
|  |  |
| --help | Display this help and exit |  |
| --version | Output version information and exit |  |

IMPLEMENTATION:

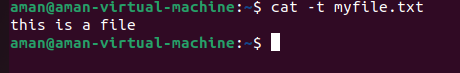


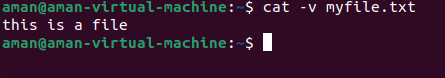


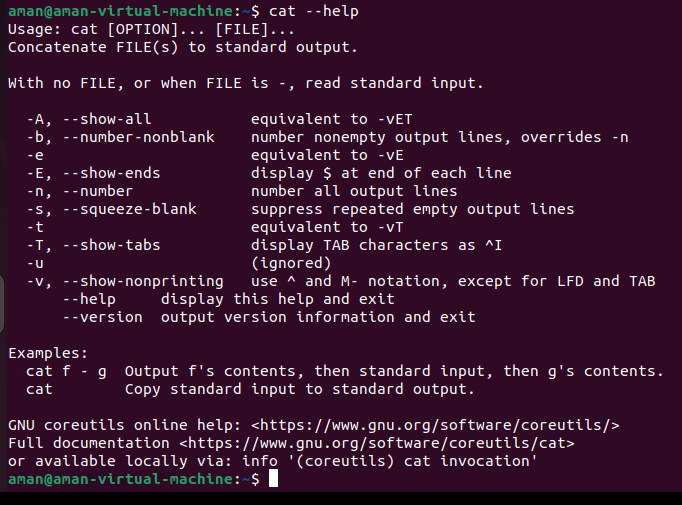




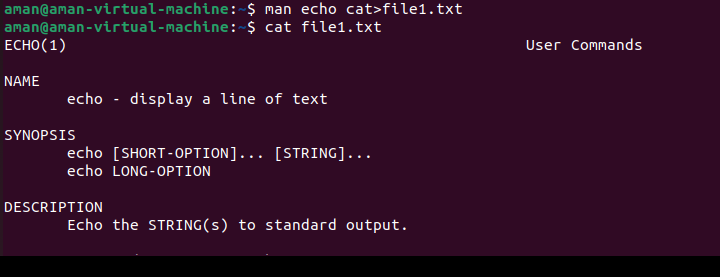


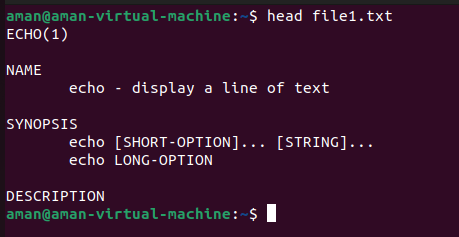
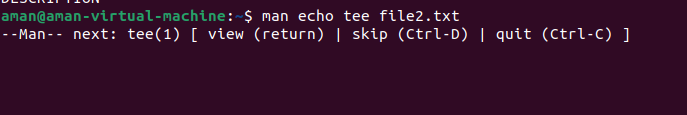










**4.touch**

**Description:**

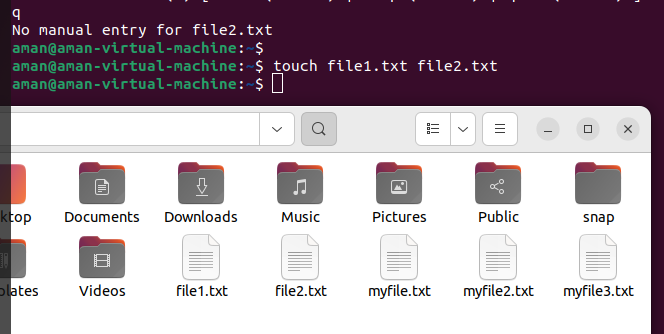
The touch command is a standard command used in UNIX/Linux operating system which is used to create, change and modify timestamps of a f

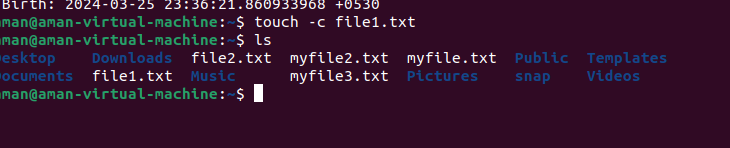
Initially, we are in home directory and this can be checked using the [pwd](https://www.geeksforgeeks.org/pwd-command-in-linux-with-examples/) command. Checking the existing files using command ls and then long listing command(ll) is used to gather more details about existing files.

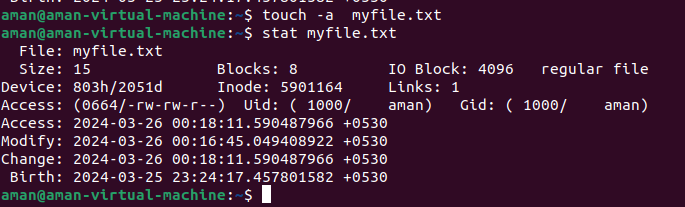
**Syntax :** touch file\_name

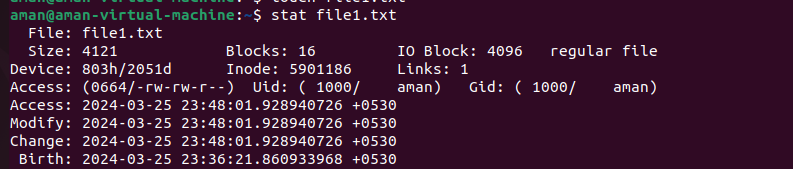
|  |  |
| --- | --- |
| **Tags** | **Description** |
|  |  |
| -a | Change only the access time |
|  |  |
| -c | Do not create any files |
|  |  |
| -d | Parse STRING and use it instead of current time |
|  |  |
| -f | Ignored |
|  |  |
| -m | Change only the modification time |
|  |  |
| -r | Use this file’s times instead of current time |
|  |  |
| -t | Use [[CC]YY]MMDDhhmm[.ss] instead of current time |
|  |  |

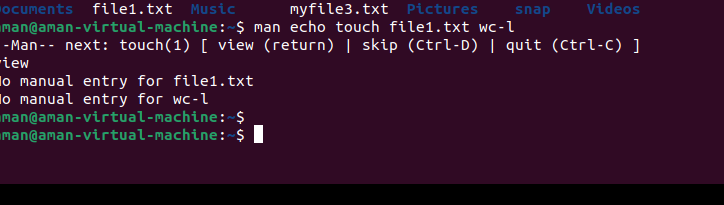
**Implementation:**

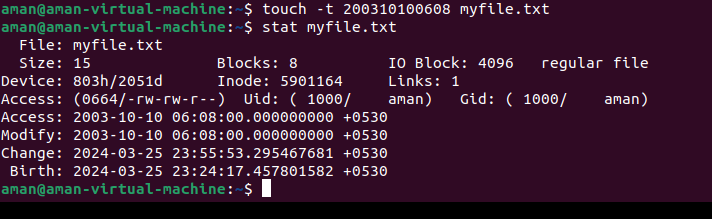


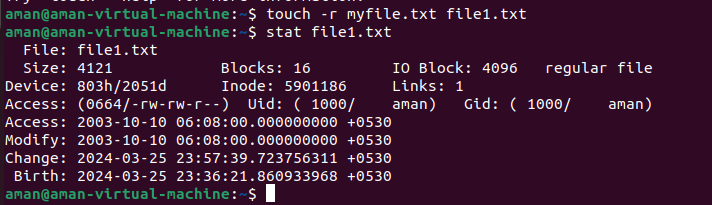


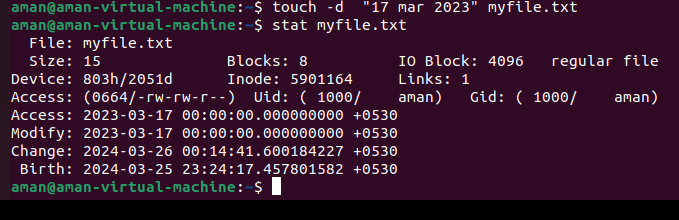


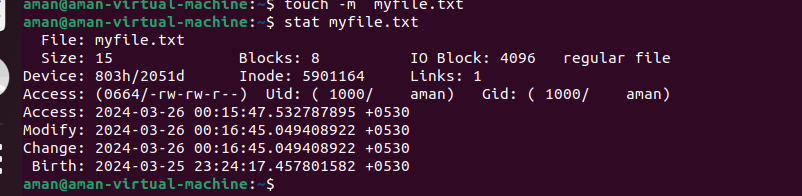










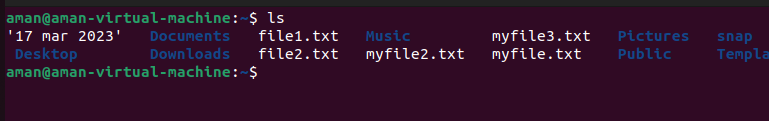


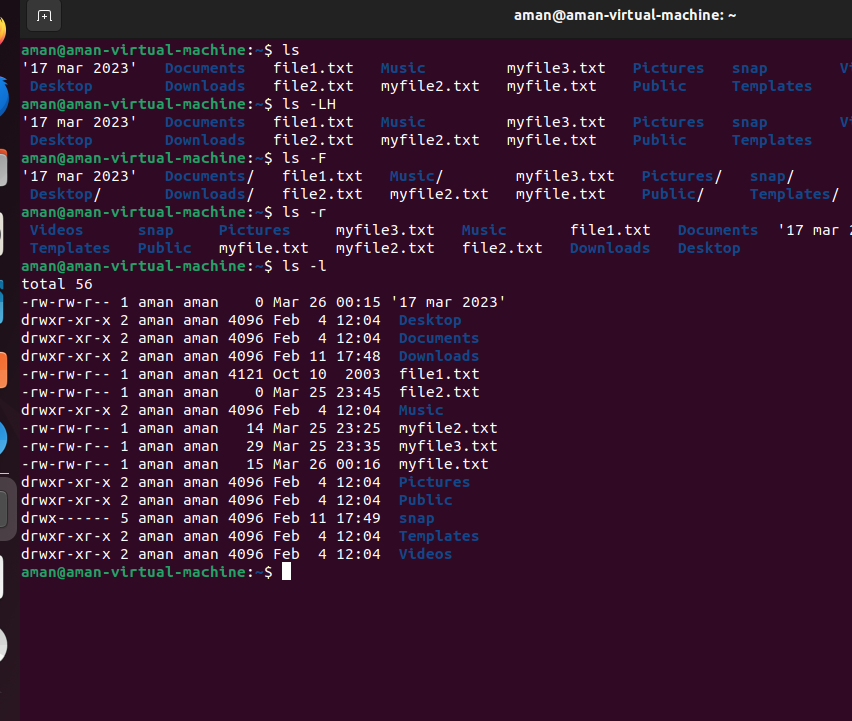
**5 ls** **Description**

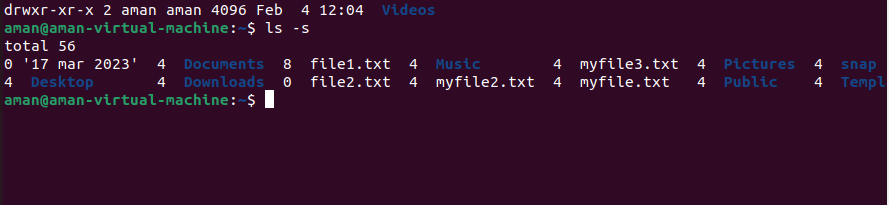
ls is a Linux shell command that lists directory contents of files and directories. Syntax: ls [OPTIONS] [FILES]

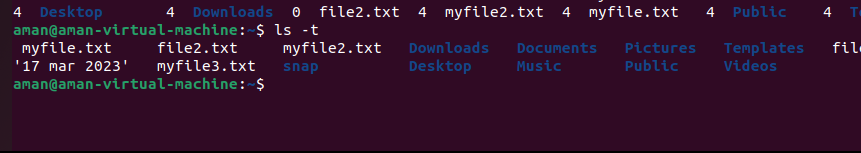
|  |  |
| --- | --- |
| **Option** | **Description** |
| [ls -a](https://www.rapidtables.com/code/linux/ls/ls-a.html) | List all files including hidden file starting with '.' |
| ls --color | Colored list [=always/never/auto] |
| ls -d | List directories - with ' \*/' |
| ls -F | Add one char of \*/=>@| to enteries |
| ls -i | List file's inode index number |
| [ls -l](https://www.rapidtables.com/code/linux/ls/ls-l.html) | List with long format - show permissions |
| [ls -la](https://www.rapidtables.com/code/linux/ls/ls-l.html) | List long format including hidden files |
| [ls -lh](https://www.rapidtables.com/code/linux/ls/ls-l.html) | List long format with readable file size |
| [ls -ls](https://www.rapidtables.com/code/linux/ls/ls-l.html) | List with long format with file size |
| [ls -r](https://www.rapidtables.com/code/linux/ls/ls-r.html#reverse) | List in reverse order |
| [ls -R](https://www.rapidtables.com/code/linux/ls/ls-r.html#recursive) | List recursively directory tree |
| [ls -s](https://www.rapidtables.com/code/linux/ls/ls-s.html#size) | List file size |
| [ls -S](https://www.rapidtables.com/code/linux/ls/ls-s.html#sort-size) | Sort by file size |
| [ls -t](https://www.rapidtables.com/code/linux/ls/ls-t.html) | Sort by time & date |
| ls -X | Sort by extension name |

**Implementation:**









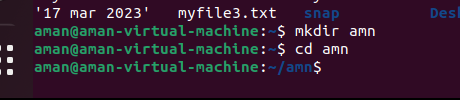
**6. mkdir**

**DESCRIPTION:**

Create the DIRECTORY(ies), if they do not already exist. Mandatory arguments to long options are mandatory for short options too.

**Syntax:** mkdir [OPTION]... DIRECTORY...

|  |  |
| --- | --- |
|  |  |
| TAG | DESCRIPTION |
|  |  |
| -m, | set file mode (as in chmod), not a=rwx - umask |
|  |  |
| -p, --parents | no error if existing, make parent directories as needed |
|  |  |
| -v, --verbose | print a message for each created directory |
|  |  |
| -Z | set the SELinux security context of each created |
|  |  |





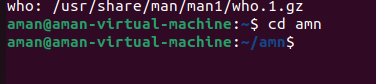
## **7.cd:**

cd command in linux known as change directory command. It is used to change current working directory.

## SYNOPSIS

cd [directory]

|  |  |
| --- | --- |
| -L | If you tell cd to move into a "directory", which is actually a symbolic link to a directory, it moves into the directory the symbolic link points to. |
| -P | Use the physical directory structure without following symbolic links. In other words, only change into the specified directory if it actually exists as named; symbolic links will not be followed. This option is the opposite of the -L option, and if they are both specified, this option will be ignored. |
| -e | If the -P option is specified, and the current working directory cannot be determined, this option tells cd to exit with an error. If -P is not specified along with this option, this option has no function. |
| -@ | present a file with extended attributes as a directory containing the file attributes |



**8.cp**

**Description:**

cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. *cp* command require at least two filenames in its arguments.

**Syntax: cp [options]... Source... Directory**

**Tag** **Description**

-b Make backup before removal

-d Preserve links

-f Remove existing destinations, never prompt

-i Prompt before overwrite

-l Link files instead of copying

-p Preserve file attributes if possible

-P Append source path to DIRECTORY

-r Copy recursively, non-directories as files

-R Copy directories recursively

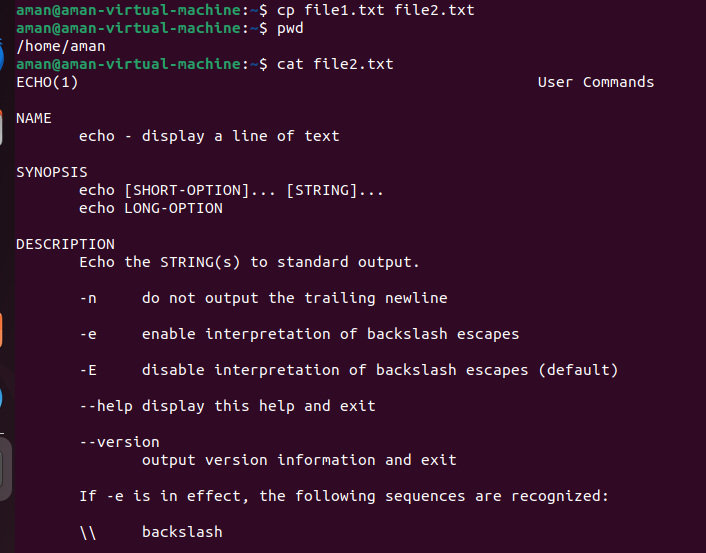
-s Make symbolic links instead of copying

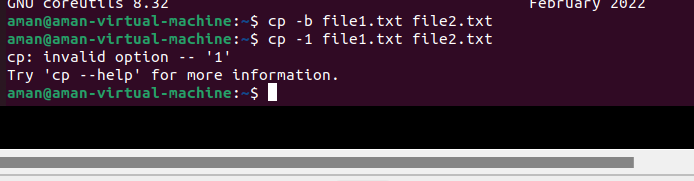
-S Override the usual backup suffix

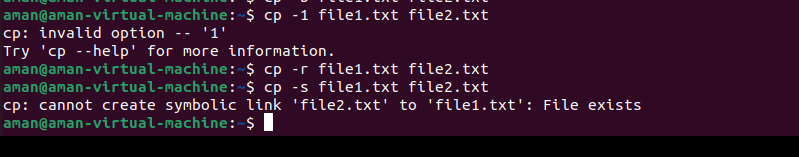
-u Copy only when the SOURCE file is newer than

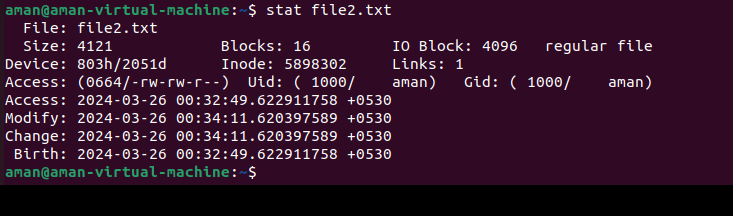
|  |  |
| --- | --- |
|  | destination file or when the destination file is missing |
| -v, | Explain what is being done |
| --help | Display this help and exit |
| --version | Output version information and exit. |

**Implementation:**









**9.pwd**

**Description:**

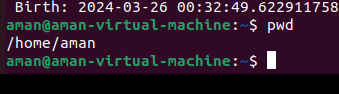
pwd stands for Print Working Directory. It prints the path of the working directory, starting from the root. $PWD is an [environment variable](https://www.geeksforgeeks.org/environment-variables-in-linux-unix/) which stores the path of the current directory.

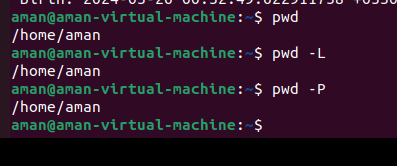
This command has two flags:

**pwd -L**: Prints the symbolic path.

**pwd -P:** Prints the actual path.

**Implementation:**

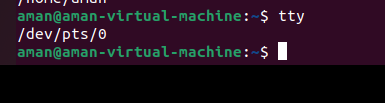




**10.tty**

Linux operating system represents everything in a file system, the hardware devices that we attach are also represented as a file. The terminal is also represented as a file. There a command exists called **tty** which displays information related to **terminal**. The **tty** command of terminal basically prints

the file name of the terminal connected to standard input. **tty** is short of teletype, but popularly known as a terminal it allows you to interact with the system by passing on the data (you input) to the system.



**11. who**

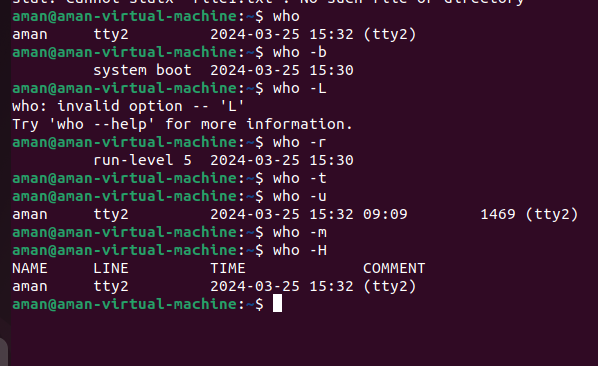
**Description:**

The who command is used to get information about currently logged in user on to system.

Syntax : who [options] [filename]

|  |  |
| --- | --- |
| **Tag** | **Description** |
|  |  |
| **-b** | Time of last system boot |
|  |  |
| **-d** | Print dead processes |
|  |  |
| **-H**, | Print line of column headings |
|  |  |
| **-l** | Print system login processes |
|  |  |
| **-m** | Only hostname and user associated with stdin |
|  |  |
| **-p** | Print active processes spawned by init |
|  |  |
| **-q** | All login names and number of users logged on |
|  |  |
| **-r** | Print current runlevel |
|  |  |
| **-s** | Print only name, line, and time (default) |
|  |  |
| **-t**, | Print last system clock change |
|  |  |
| **-u** | List users logged in |
|  |  |
| **--help** | Display this help and exit |
|  |  |
| **--version** | Output version information and exit |
|  |  |

**Implementation:**



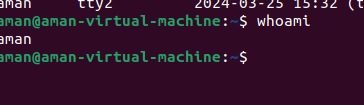
**12. whoami**

**Description:**

It is basically the concatenation of the strings “who”,”am”,”i” as whoami. It displays the username of the current user when this command is invoked.

|  |  |  |
| --- | --- | --- |
| **Syntax:** | whoami [OPTION] | |
|  |  |  |
| **Tag** |  | **Description** |
|  |  |  |
| **--help** |  | display this help and exit |
|  |  |  |
| **--version** |  | output version information and |
|  |  | exit |
|  |  |  |

**Implementation:**



## **13.wc:**

and characters count in the files specified in the file arguments.

By default it displays four-columnar output.

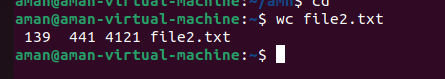
First column shows number of lines present in a file specified, second column shows number of words present in the file, third column shows number of characters present in file and fourth column itself is the file name which are given as argument.

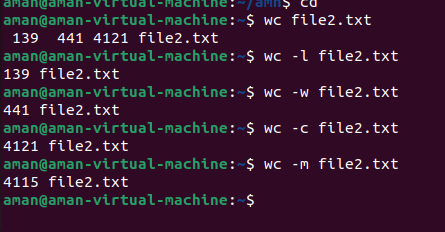
## **SYNOPSIS**

wc [OPTION]... [FILE]...

|  |  |
| --- | --- |
| -l | This option prints the number of lines present in a file. With this option wc command displays two-columnar output, 1st column shows number of lines present in a file and 2nd itself represent the file name. |
| -w | This option prints the number of words present in a file. With this option wc command displays two-columnar output, 1st column shows number of words present in a file and 2nd is the file name. |
| -c | This option displays count of bytes present in a file. With this option it display two-columnar output, 1st column shows number of bytes present in a file and 2nd is the file name. |
| -m | Using -m option ‘wc’ command displays count of characters from a file. |
| -L | It can be used to print out the length of longest (number of characters) line in a file. |

**Implementation:**





## **14.mv:**

mv stands for move. mv is used to move one or more files or directories from one place to another in file system like UNIX. It has two distinct functions:

1. It rename a file or folder.
2. It moves group of files to different directory.

This command normally works silently means no prompt for confirmation.

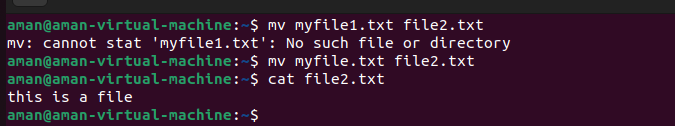
## SYNOPSIS

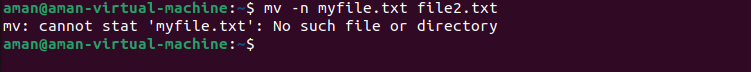
mv [OPTION] source destination

|  |  |
| --- | --- |
| -i | -i option makes the command ask the user for confirmation before moving a file that would overwrite an existing file, you have to press **y** for confirm moving, any other key leaves the file as it is. |
| -f | **-f** option overrides this minor protection and overwrite the destination file forcefully and delete the source file. |
| -n | **mv** prevent an existing file from being overwritten. |

|  |  |
| --- | --- |
| -b | With this option it is easier to take a backup of an existing file that will be overwritten as a result of the **mv** command. This will create a backup file with the tilde character(~) appended to it. |

**Implementation:**





**15.rm**

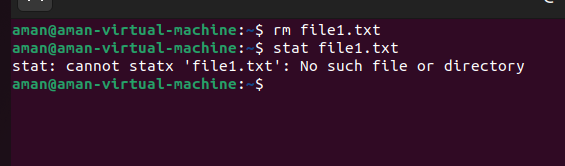
**Description:**

rm removes references to objects from the filesystem, where those objects

might have had multiple references (for example, a file with two different names). By default, it does not remove directories.

This command normally works silently and you should be very careful while running rm command because once you delete the files then you are not able to recover the contents of files and directories.

**Implementation:**



**16.rmdir:** It removes an empty directory on various operating systems.

## SYNOPSIS

rmdir name\_of\_directory

where name\_of\_directory corresponds with the name of the directory one wishes to delete.

|  |  |
| --- | --- |
| -p | removes parent directories if they are also empty. |

**Implementation:**



**PRACTICAL-3**

**OBJECTIVE:**  To implement the commands:

whatis , whereis,find,type,bc,expr, diff,cmp,comm,sort,tee,cut,tr, grep,head,tail,free,df,du,ulimit, cal,ncal,pipe(|)whatis:

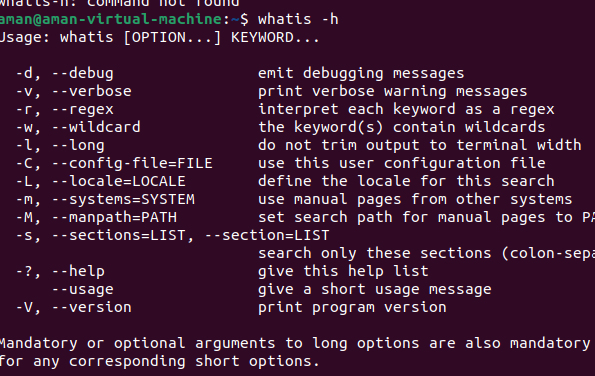
This command is helpful to get brief information about Linux commands or functions. Whatis command displays man page single line description for command that matches string passed as a command line argument to whatis command. Whatis command searches for string in its index databases which is maintained by mandb program. Whatis command picks short description of NAME section of man page of command that matches to input given to the whatis command.

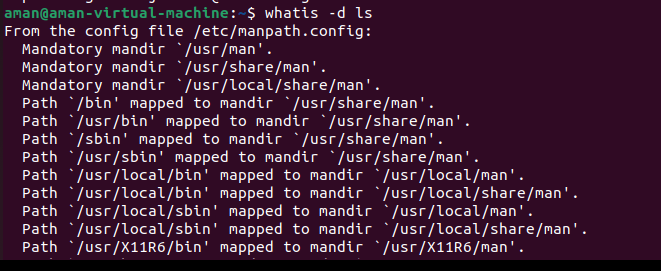
## SYNOPSIS

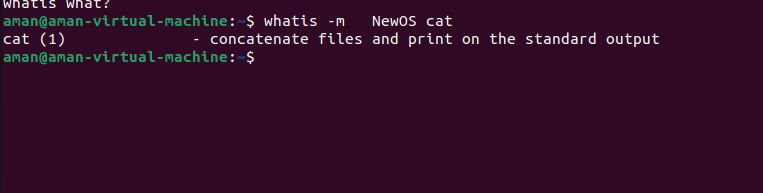
whatis [OPTIONS]

|  |  |
| --- | --- |
| -s | It is used to get Linux command information from specific section of man pages. |
| -w | It is used to search Linux commands or functions information using wild card. |
| -r | It is used to search Linux commands or functions information using regular expressions. |
| -l | It trims long output of Linux commands or functions information to avoid “Not good” output display on terminal that is going beyond screen |
| -M | It is used to restrict search up to specified path of man pages. |











## whereis:

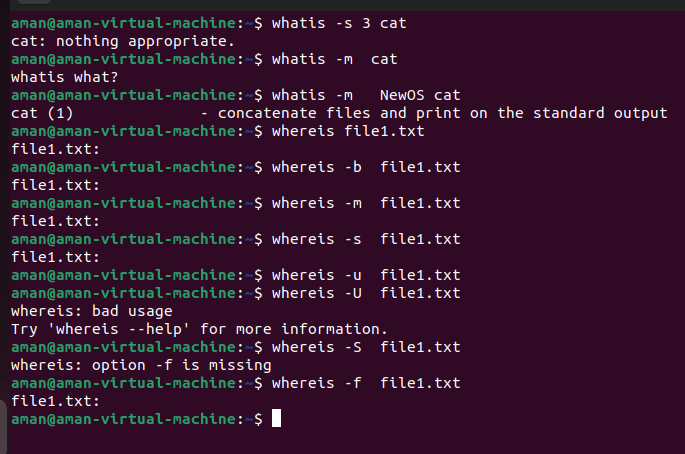
This command is used to find the location of source/binary file of a command and manuals sections for a specified file in Linux system. If we compare *whereis* command with *find* command they will appear similar to each other as both can be used for the same purposes but *whereis* command produces the result more accurately by consuming less time comparatively.

## SYNOPSIS

whereis [OPTIONS] filename...

|  |  |
| --- | --- |
| -b | This option is used when we only want to search for binaries. |
| -m | This option is used when we only want to search for manual sections. |
| -s | This option is used when we only want to search for sources. |
| -u | This option search for unusual entries. |
| -B | This option is used to change or otherwise limit the places where *whereis* searches for binaries. |

|  |  |
| --- | --- |
| -M | This option is used to change or otherwise limit the places where *whereis* searches for manual sections. |
| -S | This option is used to change or otherwise limit the places where *whereis* searches for sources. |
| -f | This option simply terminate the last directory list and signals the start of file names. |
| -V | Displays the version information and exit. |
| -h | Displays this help and exit. |



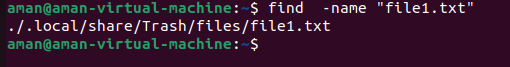
## find:

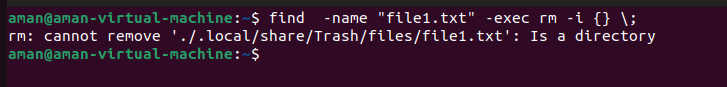
The find command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date,owner and permissions. By using the ‘-exec’ other UNIX commands can be executed on files or folders found.

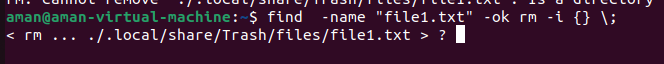
## SYNOPSIS

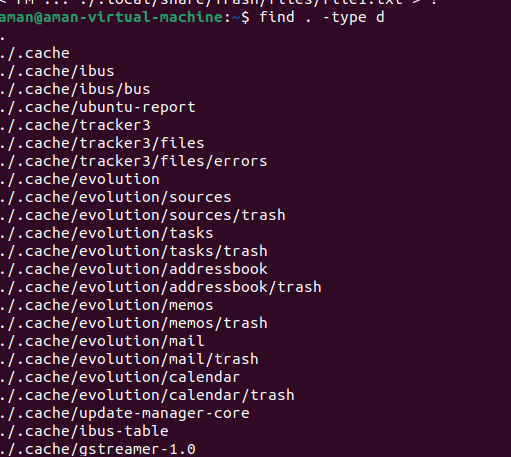
find [where to start searching from] [expression determines what to find] [-options] [what to find]

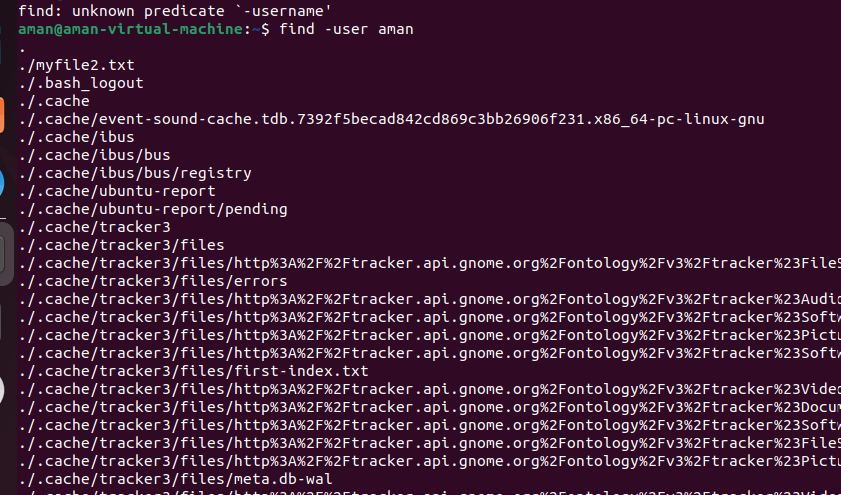
|  |  |
| --- | --- |
| -exec CMD | The file being searched which meets the above criteria and returns 0 for as its exit status for successful command execution. |
| -ok CMD | It works same as -exec except the user is prompted first. |
| -inum N | Searches for file number with inode number ‘N’. |
| -links N | Search for files with ‘N’ links. |
| -name demo | Search for files that are specified by ‘demo’. |
| -newer file | Search for files that were modified/created after ‘file’. |
| -perm octal | Search for the file if permission is ‘octal’. |
| -print | Display the path name of the files found by using the rest of the criteria. |
| -empty | Searches for empty files and directories. |
| -size +N/-N | Search for files of ‘N’ blocks; ‘N’ followed by ‘c’can be used to measure size in characters; ‘+N’ means size > ‘N’ blocks and ‘-N’ means size < 'N' blocks. |
| -user name | Search for files owned by user name or ID ‘name’. |
| \(expr \) | True if ‘expr’ is true; used for grouping criteria combined with OR or AND. |
| ! expr | True if ‘expr’ is false. |

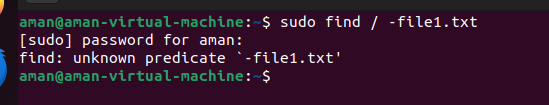


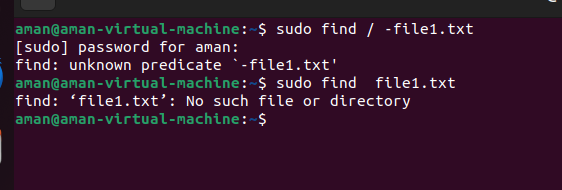












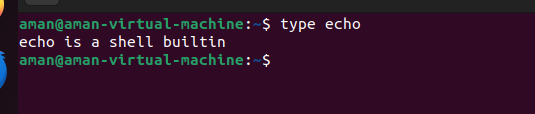
## type:

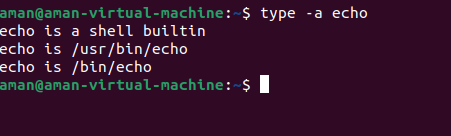
This command is used to describe how its argument would be translated if used as commands. It is also used to find out whether it is built-in or external binary file.

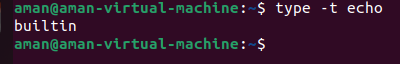
## SYNOPSIS

type [OPTIONS] command names

|  |  |
| --- | --- |
| -a | This option is used to find out whether it is an alias, keyword or a function and it also displays the path of an executable, if available. |
| -t | This option will display a single word as an output.  alias – if command is a shell alias  keyword – if command is a shell reserved word builtin – if command is a shell builtin  function – if command is a shell function file – if command is a disk file |
| -p | This option displays the name of the disk file which would be executed by the shell. It will return nothing if the command is not a disk file. |







## bc(binary calculator):

bc command is used for command line calculator. It is similar to basic calculator by using which we can do basic mathematical calculations.

## SYNOPSIS

bc [ -hlwsqv ] [long-options] [ file ... ]

|  |  |
| --- | --- |
| -h{--help} | Print the usage and exit |
| -i{--interactive} | Force interactive mode |
| -l{--mathlib} | Define the standard math library |
| -w{--warn} | Give warnings for extensions to POSIX bc |
| -s{--standard} | Process exactly the POSIX bc language |
| -q{--quiet} | Do not print the normal GNU bc welcome |
| -v{--version} | Print the version number and copyright and quit |

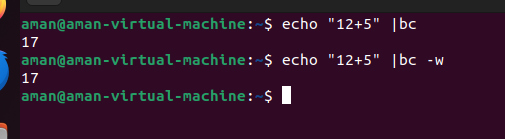
The bc command supports the following features: Arithmetic operators

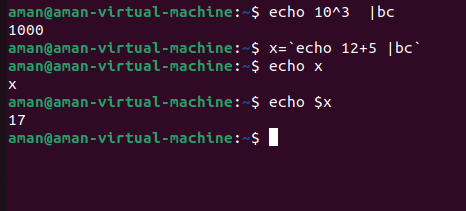
Increment or Decrement operators

Assignment operators

Comparison or Relational operators Logical or Boolean operators

Math functions Conditional statements Iterative statements





**6.expr:** evaluates expressions

## SYNOPSIS

expr EXPRESSION expr OPTION

EXPRESSIONS:

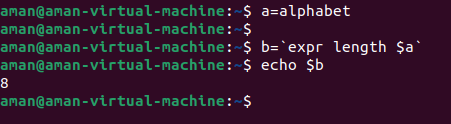
|  |  |
| --- | --- |
| ARG1 | ARG2 | ARG1 if it is neither null nor 0, otherwise ARG2 |
| ARG1 & ARG2 | ARG1 if neither argument is null or 0, otherwise 0 |
| ARG1 < ARG2 | ARG1 is less than ARG2 |
| ARG1 <= ARG2 | ARG1 is less than or equal to ARG2 |
| ARG1 = ARG2 | ARG1 is equal to ARG2 |
| ARG1 != ARG2 | ARG1 is unequal to ARG2 |
| ARG1 >= ARG2 | ARG1 is greater than or equal to ARG2 |
| ARG1 > ARG2 | ARG1 is greater than ARG2 |
| ARG1 + ARG2 | arithmetic sum of ARG1 and ARG2 |
| ARG1 - ARG2 | arithmetic difference of ARG1 and ARG2 |
| ARG1 \* ARG2 | arithmetic product of ARG1 and ARG2 |
| ARG1 / ARG2 | arithmetic quotient of ARG1 and ARG2 |
| ARG1 % ARG2 | arithmetic remainder of ARG1 and ARG2 |
| STRING : REGEXP | anchored pattern match of REGEXP in STRING |
| match STRING REGEXP | same as STRING : REGEXP |
| substr STRING POS LENGTH | substring of STRING, POS counted from 1 |
| index STRING CHARS | index in STRING where any CHARS is found, or 0 |
| length STRING | length of STRING |
| + TOKEN | iterpret TOKEN as a string, even if it is a keyword like 'match' or an operator like '/' |

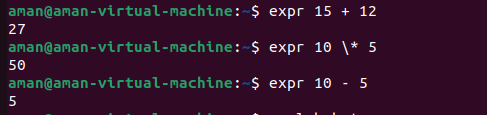
|  |  |
| --- | --- |
| (EXPRESSION) | value of EXPRESSSION |

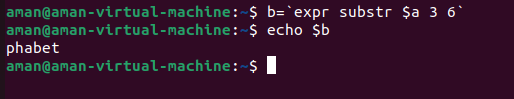
OPTION

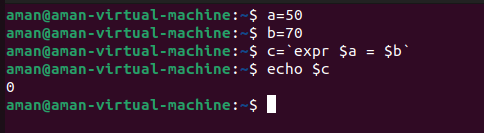
|  |  |
| --- | --- |
| --help | display help and exit |
| --version | output varsion info and exit |

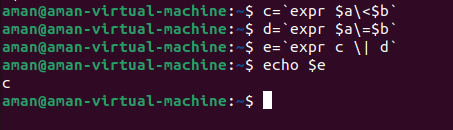
## 











## diff**:**

diff stands for difference. This command is used to display the differences in the files by comparing the files line by line. Unlike its fellow members, cmp and comm, it tells us which lines in one file have is to be changed to make the two files identical. It tells you the instructions on how to change the first file to make it match the second file.

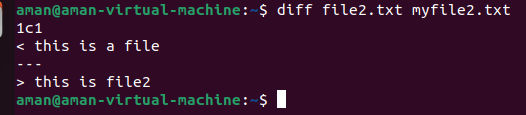
Some special symbols are used by diff command. These are:- a : add

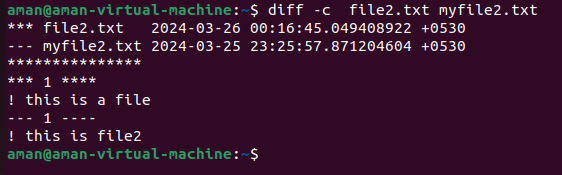
c : change d : delete

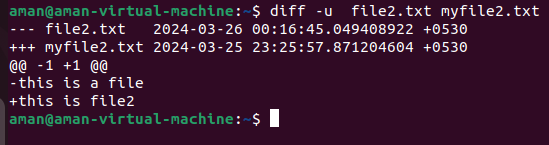
## SYNOPSIS

diff [OPTION] File1 File2

|  |  |
| --- | --- |
| -c | To view differences in context mode, use the -c option. |
| -u | To view differences in unified mode, use the -u option. It is similar to context mode but it doesn’t display any redundant information or it shows the information in concise form. |
| -i | By default this command is case sensitive. To make this command case in-sensitive use -i option with diff. |







## cmp:

cmp command in Linux/UNIX is used to compare the two files byte by byte and helps you to find out whether the two files are identical or not.

When cmp is used for comparison between two files, it reports the location of the first mismatch to the screen if difference is found and if no difference is found *i.e* the files compared are identical.

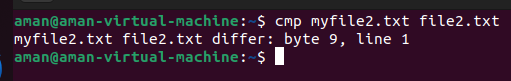
cmp displays no message and simply returns the prompt if the the files compared are identical.

## SYNOPSIS

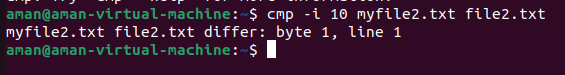
cmp [OPTION]... FILE1 [FILE2 [SKIP1 [SKIP2]]]

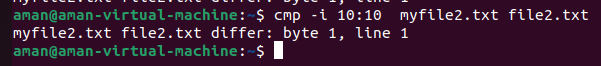
|  |  |
| --- | --- |
| -b | If you want cmp displays the differing bytes in the output when used with -b option. |
| -i  [bytes-to-be-skipped  ] | It helps to skip a particular number of initial bytes from both the files and then after skipping it compares the files. This can be done by specifying the number of bytes as argument to the -i command line option. |
| -i [bytes to be skipped from first file] : [bytes to be skipped from second file] | It allows us to input the number of bytes we want to skip from both the files separately. |

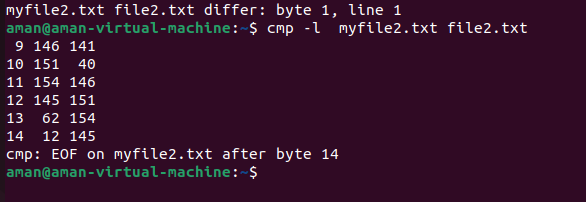
|  |  |
| --- | --- |
| -l | This option makes the cmp command print byte position and byte value for all differing bytes. |
| -s | This allows you to suppress the output normally produced by cmp command  i.e it compares two files without writing any messages. |
| -n | This option allows you to limit the number of bytes you want to compare. |













## comm:

comm compare two sorted files line by line and write to standard output; the lines that are common and the lines that are unique. Points to be kept in mind are:

As using comm, we are trying to compare two files therefore the syntax of comm command needs two filenames as arguments.

With no OPTION used, comm produces three-column output where first column contains lines unique to FILE1 ,second column contains lines unique to FILE2 and third and last column contains lines common to both the files.

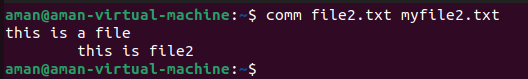
comm command only works right if you are comparing two files which are already sorted.

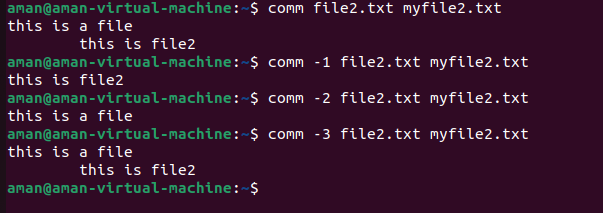
## SYNOPSIS

comm [OPTION]... FILE1 FILE2

|  |  |
| --- | --- |
| -1 | suppress column 1 (lines unique to FILE1) |
| -2 | suppress column 2 (lines unique to FILE2) |
| -3 | suppress column 3 (lines that appear in both files) |

|  |  |
| --- | --- |
| --check-order | check that the input is correctly sorted, even if all input lines are pairable |
| --nocheck-order | do not check that the input is correctly sorted |
| --total | output a summary |
| --output-delimiter=ST R | separate columns with STR |





## sort:

The command sorts the lines of text files. It writes sorted concatenation of all FILE(s) to standard output.

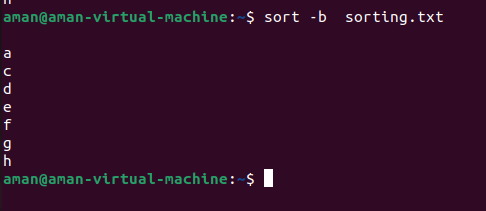
## SYNOPSIS

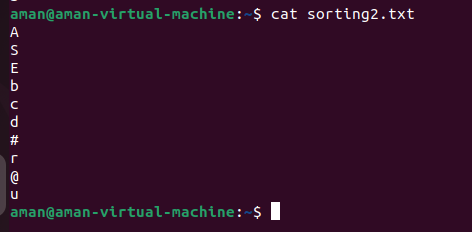
sort [OPTION]... [FILE]...

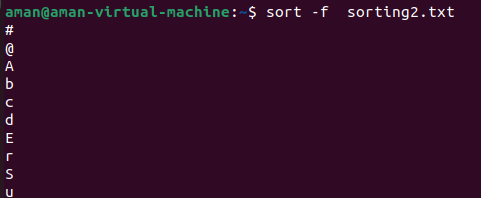
|  |  |
| --- | --- |
| -b, --ignore-leading-blanks | ignore leading blanks |
| -d, --dictionary-order | consider only blanks and alphanumeric characters |
| -f, --ignore-case | fold lower case to upper case characters |
| -g, --general-numeric-sort | compare according to general numerical value |
| -i, --ignore-nonprinting | consider only printable characters |
| --help | display help and exit |
| --version | output version info and exit |











## tee:

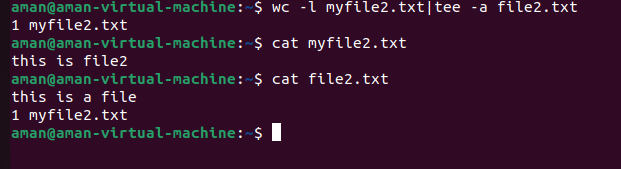
This command reads the standard input and writes it to both the standard output and one or more files. It basically breaks the output of a program so that it can be both displayed and saved in a file. It does both the tasks simultaneously, copies the result into the specified files or variables and also display the result.

## SYNOPSIS

tee [OPTION]... [FILE]...

|  |  |
| --- | --- |
| -a | It basically do not overwrite the file but append to the given file. |
| --help | It gives the help message and exit. |
| --version | It gives the version information and exit. |





1. **cut:** remove sections from each line of files

## SYNOPSIS

cut OPTION... [FILE]...

|  |  |
| --- | --- |
| -c, --characters=LIST | select only these characters |
| -b, --bytes=LIST | select only these bytes |
| -d, --delimiter=DELIM | use DELIM instead of TAB for field delimiter |
| --complement | complement the set of selected bytes, characters or fields |
| --help | display this help and exit |
| --version | output version information and exit |

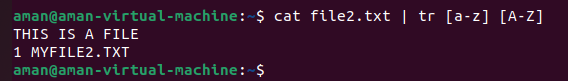


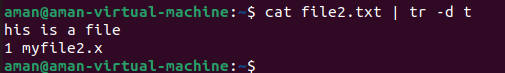
1. **tr:** translate or delete characters

## SYNOPSIS

tr [OPTION]... SET1 [SET2]

|  |  |
| --- | --- |
| -c, -C, --complement | use the complement of SET1 |
| d, --delete | delete characters in SET1, do not translate |
| -s, --squeeze-repeats | replace each sequence of a repeated character that is listed in the last specified SET, with a single occurrence of that character |
| -t, --truncate-set | first truncate SET1 to length of SET2 |
| --help | display this help and exit |
|  |  |





## grep:

The command print lines that match patterns.

Grep searches for PATTERNS in each FILE. PATTERNS is one or patterns separated by newline characters, and grep prints each line that matches

a pattern. A FILE of “-” stands for standard input. If no FILE is given, recursive searches examine the working directory, and nonrecursive searches read standard input.

## SYNOPSIS

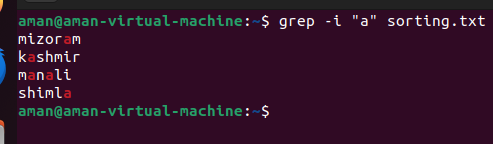
grep [OPTION...] PATTERNS [FILE...]

grep [OPTIONS] -e PATTERN ... [FILE...]

grep [OPTIONS] -f FILE ... [FILE...]

|  |  |
| --- | --- |
| --help | Output a usage message and exit. |

|  |  |
| --- | --- |
| -V, --version | Output the version number of grep and exit. |
| -E, --extended-regexp | Interpret PATTERN as an extended regular expression (ERE, see below) |
| -F, --fixed-strings | Interpret PATTERN as a list of fixed strings (instead of regular expressions), separated by newlines, any of which is to be matched. |
| -G, --basic-regexp | Interpret PATTERN as a basic regular expression (BRE, see below). This is the default. |
| -P, --perl-regexp | Interpret the pattern as a Perl-compatible regular expression(PCRE). This is experimental and grep -P may warn of unimplemented features. |

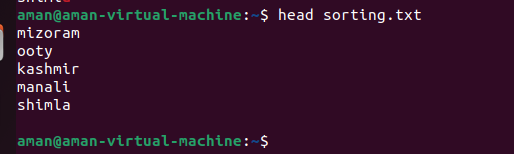


1. **head:** output the first part of files

## SYNOPSIS

head [OPTION]... [FILE]....

|  |  |
| --- | --- |
| -c, --bytes=[-]NUM | print the first NUM bytes of each file; with the leading '-', print all but the last NUM bytes of each file |
| -n, --lines=[-]NUM | print the first NUM lines instead of the first 10; with the leading '-', print all but the last NUM lines of each file |
| -q, --quiet, --silent | never print headers giving file names |
| -v, --verbose | always print headers giving file names |
| -z, --zero-terminated | line delimiter is NUL, not newline |

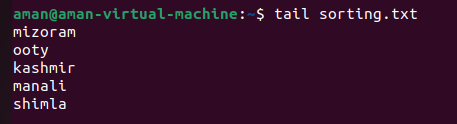


1. **tail:** output the last part of files

## SYNOPSIS

tail [OPTION]... [FILE]....

|  |  |
| --- | --- |
| -c, --bytes=[+]NUM | output the last NUM bytes; or use -c +NUM to output starting with byte NUM of each file |
| -f, --follow[= {name|descriptor}] | output appended data as the file grows;  an absent option argument means 'descriptor-' |
| -F | same as **--follow**=name --retry |
| -n, --lines=[+]NUM | output the last NUM lines, instead of the last 10; or use **-n** +NUM to output starting with line NUM |
| --max-unchanged-stats=N | with --follow=name, reopen a FILE which has not |



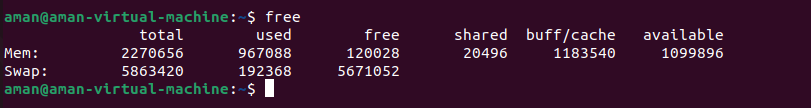
|  |  |
| --- | --- |
| --pid=PID | with -f, terminate after process ID, PID dies |
| -q, --quiet, --silent | never output headers giving file names |
| --retry | keep trying to open a file if it is inaccessible |
| -s, --sleep-interval=N | with -f, sleep for approximately N seconds (default 1.0) between iterations; with inotify and --pid=P, check process P at least once every N seconds |
| -v, --verbose | always output headers giving file names |
| -z, --zero-terminated | line delimiter is NUL, not newline |

1. **free:** Display amount of free and used memory in the system

## SYNOPSIS

free [options]

|  |  |
| --- | --- |
| -b, --bytes | Display the amount of memory in bytes. |
| -k, --kibi | Display the amount of memory in kibibytes. This is the default |
| -m, --mebi | Display the amount of memory in mebibytes. |
| -g, --gibi | Display the amount of memory in gibibytes. |

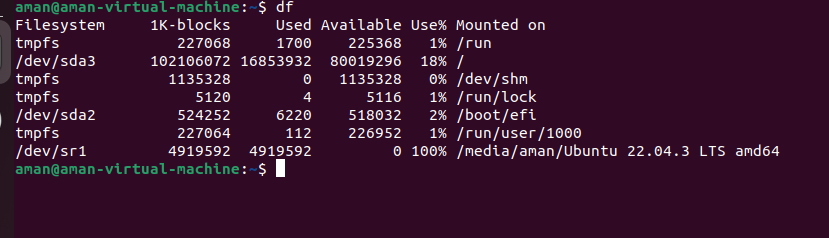


1. **df:** report file system disk space usage

## SYNOPSIS

df [OPTION]... [FILE]...

|  |  |
| --- | --- |
| -h, --human-readable | print sizes in powers of 1024 (e.g., 1023M) |
| -H, --si | print sizes in powers of 1000 (e.g., 1.1G) |
| -i, --inodes | list inode information instead of block usage |
| --help | display this help and exit |
| --version | output version information and exit |

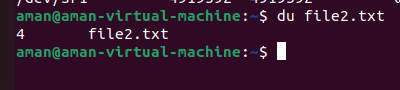


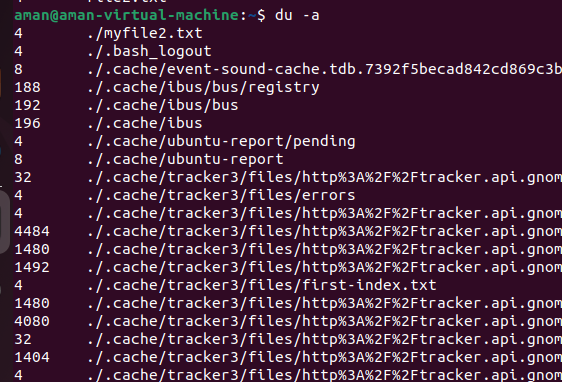
1. **du:** estimate file space usage

## SYNOPSIS

du [OPTION]... [FILE]...

|  |  |
| --- | --- |
| -a, --all | write counts for all files, not just directories |
| -0, --null | end each output line with NUL, not newline |
| -B, --block-size=SIZE | Scale sizes by SIZE before printing them; e.g., '-BM' print sizes in units of 1,048,576 bytes; |
| -b, --bytes | equivalent to '--apparent-size –block-size=1' |
| -c, --total | produce a grand total |





1. **ulimit**: get and set user limits

## SYNOPSIS

#include<ulimit.h>

long ulimit(int cmd, long newlimit)

The ulimit() call will get or set some limit for the calling process. The cmd argument can have one of the following values.

**UL\_GETFSIZE**: Return the limit on the size of a file, in units of 512 bytes.

**UL\_SETFSIZE:** Set the limit on the size of a file.

* 1. (Not implemented for Linux) Return the maximum possible adress of the data segment.
  2. (Implemented but no symbolic constant provided) Return the maximum number of files that the calling process can open.



1. **cal, ncal:** displays a calendar and the date of Easter.

## SYNOPSIS

cal [-31jy] [-A number] [-B number] [-d yyyy-mm] [[month] year]

cal [-31j] [-A number] [-B number] [-d yyyy-mm] -m month [year]

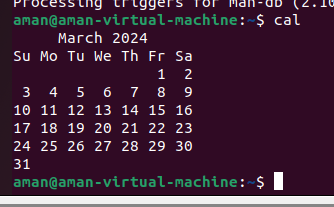
ncal [-C] [-31jy] [-A number] [-B number] [-d yyyy-mm] [[month] year]

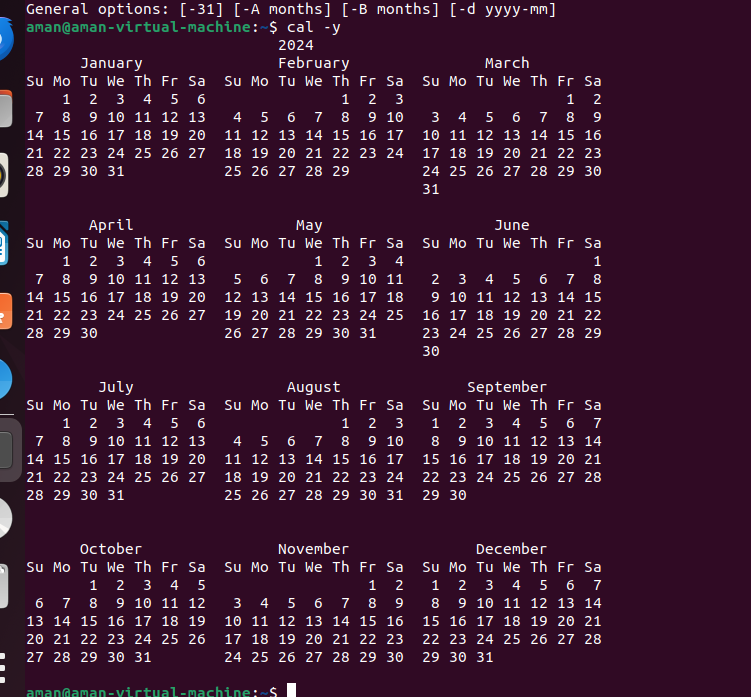
ncal [-C] [-31j] [-A number] [-B number] [-d yyyy-mm] -m month [year] ncal [-31bhjJpwySM] [-A number] [-B number] [-H yyyy-mm-dd] [-d yyyy-mm] [-s country\_code][[month] year]

ncal [-31bhJeoSM] [-A number] [-B number] [-d yyyy-mm] [year]

|  |  |
| --- | --- |
| -h | Turns off highlighting of today |
| -J | Display Julian Calendar, if combined with the -o option, display date of Orthodox Easter according to the Julian Calendar. |

|  |  |
| --- | --- |
| -e | Display date of Easter (for western churches). |
| -j | Display Julian days (days one-based, numbered from January 1). |
| -m month | Display the specified month. If month is specified as a decimal number, appending ‘f’ or ‘p’ displays the same month of the following or previous year respectively. |
| -o | Display date of Orthodox Easter (Greek and Russian Orthodox Churches). |
| -p | Print the country codes and switching days from Julian to Gregorian Calendar as they are assumed by ncal. |
| -s country\_code | If not specified, ncal tries to guess the switch date from the local environment or falls back to September 2, 1752. This was when Great Britain and her colonies switched to the Gregorian Calendar. |
| -w | Print the number of the week below each week column. |
| -y | Display a calendar for the specified year. This option is implied when a year but no month are specified on the command line. |
| -3 | Display the previous, current and next month surrounding today. |
| -1 | Display only the current month. This is the default. |
| -A number | Months to add after. The specified number of months is added to the end of the display. This is in addition to any date range selected by the -y, -3, or -1 options |
| -B number | Months to add before. The specified number of months is added to the beginning of the display. |
| -C | Completely switch to cal mode. For cal like output only, use -b instead. |
| -N | Switch to **ncal** mode |
| -d yyyy-mm | Use yyyy-mm as the current date (for debugging of date selection) |
| -H yyyy-mm-dd | Use yyyy-mm-dd as the current date (for debugging of highlighting). |
| -M | Weeks start on Monday. |
| -S | Weeks start on Sunday. |
| -b | Use oldstyle format for ncal output. |





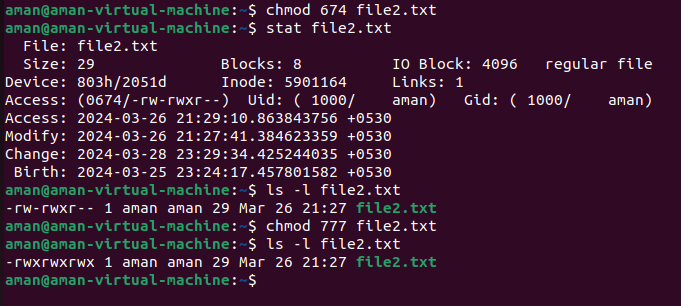
LAB 4 COMMANDS

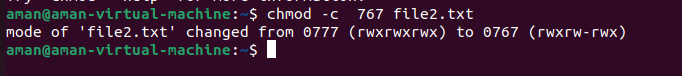
**1. chmod:** chmod command is used to change the access mode of a file.

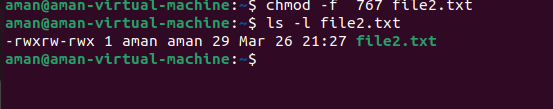
**SYNOSIS:**

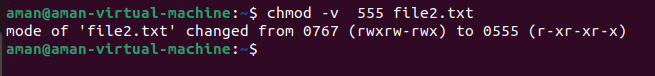
chmod [reference][operator][mode] file...

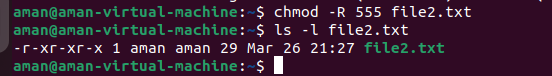
|  |  |
| --- | --- |
| - **c** | like verbose but report only when a change is made |
| **-f** | suppress most error messages |
| **-v** | output a diagnostic for every file processed |
| **-R** | change files and directories recursively |











**2. umask:**

The umask command is used to set this mask, or to show you its current value. The default creation permissions for files are 666 and for directories 777.

For example, to calculate how uname 022 will affect newly created files and directories, use:

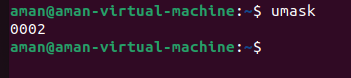
Files: 666 - 022 = 644. The owner can read and modify the files. Group and others can only read the files.

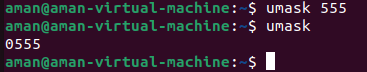
Directories: 777 - 022 = 755.The owner can cd into the directory and list read, modify, create or delete the files in the directory. Group and others can cd into the directory and list and read the files.

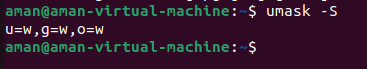
**SYNOPSIS**

umask [OPTION] [*mode*]

|  |  |
| --- | --- |
| -p | if MODE is omitted, output in a form that may be reused as input |
| -s | makes the output symbolic; otherwise an octal number is output |









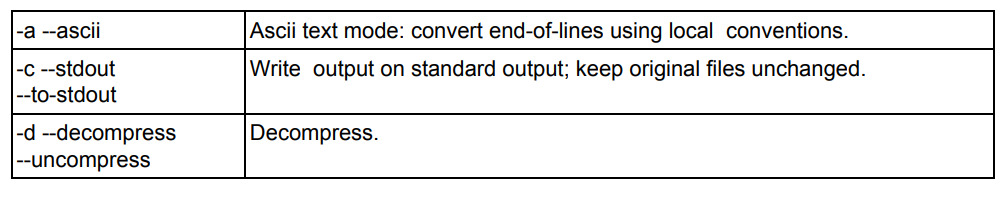
**3. gzip:** gzip, gunzip, zcat - compress or expand files

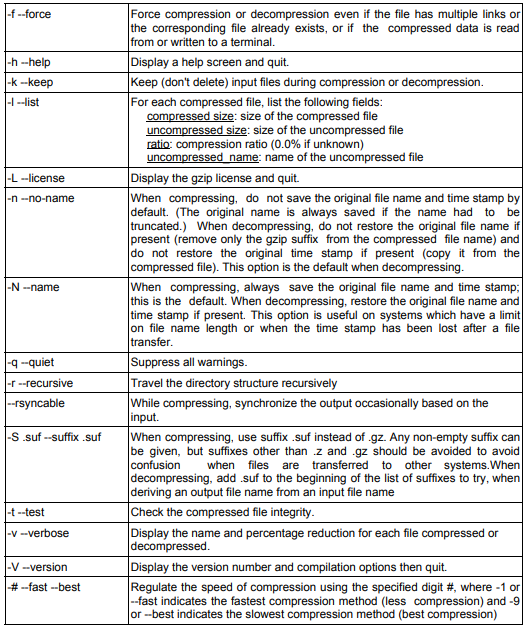
**SYNOPSIS**

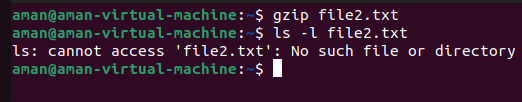
gzip[ -acdfhklLnNrtvV19 ] [--rsyncable] [-S suffix] [ name ...]

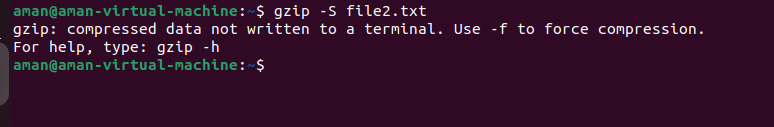
gunzip [ -acfhklLnNrtvV ] [-S suffix] [ name ... ]

zcat [ -fhLV ] [ name ...

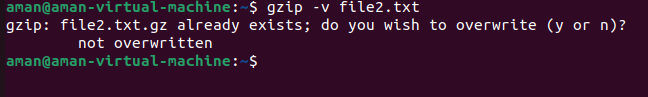


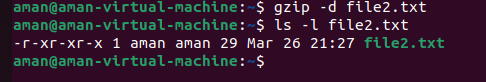


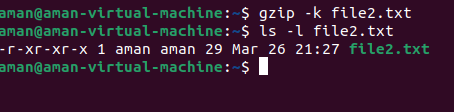


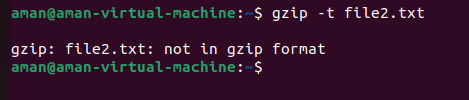












**4. tar**: an archiving utility

**SYNOPSIS**

Traditional usage

tar {A|c|d|r|t|u|x}[GnSkUWOmpsMBiajJzZhPlRvwo] [ARG...]

UNIX-style usage

tar -A [OPTIONS] ARCHIVE ARCHIVE tar -c [-f ARCHIVE] [OPTIONS] [FILE...] tar -sd [-f ARCHIVE] [OPTIONS] [FILE...]

tar -t [-f ARCHIVE] [OPTIONS] [MEMBER...]

tar -r [-f ARCHIVE] [OPTIONS] [FILE...]

tar -u [-f ARCHIVE] [OPTIONS] [FILE...]

tar -x [-f ARCHIVE] [OPTIONS] [MEMBER...] GNU-style usage

tar {--catenate|--concatenate} [OPTIONS] ARCHIVE ARCHIVE

tar --create [--file ARCHIVE] [OPTIONS] [FILE...]

tar {--diff|--compare} [--file ARCHIVE] [OPTIONS] [FILE...]

tar --delete [--file ARCHIVE] [OPTIONS] [MEMBER...]

tar --append [-f ARCHIVE] [OPTIONS] [FILE...]

tar --list [-f ARCHIVE] [OPTIONS] [MEMBER...]

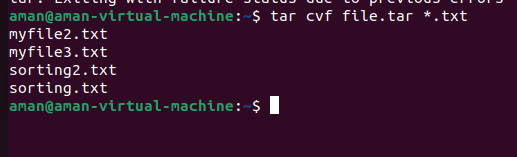
tar --test-label [--file ARCHIVE] [OPTIONS] [LABEL...]

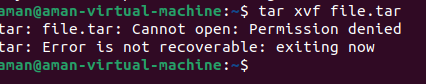
tar --update [--file ARCHIVE] [OPTIONS] [FILE...]

tar --update [-f ARCHIVE] [OPTIONS] [FILE...]

tar {--extract|--get} [-f ARCHIVE] [OPTIONS] [MEMBER...]



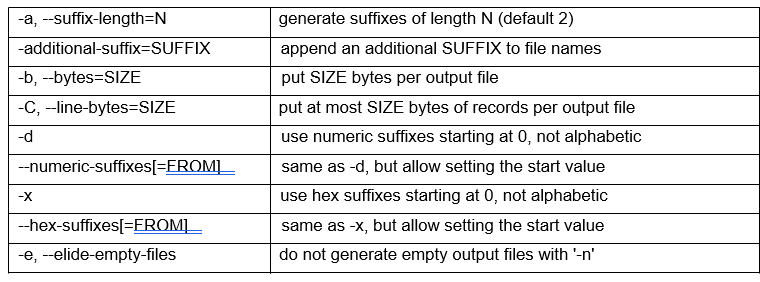


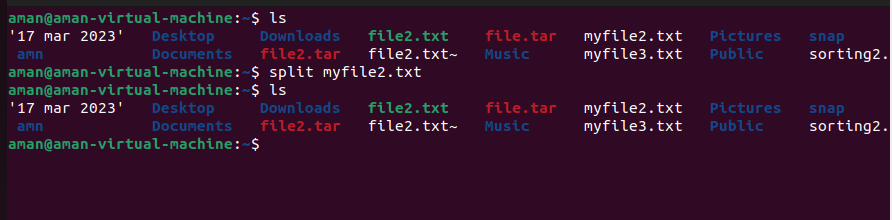
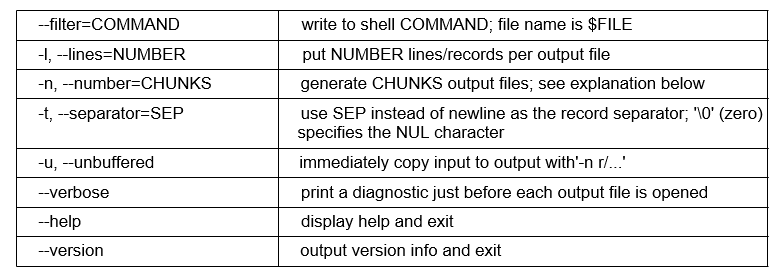
****

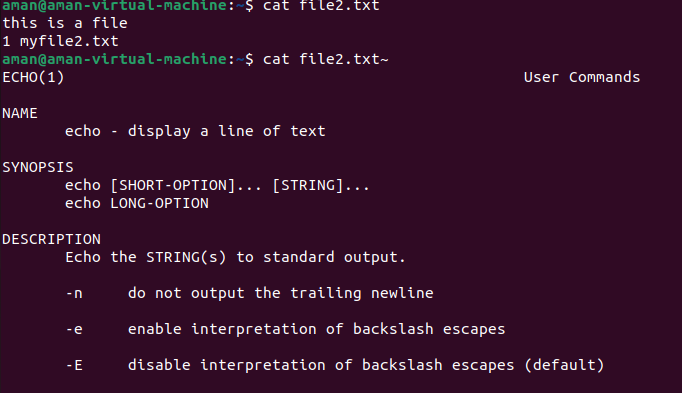
**5. split :** split a file into pieces

**SYNOPSIS**

split [OPTION]... [FILE [PREFIX]]



****

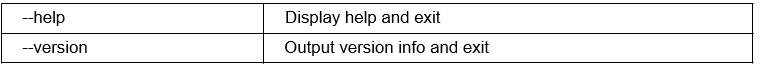
****

**6. sleep:** delay for a specified amount of time

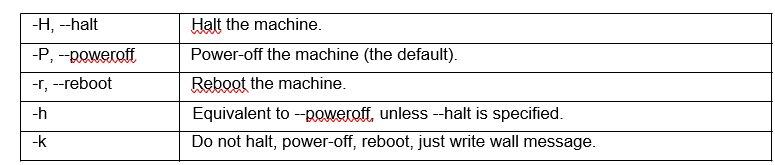
**SYNOPSIS**

sleep NUMBER[SUFFIX]

sleep OPTION

****

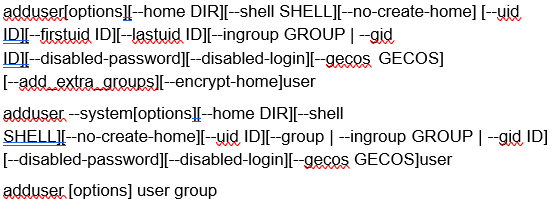
**7. shutdown:** Halt, power-off or reboot the machine

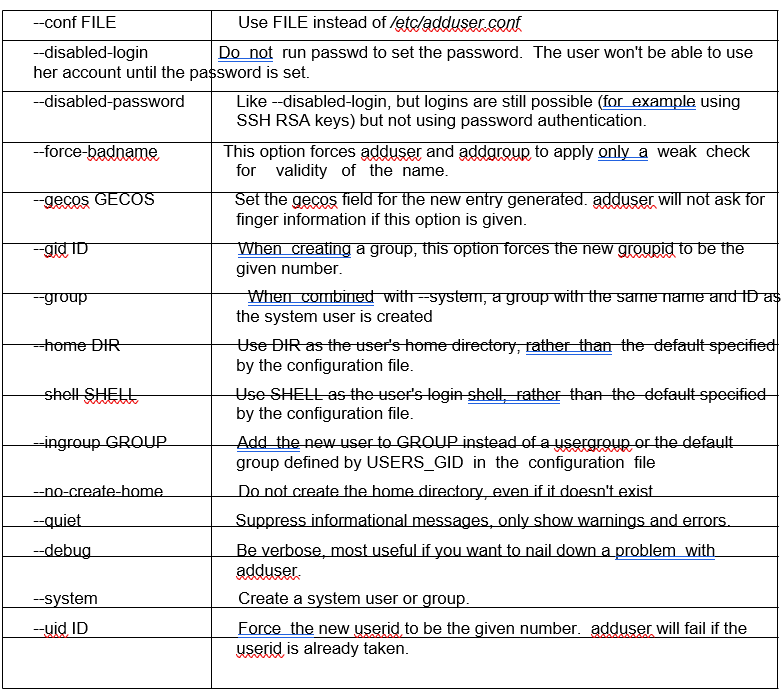
****

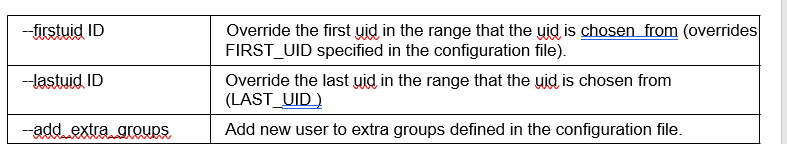
**8. adduser**

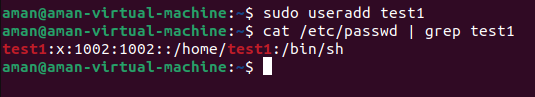
add a user to the system

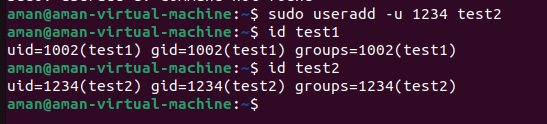
**SYNOPSIS**

****

****

****

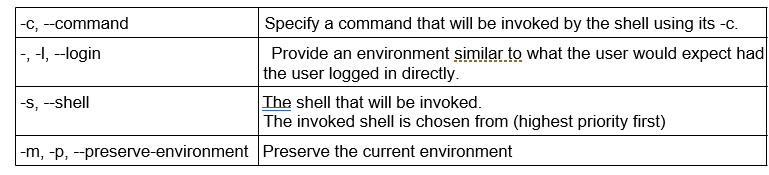
****

****

**9. su:** change user ID or become superuser

**SYNOPSIS**

su [options] [username]

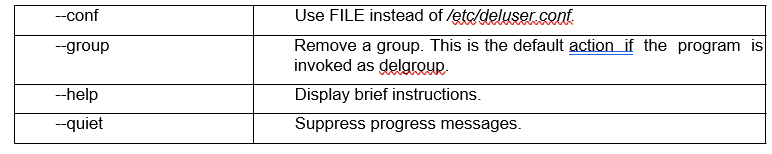


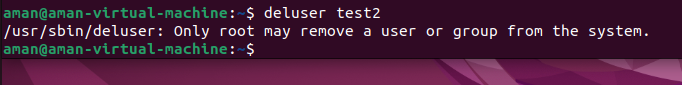


**10. deluser:** remove a user from the systemdedeluserde

**SYNOPSIS**

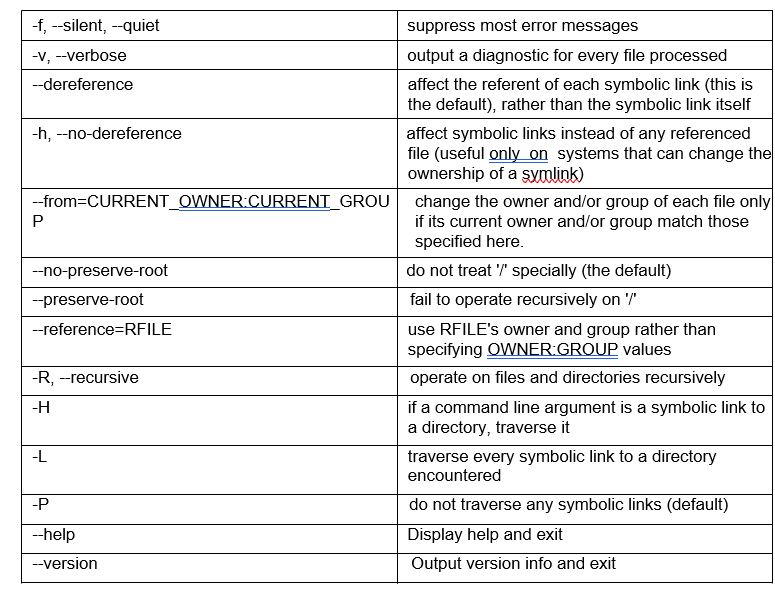
deluser[options] [--remove-home] [--remove-all-files] [--backup] user

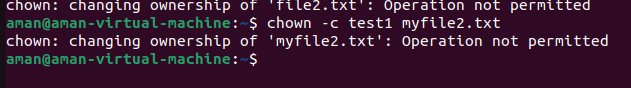


**11. chown:** change file owner and group

**SYNOPSIS**

chown [OPTION]... [OWNER][:[GROUP]] FILE... chown [OPTION]... --reference=RFILE FILE...





# **LAB 5 COMMANDS**

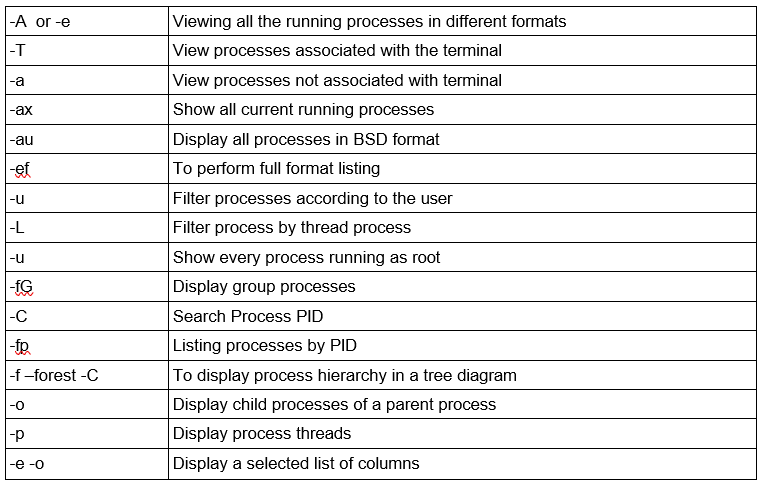
### ps:

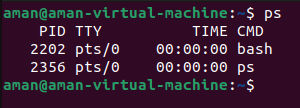
ps command is for viewing information related with the processes on a system which stands as abbreviation for “Process Status”. ps command is used to list the currently running processes and their PIDs along with some other information depends on different options.

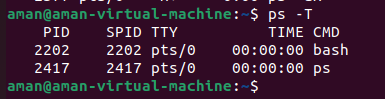
ps provides numerous options for manipulating the output according to our need.

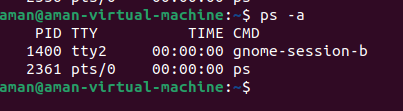
## SYNOPSIS

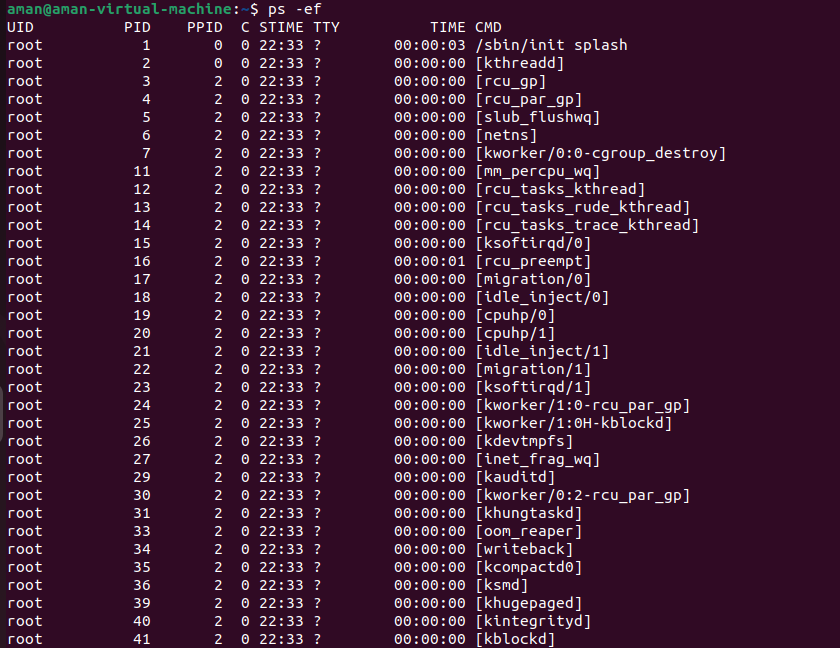
ps[options]

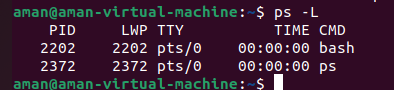


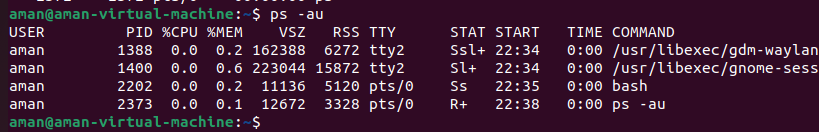


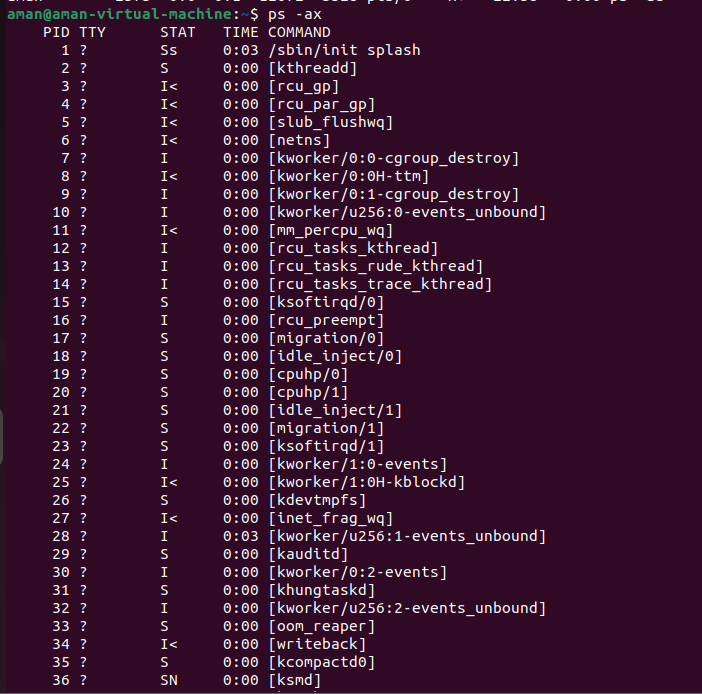


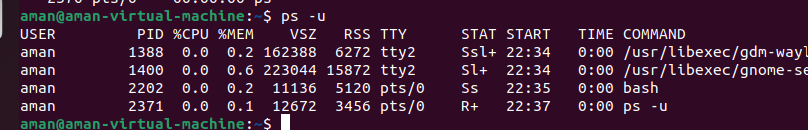


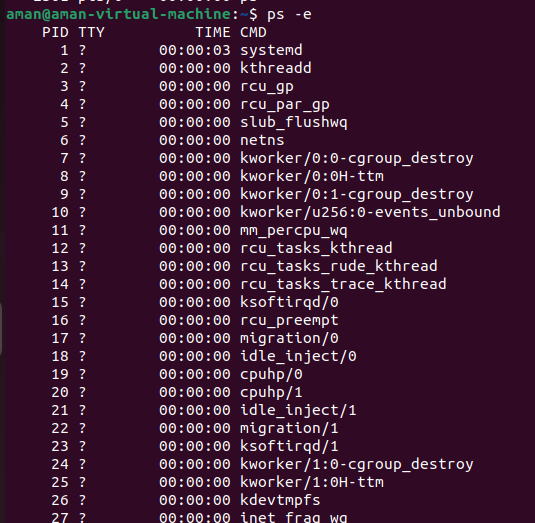










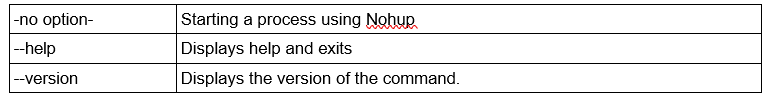


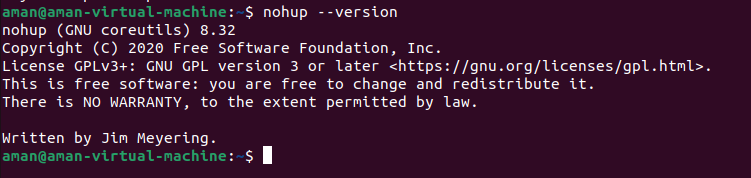
### 2.nohup (no hangup):

Nohup, short for no hang up is a command in Linux systems that keep processes running even after exiting the shell or terminal. Nohup prevents the processes or jobs from receiving the SIGHUP (Signal Hang UP) signal. This is a signal that is sent to a process upon closing or exiting the terminal.

## SYNOPSIS

nohup command arguments OR nohup [OPTIONS]

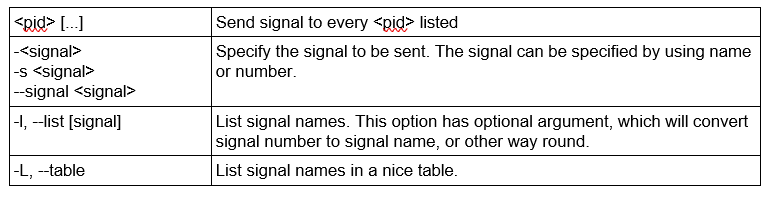


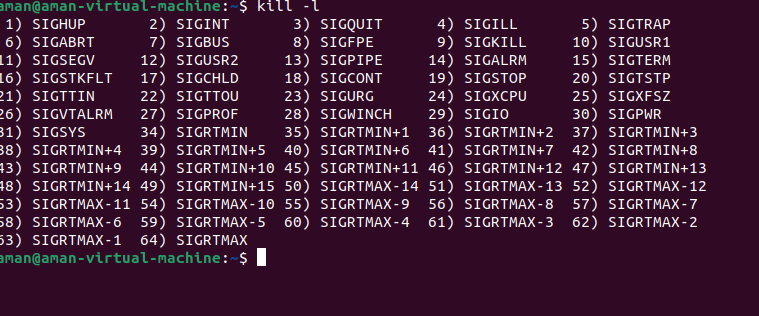


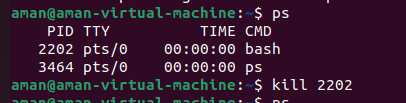
**3.kill:** It send a signal to a process

## SYNOPSIS

kill [options] <pid> [...]



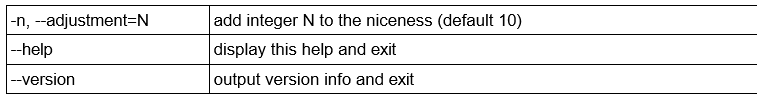


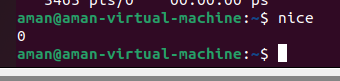


**4 .nice:** run a program with modified scheduling priority

## SYNOPSIS

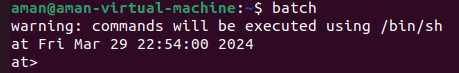
nice [OPTION] [COMMAND [ARG]...]

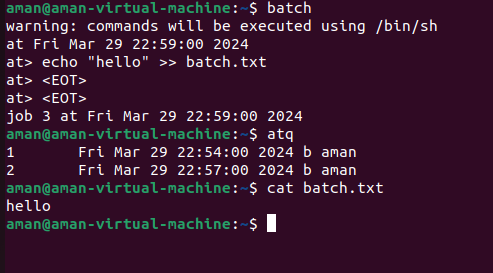




**5.batch:** sed to read commands from standard input or a specified file and execute them when system load levels permit i.e. when the load average drops below *1.5*

It is important to note that batch does not accepts any parameters.



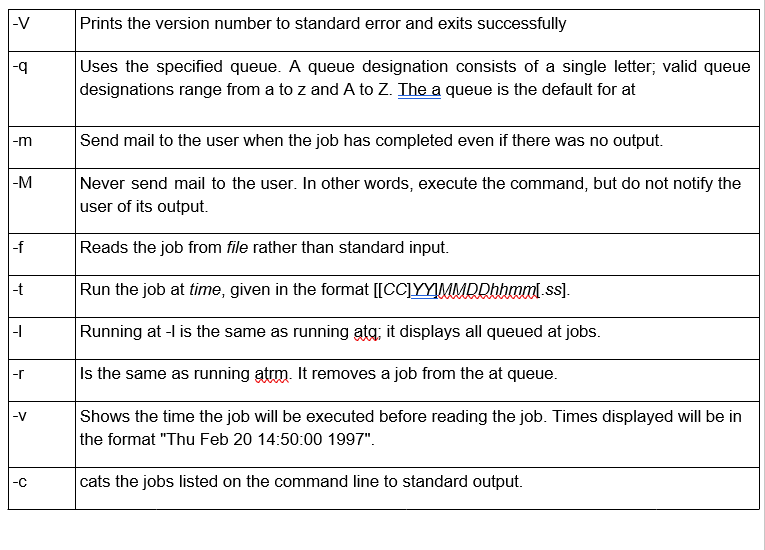


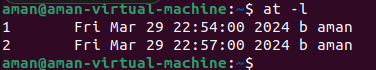
**6.at:** executes commands at a specified time.

## SYNOPSIS

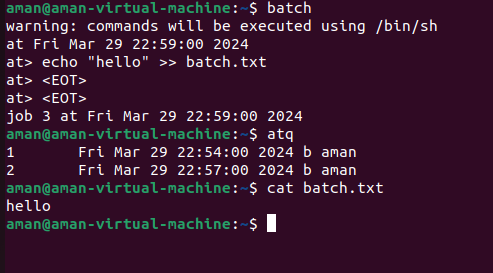
at [-V] [-q *queue*] [-f *file*] [-mMlv] *timespec*...

at [-V] [-q *queue*] [-f *file*] [-mMkv] [-t *time*] at -c *job* [*job*...]









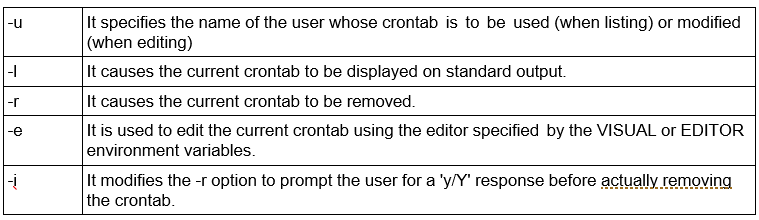
### 7.crontab:

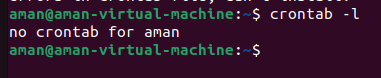
The crontab is a list of commands that we want to run on a regular schedule, and also the name of the command used to manage that list. Crontab stands for “cron table, ” because it uses the job scheduler cron to execute tasks.

## SYNOPSIS

crontab [ -u user ] file

crontab [ -u user ] [ -i ] { -e | -l | -r }

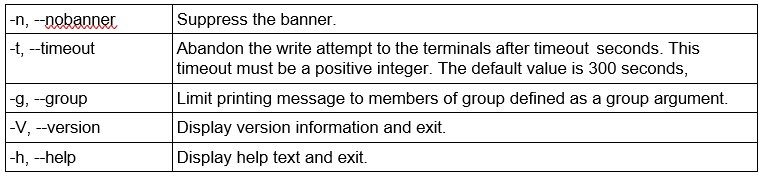




**8.wall:** write a message to all users

## SYNOPSIS

wall [-n] [-t timeout] [-g group] [message | file]

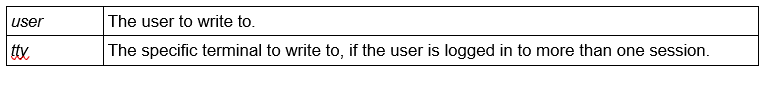


### 9.write:

The write command sends a message to another user. The write utility allows you to communicate with other users by copying lines from your terminal to theirs. When you run the write command, the user you are writing to gets a message of the format:

## SYNOPSIS

write *user* [*tty*]

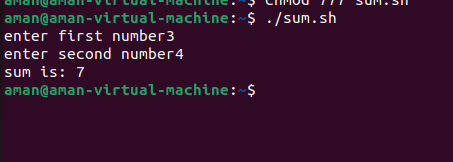


**LAB 7 COMMANDS**

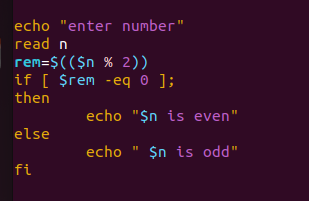
Programs:

* Addition of two numbers



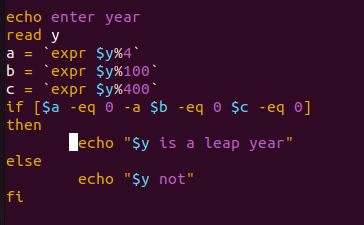


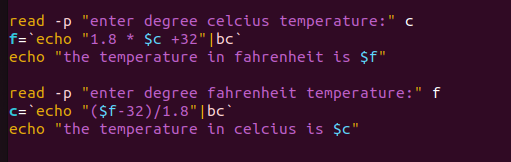
* Check whether the number entered is odd/even

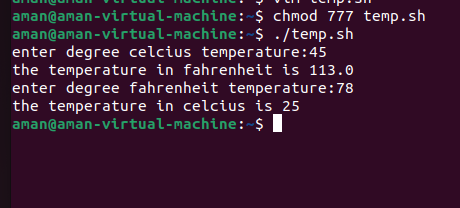




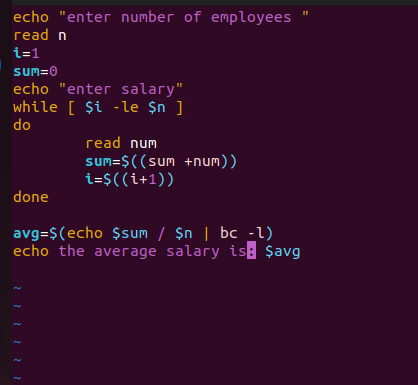
* Check whether the year is leap or not

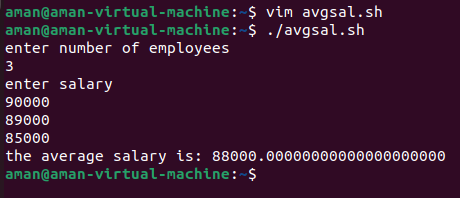


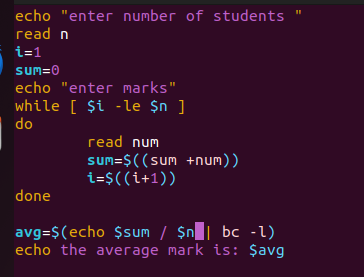


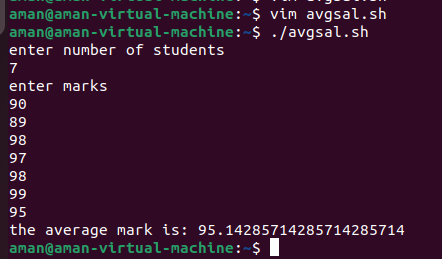


* Calculate the average salaries of employees

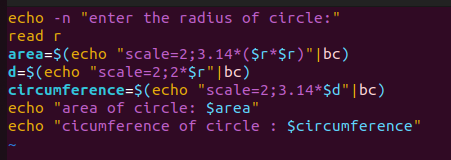


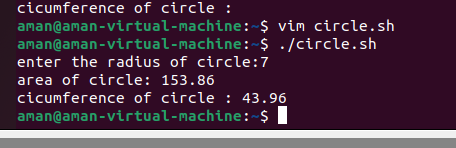
* 
* Calculate the average marks of students



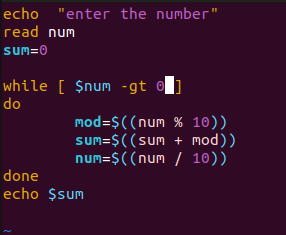


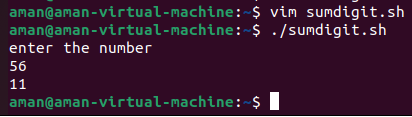
* To find circumference and area of circle



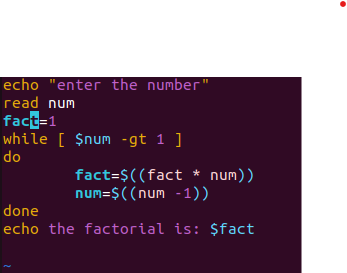


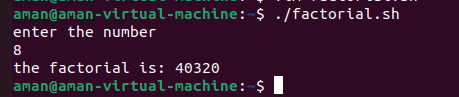
* To find the sum of digits of a number.



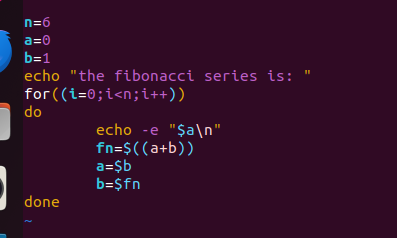


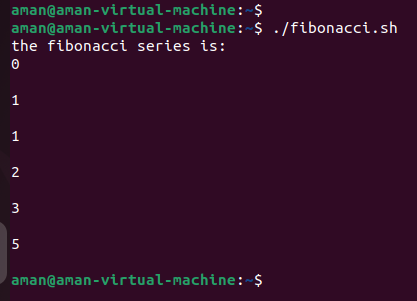
* Find the factorial of a number



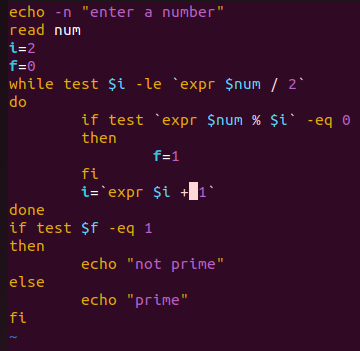


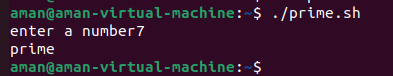
* Find the fibonacci series





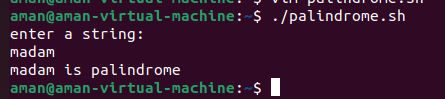
* Check whether a number is prime or not



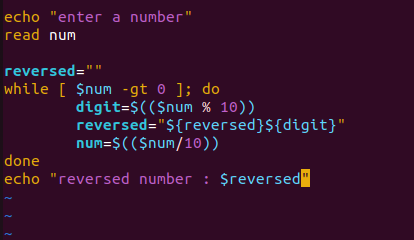


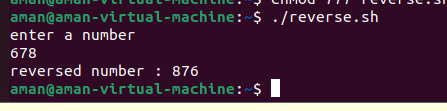
* Check whether the number is palindrome or not



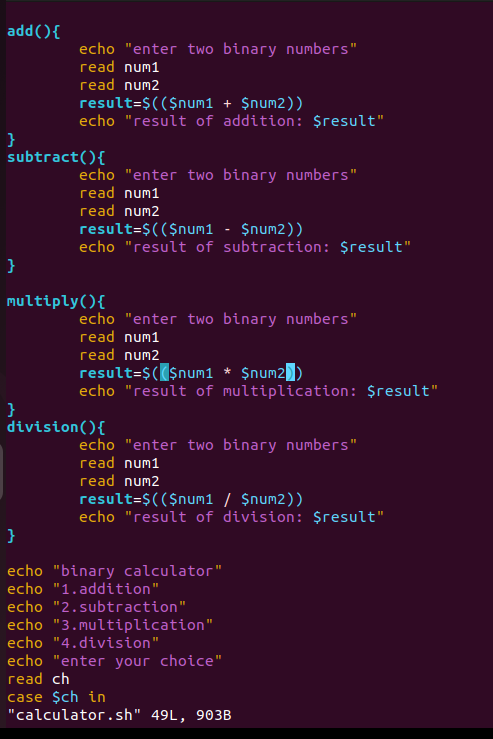


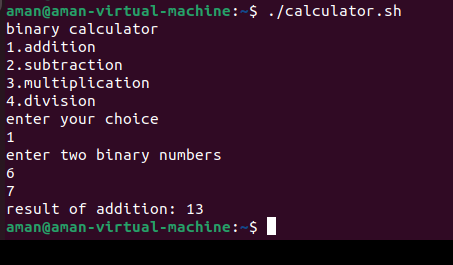
* Reverse of a given number





* To create a binary calculator using switch case





* Linear Search
* Binary search
* Bubble sort