MSc Computational Sciences (FYIMP)

Department of IT

Kannur University

Foundations of Computer Science

KU2DSCCSE102

Credits = 4

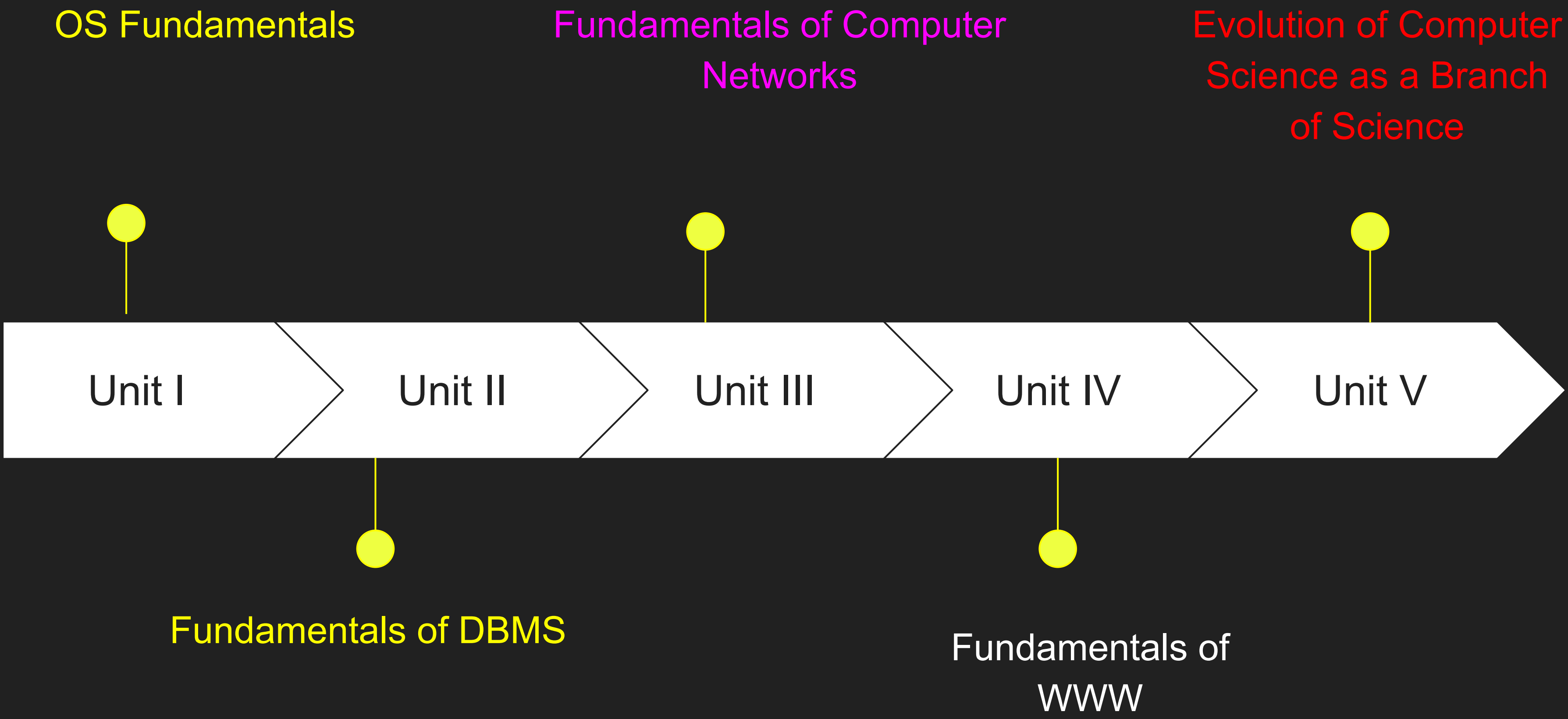
Course Objectives

- To understand the basics of Operating System
- To understand the basics of Database Management System
- To understand the basics of Computer Networks
- To understand the basics of Web Design

Course Outcomes

Student will be able to

- Explain the basics of Operating System
- Illustrate various features of Database Management System
- Explain the fundamentals of Computer Networks
- Explain the basics of web design and design simple static websites



Unit I

OS Fundamentals

Unit II

Fundamentals of DBMS

Unit III

Fundamentals of Computer Networks

Unit IV

Fundamentals of WWW

Unit V

Evolution of Computer Science as a Branch of Science

Class 1

Unit I

Definition of Computer Science - Classifications of Computer - Basic Building Block of Computer - Vonn Nueman Concept - Computer Hardware and Software - Classification of Software. Operating Systems: Definition - Structure - Types - Functions. Features of Windows and Linux Operating Systems. Case Study: Basic shell commands in Linux: Introduction to Shell Commands - Basic Folders - File and Folder Management: listing files, viewing contents in files, creating and deleting folders, creating files - moving and copying files and/or folders, man pages, setting permissions on files/directories. listing users that are logged in, listing the current user on the shell, listing all the details of the current user on the shell, listing the path of the current folder, adding a user, listing users that are currently created in the system, displaying the currently running process, the Kernel name, the processor name, the details of the operating system, and the information about the primary and secondary memory installed in the device.

What is Computer Science?

What is Computer Science?

It's all about problem solving!

What is Computer Science?

Its all about developing correct and efficient solutions for a problem!

What is Computer Science?

Computer Science is the scientific and practical approach to computation and its applications.

What is Computer Science?

Computer Science is a science of abstraction - creating the right model for a problem (representation) and devising the appropriate mechanizing techniques to solve it (algorithm).

What is Computer Science?

To tackle this “simple” issue Computer Scientists do a lot of things!

They study the very nature of computing to determine which problems are (or are not) computable.

What is Computer Science?

To tackle this “simple” issue Computer Scientists do a lot of things!

They compare various algorithms to determine if they provide a correct and efficient solution to a concrete problem.

What is Computer Science?

To tackle this “simple” issue Computer Scientists do a lot of things!

They design programming languages to enable the specification and expression of such algorithms.

What is Computer Science?

To tackle this “simple” issue Computer Scientists do a lot of things!

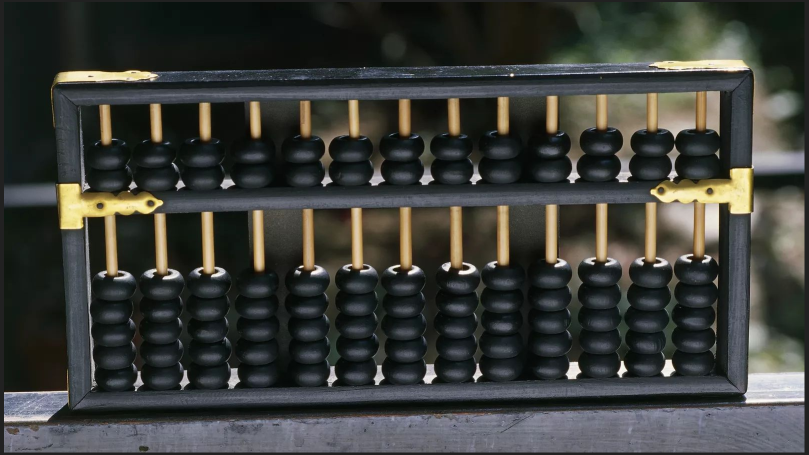
They design, evaluate, and build computer systems that can efficiently execute such specifications.

History of Computer Science

- Foundations of Computer Science were laid by our ancestors even long before the invention of modern computers!

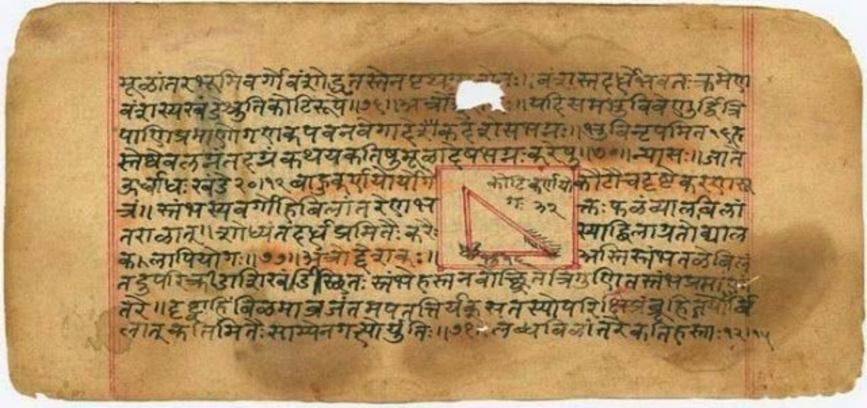
History of Computer Science

- Ancient humans used pebbles for their calculation!
- Later the volume, variety and veracity of the need increased - (s)he realised that a mere set of pebbles were not enough!
- See our ancestors had a real problem to solve!



History of Computer Science

- Abacus was invented and was is in use as early in 2400 BC!
- They did their additions and subtractions more elegantly.
- But how did they knew how to add and subtract?

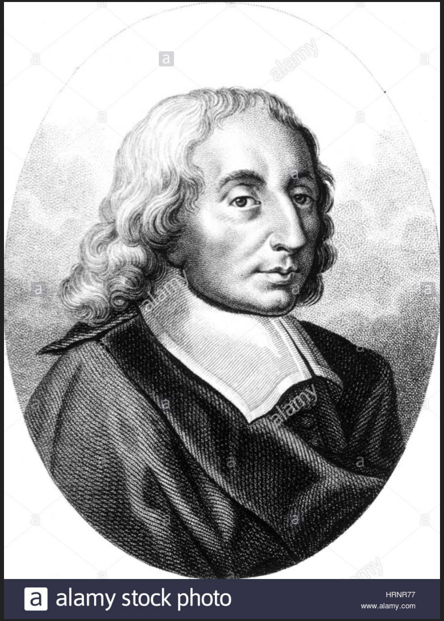


History of Computer Science

- Rules (say algorithms) that describe the steps for basic mathematical operations were authored by our ancients - way ahead of the inventions of any devices!
- The ancient Sanskrit book Shulba Sutras, or "Rules of the Chord" was written in 800 BC.
- Rules for constructing geometric objects like altars were meticulously outlined in this ancient book

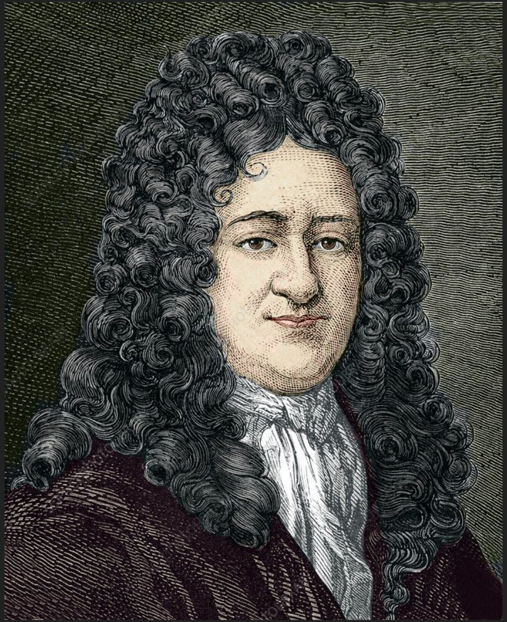
History of Computer Science

- Muhammad Al-Khowarizmi developed the procedures we commonly use when adding, subtracting, multiplying and dividing two decimal numbers with pencil and paper.
- The word algorithm was derived from his name!
- His works were based on 7th century Indian mathematician Brahmagupta.



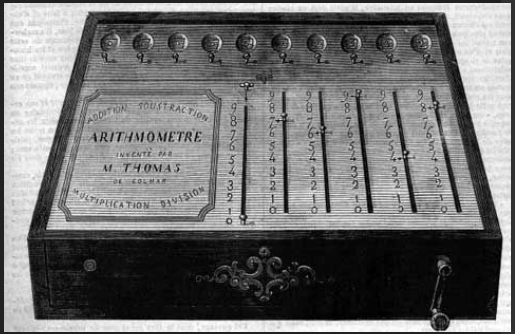
History of Computer Science

- Several attempts of "automatic" calculators were reported.
- Blaise Pascal designed and constructed the first working mechanical calculator in 1642.



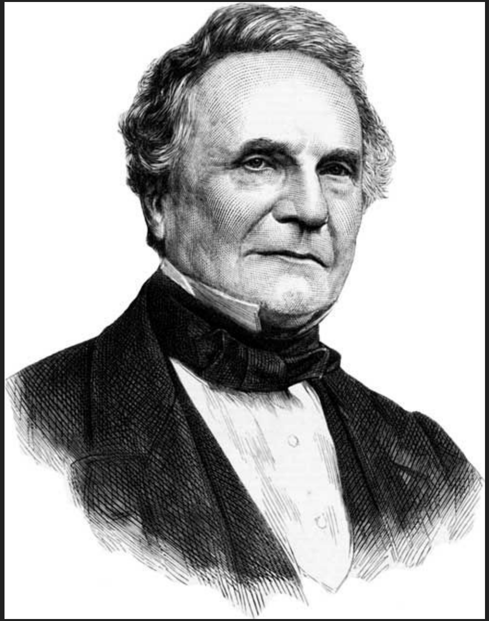
History of Computer Science

- In 1673 Gottfried Leibniz demonstrated a digital mechanical calculator, called the 'Stepped Reckoner'.
- His major contribution was documentation of the binary number system.



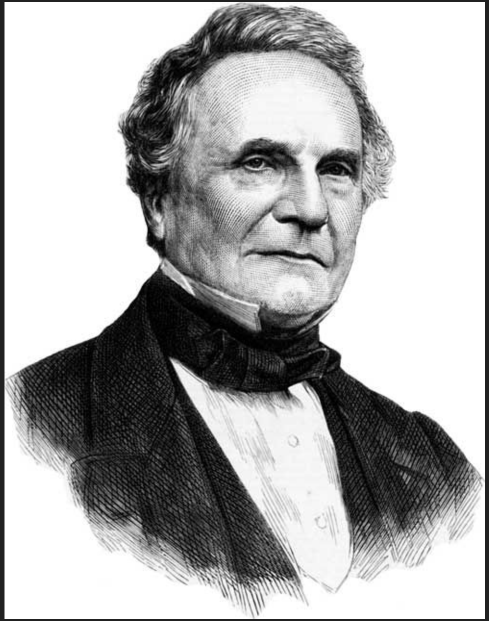
History of Computer Science

- In 1820, Thomas de Colmar launched the mechanical calculator called as simplified arithmometer.
- This device was the first calculating machine strong and reliable enough to be used daily in an office environment.



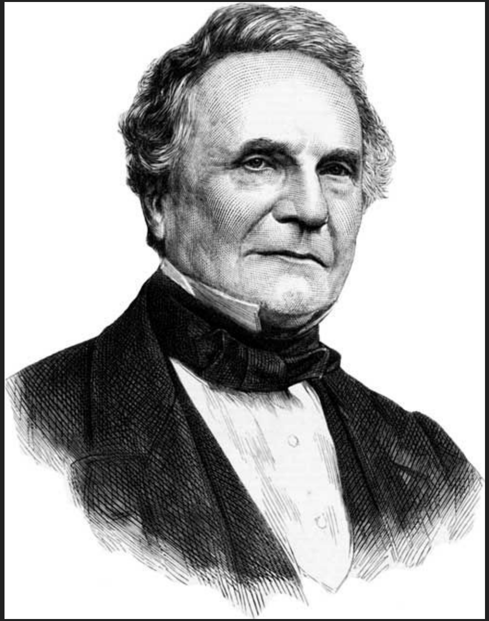
History of Computer Science

- Charles Babbage started the design of the first automatic mechanical calculator
- He designed difference engine in 1822.



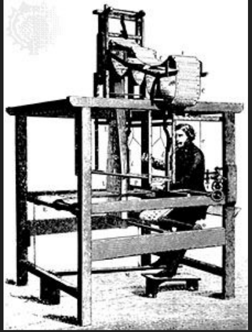
History of Computer Science

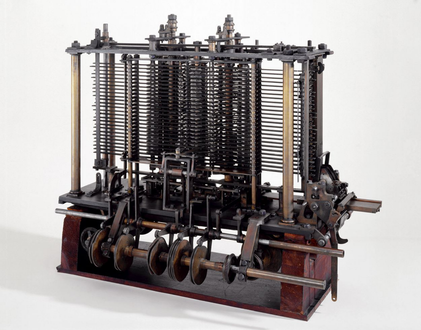
- Humans successfully implemented “machines” to do calculations.
- But still they had a problem - how to make it programmable?!



History of Computer Science

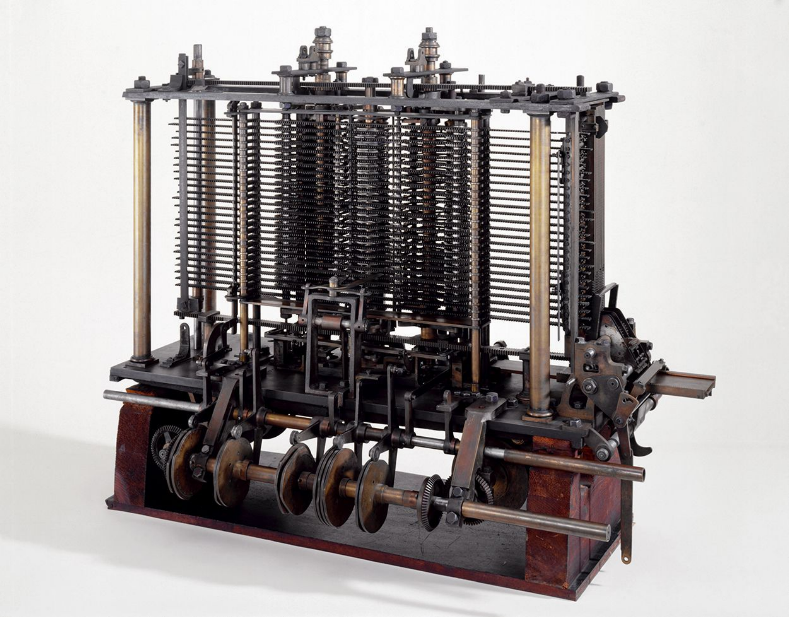
- Charles Babbage had a dream about “programmable” calculator!
- He started the design of “Analytical Engine in 1834.
- Within two years he could sketch out the design of his dream machine!





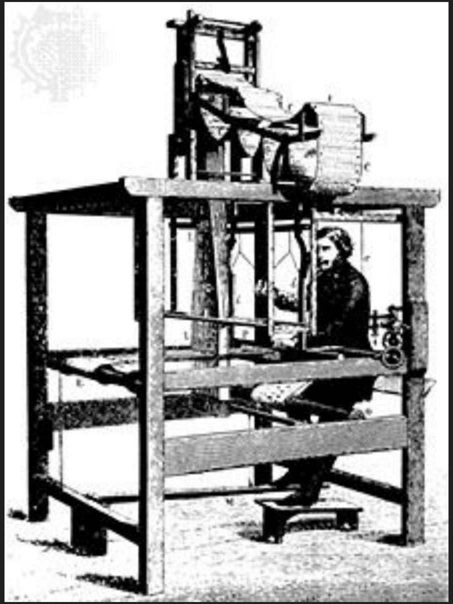
History of Computer Science

- Taking clues from jacquard Loom, Charles Babbage decided to make use of punched cards to make his device programmable!
- A true “interdisciplinary research”!



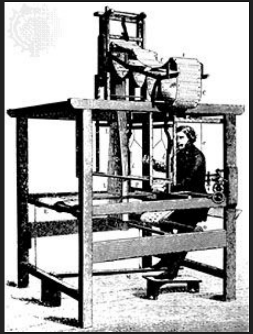
History of Computer Science

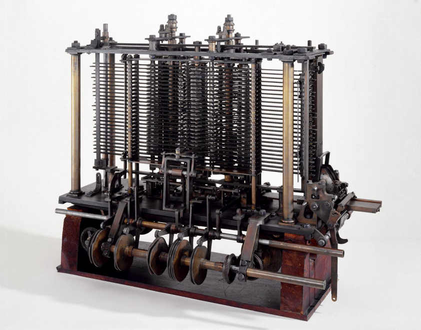
- Amazingly Charles Babbage had sketched out many of the salient features of the modern computer in his design!



History of Computer Science

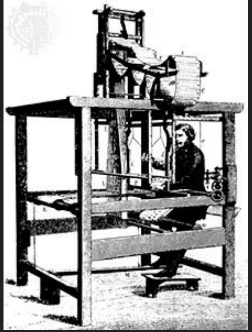
- Prior to 18th century, beautiful woven design was hand made.
- Only few experts could do this kind of work - at high costs.
- In 1804 Joseph-Marie Jacquard developed the Jacquard Loom - a significant step in mehanising pattern design on textiles!
- Jacquard used punched cards to create many designs with a machine!

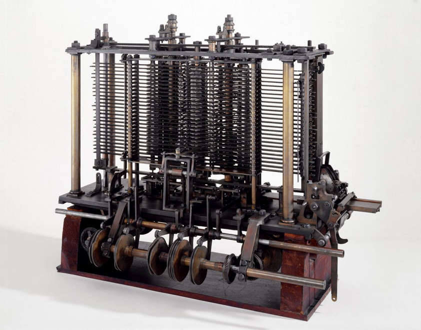




History of Computer Science

- Taking clues from jacquard Loom, Charles Babbage decided to make use of punched cards to make his device programmable!
- A true “interdisciplinary research”!





- Taking clues from jacquard Loom, Charles Babbage decided to make use of punched cards to make his device programmable!
- A true “interdisciplinary research”!*

History of Computer Science

History of Computer Science

- Along with “programmable” devices, there has to be program design and programmers!!!
- Ada Lovelace, the countess of Lovelace, and daughter to the poet Lord Byron was a friend of Charles Babbage.

History of Computer Science

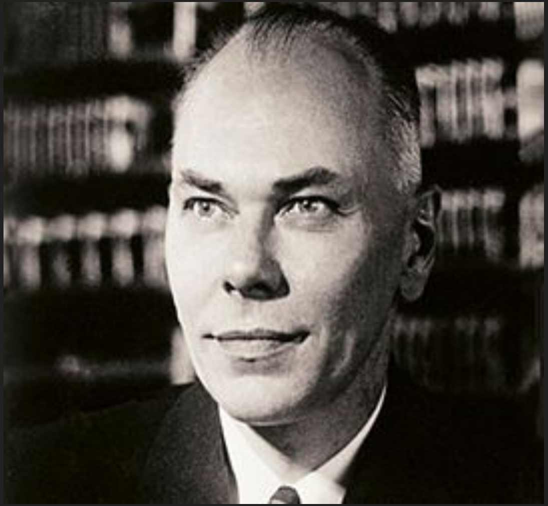
- In 1843, Lovelace translated an article on the Analytical Engine by Italian Engineer Luigi Menabrea.
- Her translation ran thrice the length of the original manuscript!
- In these notes, she included, an algorithm to compute the Bernoulli numbers - and this is considered as the first algorithm!

History of Computer Science

- Since the technology was not ripe enough, Babbage could not make the machine.
- Even Ada’s algorithm remained in paper!

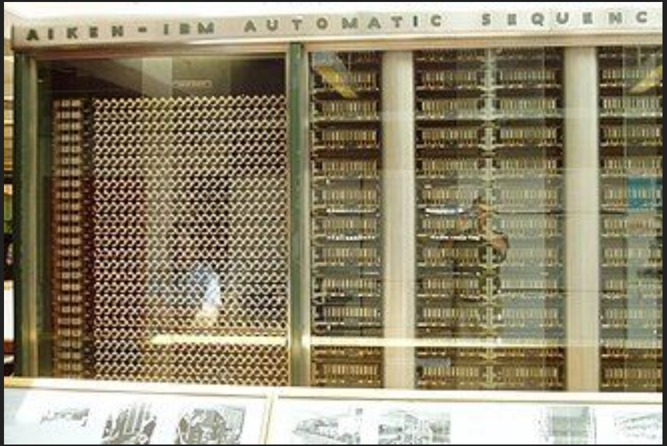
History of Computer Science

- From 2009 onward, 13 Oct is observed as Ada Lovelace Day (ALD).
- An international day of celebration to commemorate women in the STEM fields.



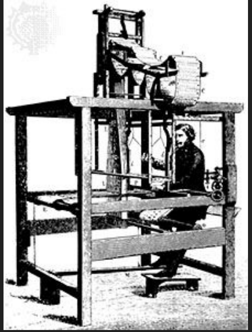
History of Computer Science

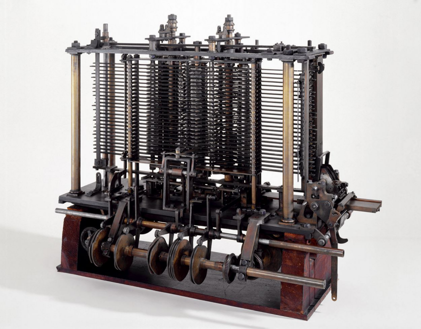
- It took 100 years to realise the dream of Charles Babbage!
- Howard Aiken, a US Physicist and a Pioneer in Computing, had designed a blueprint of a giant programmable calculator called as ASCC/Harvard Mark I.
- This was based on Babbage's analytical engine.



History of Computer Science

- In 1937, he convinced IBM about the necessity to develop this machine.
- In 1944, IBM installed the the machine at Harvard*.





https://history-computer.com/

History of Computer Science

Department of Information Technology, Kannur University

Computer System

- A computer system consists of both physical components (hardware) and programs (software) that collaborate each other to accomplish various tasks using the system
- Hardware includes physical elements like the processor, memory, storage, and input/output devices
- Software refers to a collection of programs (apps) or instructions that guide a computer on what actions to take and how to execute them.
- By functioning together, these elements process data, manage resources, and enable communication within the system and with external networks

Classification of Computers

- Computer systems can be classified based on size, purpose, and performance into the following categories:
    - Microcomputers (Personal Computers)
    - Minicomputers
    - Mainframe Computers
    - Supercomputers
    - Embedded Systems

Classification of Computers

- Computer systems can be classified based on size, purpose, and performance into the following categories:
    - Microcomputers (Personal Computers)
        - Small, affordable computers designed for individual use
        - The most commonly used type of computer
        - Available in compact and portable designs such as desktops, laptops, and smartphones
        - Equipped with microprocessors and applications to handle everyday tasks

Classification of Computers

- Computer systems can be classified based on size, purpose, and performance into the following categories:
    - Microcomputers (Personal Computers)
        - Applications of microcomputers are available in multiple fields such as education (for creating assignments, conducting research, and online learning), business (managing spreadsheets, emails, and presentations) and entertainment (gaming, streaming, and music)

Classification of Computers

- Computer systems can be classified based on size, purpose, and performance into the following categories:
    - Minicomputers (mid-range computers)
        - More powerful than microcomputers and support multiple users simultaneously
        - Capable of multitasking
        - Ideal for handling tasks that require moderate computing resources
        - Applications of these types of machines include managing databases in small organizations and scientific research where simultaneous data analysis is needed

Classification of Computers

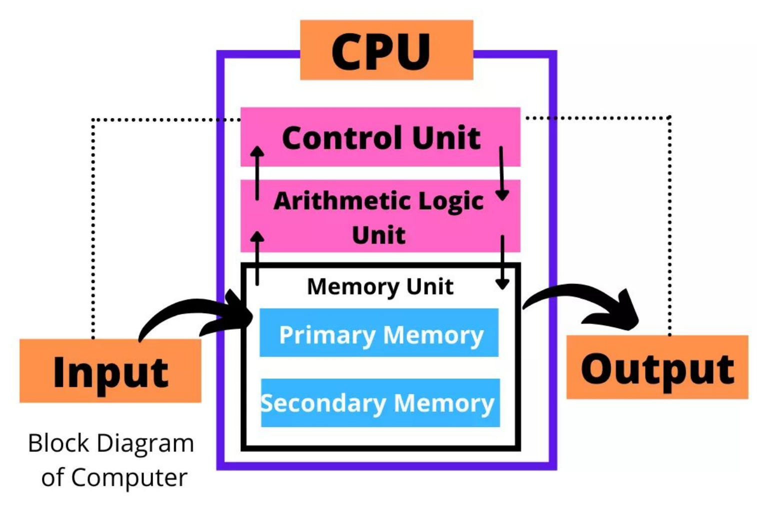
- Computer systems can be classified based on size, purpose, and performance into the following categories:
    - Mainframe computers (mid-range computers)
        - Mainframes are large and powerful machines designed to handle huge amounts of data and multiple simultaneous users
        - High reliability and efficiency in data processing
        - Excellent for tasks requiring heavy computation and storage
        - They are used by institutions such as banks (for processing millions of financial transactions daily) and airlines (for managing ticket bookings and flight schedules and governments (storing and processing census data

Classification of Computers

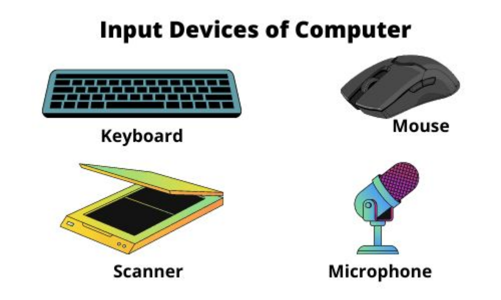
- Computer systems can be classified based on size, purpose, and performance into the following categories:
  - Supercomputers
    - The fastest and most powerful computers, designed for complex scientific calculations
    - Capable of performing billions to trillions of calculations per second
    - Extremely fast processing speeds due to parallel processing techniques
    - Specialized for solving complex problems
    - Applications include scientific research (simulating climate models, physics experiments, and astrophysical phenomena), medical research (drug discovery and genomics) and engineering (modeling aerodynamics and structural analysis)

Classification of Computers

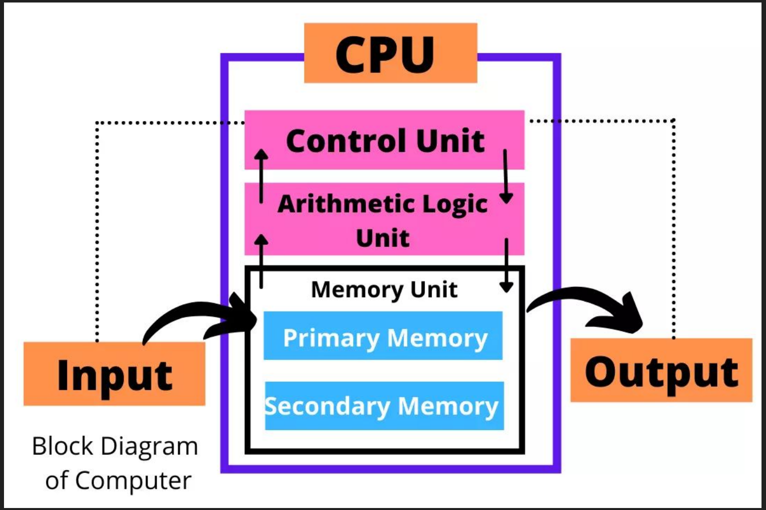
- Computer systems can be classified based on size, purpose, and performance into the following categories:
    - Embedded Systems
        - Embedded systems are computers integrated into devices to perform specific tasks. Unlike other computer types, these are not general-purpose systems
        - Minimal user interaction
        - Optimized for reliability and efficiency in dedicated tasks
        - Applications include consumer electronics (TVs, washing machines, and smart home devices), automobiles (airbag control, engine monitoring, and GPS navigation, and healthcare (heart rate monitors and infusion pumps)



Basic Building Blocks

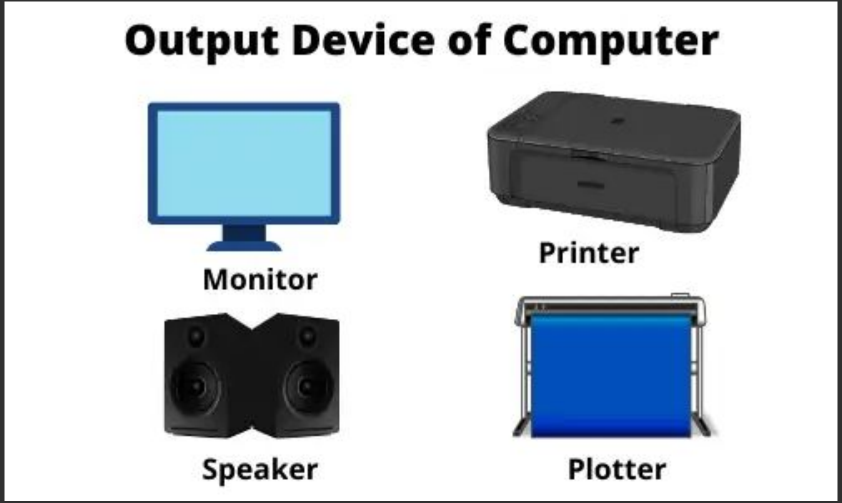


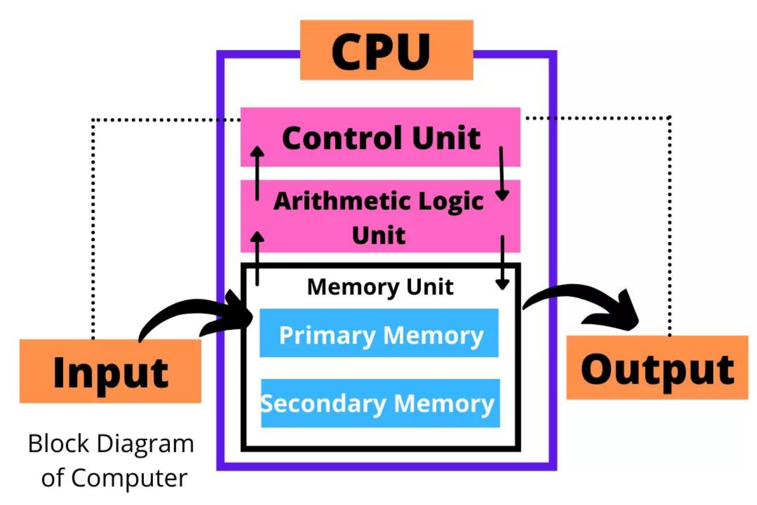
- Input Unit
  - The part of computer that helps us to give information or commands to the computer
  - Eg:- keyboard, mouse



Basic Building Blocks

- Output Unit
  - The part of the computer that shows or sends out the result of processing
  - Eg:- monitor, printer

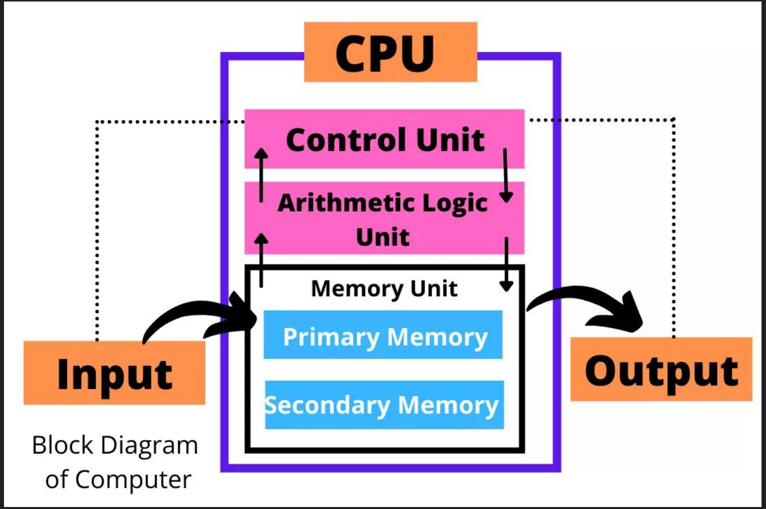




Block Diagram of Computer

Basic Building Blocks

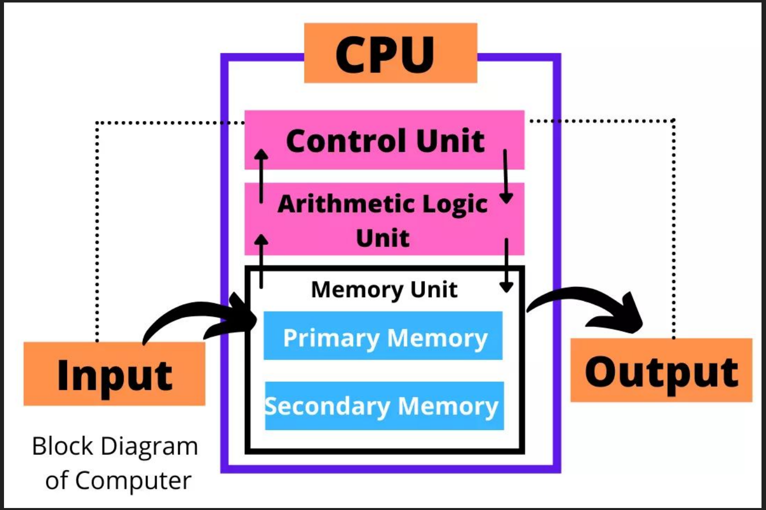
- The Central Processing Unit (CPU)
  - The primary component of a computer responsible for processing instructions and managing all other parts of the system, enabling the computer to function effectively
  - It has two subunits
    - Control Unit
    - Arithmetic and Logic Unit



Block Diagram of Computer

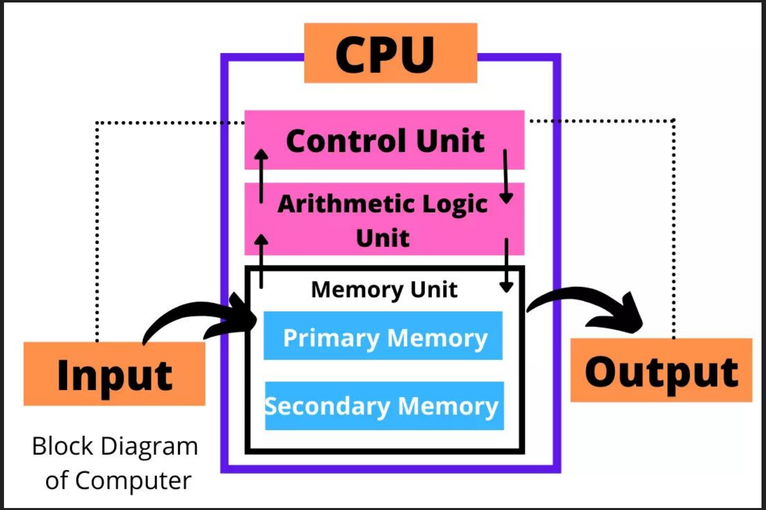
Basic Building Blocks

- The Central Processing Unit (CPU)
    - It has two subunits
        - The Control Unit (CU)
            - The part of the CPU that manages and directs how the different components of the computer work together to process instructions



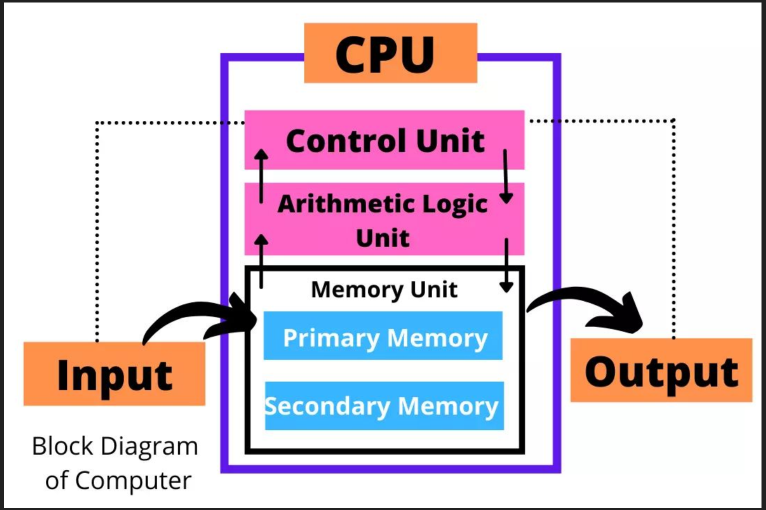
Basic Building Blocks

- The Central Processing Unit (CPU)
    - It has two subunits
        - The arithmetic unit (ALU)
            - The part of the CPU that performs math calculations and makes decisions based on logical comparisons



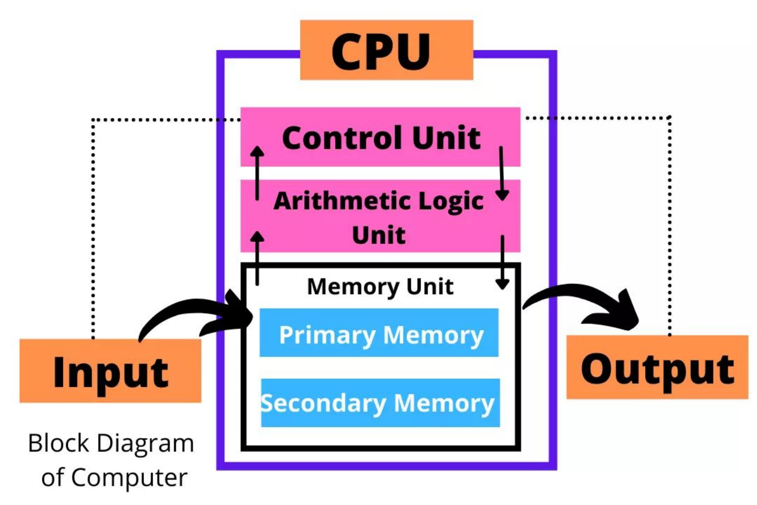
Basic Building Blocks

- Memory Unit
    - This component of a computer is responsible for storing data and instructions for the CPU to access as needed
    - It is of two kinds:
        - Primary Memory
        - Secondary Memory



Basic Building Blocks

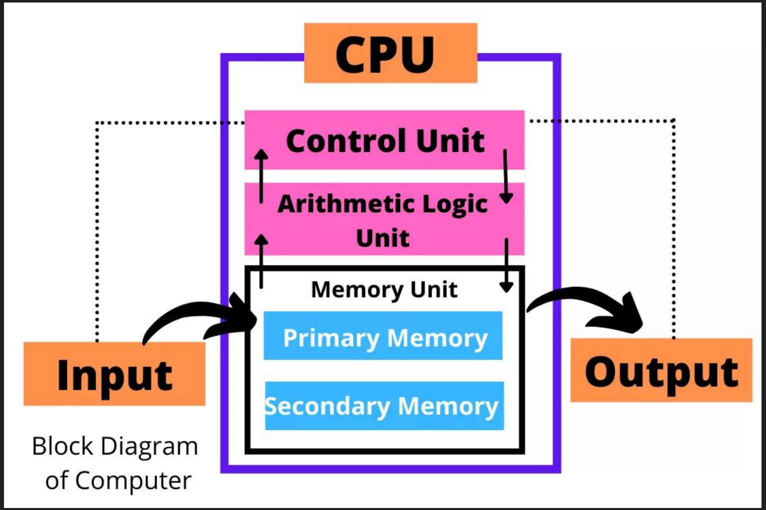
- Primary memory of a computer is the main storage area that holds data and programs currently in use, allowing for quick access by the CPU.
- Primary memory is like a scratch pad - CPU can only process the the instructions and data that are in primary memory
- It can not retain its contents when the computer is powered off - volatile memory
- RAM chip is an example for primary memory



Block Diagram of Computer

Basic Building Blocks

- Secondary memory of a computer is to permanently store data and programs
- Contents of secondary memory will remain even when the computer is powered off - Non Volatile
- Storage capacity of secondary memory is higher than primary memory
- Speed of secondary ,memory is less that primary memory
- Eg:- Hard Disk



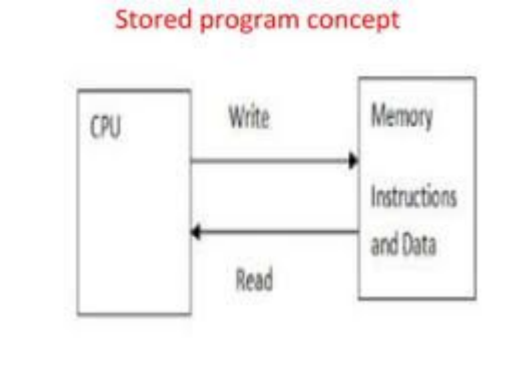
Block Diagram of Computer

Basic Building Blocks

- Secondary memory of a computer is to permanently store data and programs
- Contents of secondary memory will remain even when the computer is powered off - Non Volatile
- Storage capacity of secondary memory is higher than primary memory
- Speed of secondary ,memory is less that primary memory
- Eg:- Hard Disk

The Von Neumann Architecture

- John von Neumann (28/12/1903 – 08/02/1957) is a Hungarian-born American mathematician
- He revolutionized Computer Science by developing the stored-program concept in 1945, which became the foundation for modern computer architecture



The Von Neumann Architecture

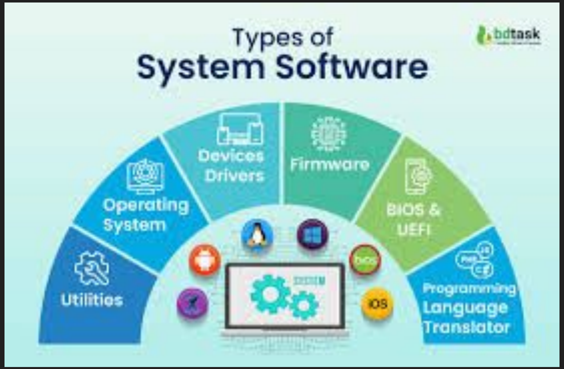
- The stored program concept means that a computer can keep both information and the steps to use that information in its memory, letting it run different tasks easily
- Hence the program and data need to be in primary memory for a machine to its processing
- Even today computers work with this concept!

The Von Neumann Architecture

- The The stored-program concept says, "Hey, before the CPU can do its magic, we need to store some programs or data in memory!"
- But here’s the kicker: Can our memory devices are keeping pace with the speedy CPU!
- Faster memory? Oh, that’s a luxury item — faster memories will cost more than the slower ones!
- RAM, our speedy superstar, zooms past secondary storage such as a hard disk in a race!
- That’s why RAM gets the VIP treatment as primary memory—it's where all the action happens before the CPU takes the stage!
- Just imagine a world where we have super-efficient, wallet-friendly primary memory that can keep up with the

Computer System

- A computer system consists of both physical components (hardware) and programs (software) that collaborate each other to accomplish various tasks using the system
- Hardware includes physical elements like the processor, memory, storage, and input/output devices
- Software refers to a collection of programs (apps) or instructions that guide a computer on what actions to tak and how to execute them
- By functioning together, these elements process data, manage resources, and enable communication within the system and with external networks

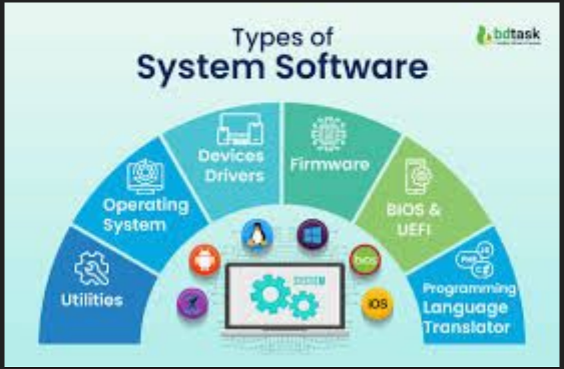


System Software

- There are many kinds of system software that caters various needs and purposes

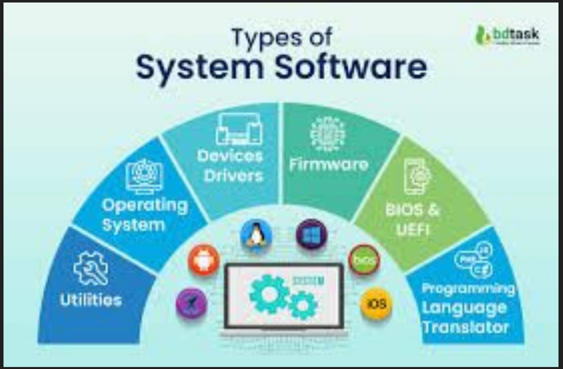
- Operating System: The main software that manages a computer's hardware and allows other programs to run, acting as a bridge between the user and the computer

- Language Translator: this type of system software is used for translating high-level programming languages into low-level machine languages



System Software

- There are many kinds of system software that caters various needs and purposes
- Operating System: The main software that manages a computer's hardware and allows other programs to run, acting as a bridge between the user and the computer
- Language Translator: this type of system software is used for translating high-level programming languages into low-level machine languages

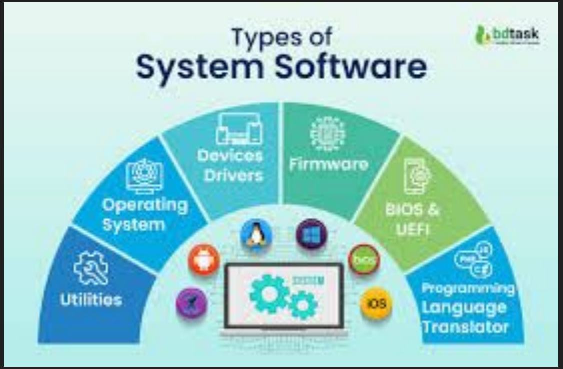


System Software

- There are many kinds of system software that caters various needs and purposes

- Utility software: Utility software is the software that optimizes, maintains, and controls computer resources. Antivirus software, backup software, etc., are examples of utility software.

- Device Driver: a tiny program that helps to draw a communication connection between the computer and different external hardware devices (printers, scanners, and keyboards)



System Software

- There are many kinds of system software that caters various needs and purposes

- Firmware: a type of software that is embedded with hardware devices. Usually, it is stored in ROM and cannot be modified or deleted in any way

- BIOS and UEFI: The full form of BOIS(basic input/ output system) and UEFI (unified extensible firmware interface). It’s a form of firmware code that starts working when the computer is powered on

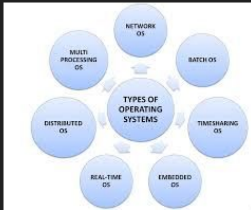
Operating System ^

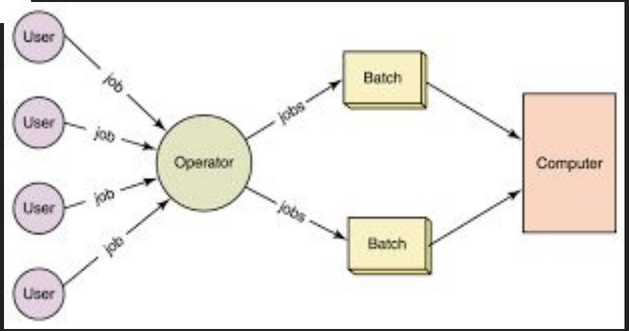
- An operating system (OS) is software that allows users to interact with a computer and manage its hardware

- The OS provides a way for users to control the computer, either through graphical menus and icons or by typing commands

- The operating system manages the computer's hardware components, ensuring they work together effectively to run programs

- An OS organizes and keeps track of files on the computer, allowing users to create, access, and manage their documents and data



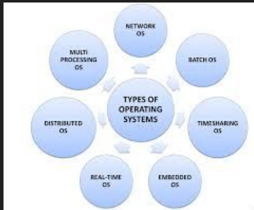


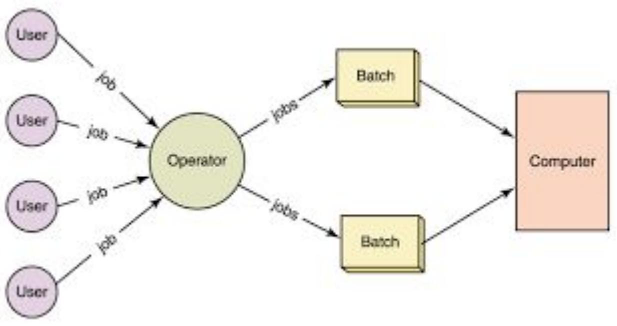
Types of Operating System

Batch Operating System

- Job Scheduling
    a. Processes jobs in batches without user interaction
    b. Jobs with similar needs are grouped together

- Automatic Job Execution
    a. Once jobs are submitted, they are processed automatically without user intervention until the batch is complete



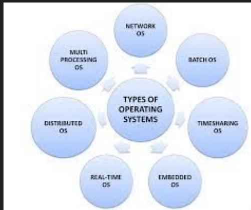


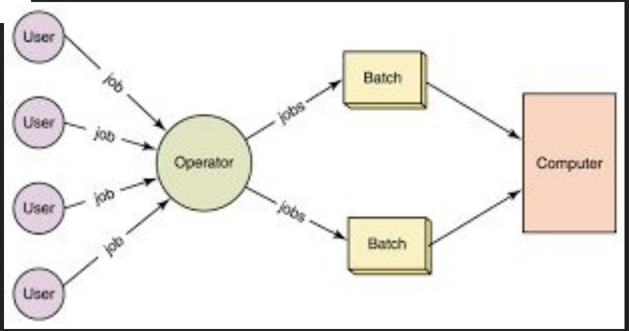
Types of Operating System

Batch Operating System

- Noninteractive
    a. Users submit jobs and do not interact with them while they are being processed, limiting flexibility during execution

- Sequential Processing
    a. Jobs are executed one after the other, reducing idle time by using system resources efficiently



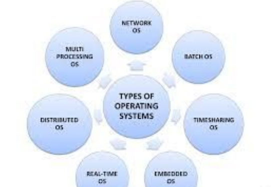


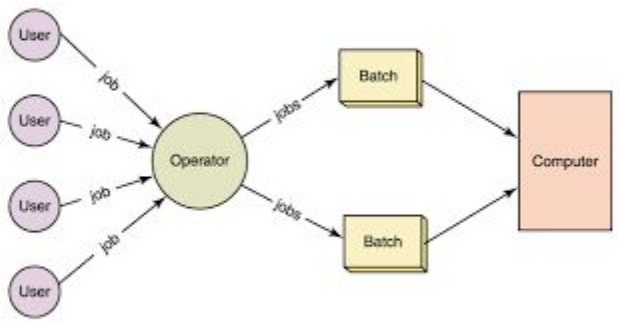
Types of Operating System

Batch Operating System

- Long Turnaround Time
    a. Jobs are processed in batches one at time, one after the other
    b. Each job has to wait for its turn

IBM OS/360 – Used on IBM System/360 mainframe computers in the 1960s. it was one of the earliest batch processing systems





Types of Operating System

Batch Operating System

- Long Turnaround Time
    a. Jobs are processed in batches one at time, one after the other
    b. Each job has to wait for its turn

IBM OS/360 – Used on IBM System/360 mainframe computers in the 1960s. it was one of the earliest batch processing systems

Foundations of Computer Science

- Have you thought about the impact that could be made by personal project works you venture into, while you try to complete your masters in Computational Science!

  ...

Foundations of Computer Science

Linux

- The birth of Linux is tied to the personal journey of Linus Torvalds, a Finnish computer science student
- In 1991, he was a student at the University of Helsinki in Finland
- He was using Minix operating system (a Unix like system used for educational purposes)
- Frustrated with the limitations of Minix, he thought about designing his own OS kernel!
- He released the initial version of Linux in August 1991

Foundations of Computer Science

Linux

- He released the initial version of Linux in August 1991
  - “I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).
  - I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)...”

Linux - Common Features

- Open Source: Linux is distributed under open source licenses, allowing users to view, modify, and distribute the source code
- Multitasking: Linux supports multitasking, allowing multiple processes to run concurrently
- Multiuser Support: Multiple users can log in and use the system concurrently. Each user has their own account and settings
- Security: Linux has a robust security model with features such as file permissions, user authentication, and access controls

Linux - Common Features

- Stability and Reliability: Linux is known for its stability and reliability. Many Linux distributions are used in server environments where uptime is critical
- Networking Capabilities: Linux has strong networking support, making it an excellent choice for servers and network - related tasks. It includes a wide range of networking utilities and protocols
- Portability: Linux is highly portable and can run on various hardware architectures. This adaptability makes it suitable for a wide range of devices, from embedded systems to servers

Linux - Common Features

- File System Support: Linux supports various file systems, including ext4, Btrfs, and XFS
- Command-Line Interface (CLI): Linux provides a powerful command - line interface (CLI) that allows users to interact with the system using commands. This is a key feature for advanced users and system administrators
- Graphical User Interface (GUI): While Linux is often associated with the command line, many distributions also include graphical user interfaces (GUIs) for a user - friendly experience. Popular desktop environments include GNOME, KDE, and Xfce

Linux - Common Features

- Device Support: Linux supports a wide range of hardware devices, and many drivers are included in the kernel.
- Package Management: Linux distributions use package management systems (e.g., apt, yum, dnf) to install, update, and remove software packages. This simplifies software management and dependency resolution
- Customizability: Users can customize nearly every aspect of the Linux system, from the kernel to the desktop environment. This flexibility allows users to tailor the system to their specific needs

Linux - Advantages

- Open Source: Distributed under open source licenses, allowing users to access and modify the source code. This promotes transparency, collaboration, and community-driven development
- Cost-Effective: Free to use, eliminating the cost of licensing fees. This makes it an economical choice for individuals, organizations, and businesses
- Stability and Reliability: Stable and and reliable particularly in server environments. It can operate for extended periods without the need for frequent reboots, reducing downtime
- Security: Has a strong security model, including user authentication, file permissions, and access controls. Security updates and patches are released regularly to address vulnerabilities

Linux - Advantages

- Customizability: Users can customize the Linux operating system extensively, from the kernel to the user interface. This flexibility allows individuals and organizations to tailor the system to their specific needs\

- Performance: Linux is designed to be efficient and performant, making it suitable for a wide range of applications

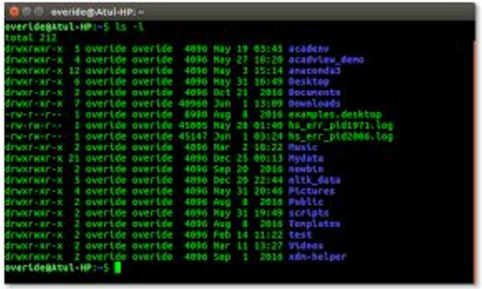
- Multitasking and Multiuser Support: Linux supports multitasking, allowing multiple processes to run concurrently. It is also a multiuser system, enabling multiple users to use the system simultaneously

Linux - Advantages

- Networking Capabilities: Has robust networking support, making it a preferred choice for servers and network-related tasks.
- Community Support: Has a vibrant and active community of users and developers. Online forums, documentation, and community-driven support provide assistance and resources for users
- Compatibility: Supports a wide range of hardware architectures and devices. Many hardware manufacturers provide Linux drivers, and community-driven efforts contribute additional support.
- Command-Line Interface (CLI): Offers a powerful command-line interface (CLI), which is favored by system administrators and advanced users for its fficiency and flexibility

Linux - Advantages

- Package Management: Linux distributions use package management systems (e.g., apt, yum, dnf) that simplify software installation, updates, and removal. Dependency resolution is handled automatically
- Scalability: Linux scales well, making it suitable for both small embedded systems and large-scale server environments. It can be adapted to meet the requirements of various computing environments.
- Long-Term Support (LTS) Releases: Many Linux distributions offer Long-Term Support (LTS) releases, ensuring stable and supported versions for an extended period. This is beneficial for organizations seeking a predictable and reliable platform

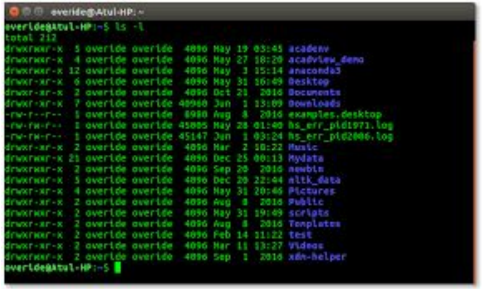


```
overide@Atul-HP:~$ ls -l
total 212
drwxrwxr-x  5 overide overide  4096 May 19 03:45 academy
drwxrwxr-x  4 overide overide  4096 May 27 18:20 acadview_demo
drwxrwxr-x 12 overide overide  4096 May  3 15:14 anaconda3
drwxr-xr-x  6 overide overide  4096 May 31 16:49 Desktop
drwxr-xr-x  2 overide overide  4096 Oct 21  2016 Documents
drwxr-xr-x  7 overide overide 40960 Jun  1 13:09 Downloads
-rw-r--r--  1 overide overide  8980 Aug  8  2016 examples.desktop
-rw-rw-r--  1 overide overide 45005 May 28 01:40 hs_err_pid1971.log
-rw-rw-r--  1 overide overide 45147 Jun  1 03:24 hs_err_pid2006.log
drwxr-xr-x  2 overide overide  4096 Mar  2 10:22 Music
drwxrwxr-x 21 overide overide  4096 Dec 25 00:13 Mydata
drwxrwxr-x  2 overide overide  4096 Sep 20  2016 newbie
drwxrwxr-x  5 overide overide  4096 Dec 20 22:44 nltk_data
drwxr-xr-x  4 overide overide  4096 May 31 20:46 Pictures
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Public
drwxrwxr-x  2 overide overide  4096 May 31 19:49 scripts
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Templates
drwxrwxr-x  2 overide overide  4096 Feb 14 11:22 test
drwxr-xr-x  2 overide overide  4096 Mar 11 13:27 Videos
drwxrwxr-x  2 overide overide  4096 Sep  1  2016 xdm-helper
overide@Atul-HP:~$ 
```

Foundations of Computer Science

Linux - Shell

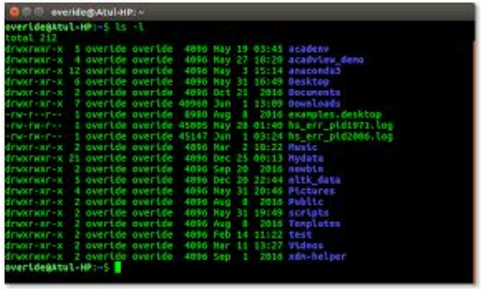
- A shell in Linux is a command-line interface (CLI) that allows users to interact with the operating system
- It serves as an intermediary between the user and the system's kernel, enabling command execution, script automation, and process management



```
overide@Atul-HP:~$ ls -l
total 212
drwxrwxr-x  5 overide overide  4096 May 19 03:45 academy
drwxrwxr-x  4 overide overide  4096 May 27 18:20 acadview_demo
drwxrwxr-x 12 overide overide  4096 May  3 15:14 anaconda3
drwxr-xr-x  6 overide overide  4096 May 31 16:49 Desktop
drwxr-xr-x  2 overide overide  4096 Oct 21  2016 Documents
drwxr-xr-x  7 overide overide 40960 Jun  1 13:09 Downloads
-rw-r--r--  1 overide overide  8980 Aug  8  2016 examples.desktop
-rw-rw-r--  1 overide overide 45005 May 28 01:40 hs_err_pid1971.log
-rw-rw-r--  1 overide overide 45147 Jun  1 03:24 hs_err_pid2006.log
drwxr-xr-x  2 overide overide  4096 Mar  2 10:22 Music
drwxrwxr-x 21 overide overide  4096 Dec 25 00:13 Mydata
drwxrwxr-x  2 overide overide  4096 Sep 20  2016 newbin
drwxrwxr-x  5 overide overide  4096 Dec 20 22:44 nltk_data
drwxr-xr-x  4 overide overide  4096 May 31 20:46 Pictures
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Public
drwxrwxr-x  2 overide overide  4096 May 31 19:49 scripts
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Templates
drwxrwxr-x  2 overide overide  4096 Feb 14 11:22 test
drwxr-xr-x  2 overide overide  4096 Mar 11 13:27 Videos
drwxrwxr-x  2 overide overide  4096 Sep  1  2016 xdm-helper
overide@Atul-HP:~$
```

Linux - Shell

- Several types of shells are available in Linux. Few of them are listed below
  - Bourne Shell (sh) – The original Unix shell
  - Bash (Bourne Again Shell) – A widely used, improved version of sh
  - C Shell (csh) – Includes C-like syntax and scripting features
  - Korn Shell (ksh) – Combines features of sh and csh, enhancing scripting capabilities
  - Z Shell (zsh) – An advanced shell with better customization and plugin support

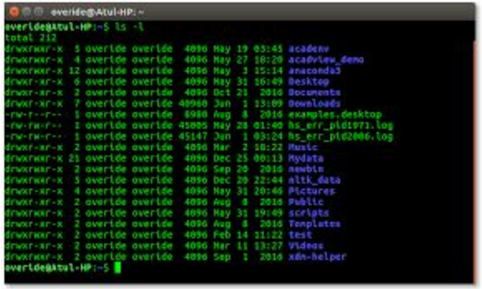


```
overide@Atul-HP:~$ ls -l
total 212
drwxrwxr-x  5 overide overide  4096 May 19 03:45 academy
drwxrwxr-x  4 overide overide  4096 May 27 18:20 acadview_demo
drwxrwxr-x 12 overide overide  4096 May  3 15:14 anaconda3
drwxr-xr-x  6 overide overide  4096 May 31 16:49 Desktop
drwxr-xr-x  2 overide overide  4096 Oct 21  2016 Documents
drwxr-xr-x  7 overide overide 40960 Jun  1 13:09 Downloads
-rw-r--r--  1 overide overide  8980 Aug  8  2016 examples.desktop
-rw-rw-r--  1 overide overide 45085 May 28 01:40 hs_err_pid1971.log
-rw-rw-r--  1 overide overide 45147 Jun  1 03:24 hs_err_pid2006.log
drwxr-xr-x  2 overide overide  4096 Mar  2 10:22 Music
drwxrwxr-x 21 overide overide  4096 Dec 25 00:13 Mydata
drwxrwxr-x  2 overide overide  4096 Sep 20  2016 newbie
drwxrwxr-x  5 overide overide  4096 Dec 20 22:44 nltk_data
drwxr-xr-x  4 overide overide  4096 May 31 20:46 Pictures
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Public
drwxrwxr-x  2 overide overide  4096 May 31 19:49 scripts
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Templates
drwxrwxr-x  2 overide overide  4096 Feb 14 11:22 test
drwxr-xr-x  2 overide overide  4096 Mar 11 13:27 Videos
drwxrwxr-x  2 overide overide  4096 Sep  1  2016 xdm-helper
overide@Atul-HP:~$ 
```

Linux - Shell

- Functions of a Shell
  - Command Execution – Users can run commands to manage files, processes, and system settings
  - Scripting – Shell scripts automate repetitive tasks using loops, conditions, and variables
  - Process Management – Users can start, stop, and monitor system processes
  - File Manipulation – Commands like ls, cp, mv, and rm help manage files and directories
  - User Interaction – Provides an environment for executing user commands and displaying outputs

Foundations of Computer Science

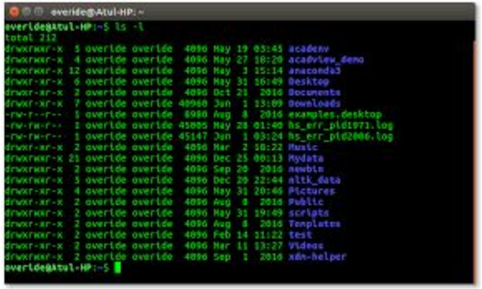


```
overide@Atul-HP: ~
overide@Atul-HP:~$ ls -l
total 212
drwxrwxr-x  5 overide overide  4096 May 19 03:45 academy
drwxrwxr-x  4 overide overide  4096 May 27 18:20 acadview_demo
drwxrwxr-x 12 overide overide  4096 May  3 15:14 anaconda3
drwxr-xr-x  6 overide overide  4096 May 31 16:49 Desktop
drwxr-xr-x  2 overide overide  4096 Oct 21  2016 Documents
drwxr-xr-x  7 overide overide 40960 Jun  1 13:09 Downloads
-rw-r--r--  1 overide overide  8980 Aug  8  2016 examples.desktop
-rw-rw-r--  1 overide overide 45005 May 28 01:40 hs_err_pid1971.log
-rw-rw-r--  1 overide overide 45147 Jun  1 03:24 hs_err_pid2006.log
drwxr-xr-x  2 overide overide  4096 Mar  2 10:22 Music
drwxrwxr-x 21 overide overide  4096 Dec 25 00:13 Mydata
drwxrwxr-x  2 overide overide  4096 Sep 20  2016 newbie
drwxrwxr-x  5 overide overide  4096 Dec 20 22:44 nltk_data
drwxr-xr-x  4 overide overide  4096 May 31 20:46 Pictures
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Public
drwxrwxr-x  2 overide overide  4096 May 31 19:49 scripts
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Templates
drwxrwxr-x  2 overide overide  4096 Feb 14 11:22 test
drwxr-xr-x  2 overide overide  4096 Mar 11 13:27 Videos
drwxrwxr-x  2 overide overide  4096 Sep  1  2016 xdm-helper
overide@Atul-HP:~$
```

Linux - Shell

- Functions of a Shell
  - Command Execution – Users can run commands to manage files, processes, and system settings
  - Scripting – Shell scripts automate repetitive tasks using loops, conditions, and variables
  - Process Management – Users can start, stop, and monitor system processes
  - File Manipulation – Commands like ls, cp, mv, and rm help manage files and directories
  - User Interaction – Provides an environment for executing user commands and displaying outputs

Foundations of Computer Science

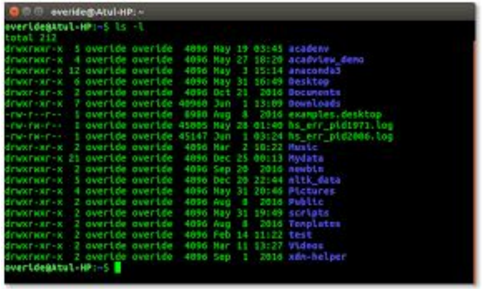


```
overide@Atul-HP:~$ ls -l
total 212
drwxrwxr-x  5 overide overide  4096 May 19 03:45 academy
drwxrwxr-x  4 overide overide  4096 May 27 18:20 acadview_demo
drwxrwxr-x 12 overide overide  4096 May  3 15:14 anaconda3
drwxr-xr-x  6 overide overide  4096 May 31 16:49 Desktop
drwxr-xr-x  2 overide overide  4096 Oct 21  2016 Documents
drwxr-xr-x  7 overide overide 40960 Jun  1 13:09 Downloads
-rw-r--r--  1 overide overide  8980 Aug  8  2016 examples.desktop
-rw-rw-r--  1 overide overide 45005 May 28 01:40 hs_err_pid1971.log
-rw-rw-r--  1 overide overide 45147 Jun  1 03:24 hs_err_pid2006.log
drwxr-xr-x  2 overide overide  4096 Mar  2 10:22 Music
drwxrwxr-x 21 overide overide  4096 Dec 25 00:13 Mydata
drwxrwxr-x  2 overide overide  4096 Sep 20  2016 newbie
drwxrwxr-x  5 overide overide  4096 Dec 20 22:44 nltk_data
drwxr-xr-x  4 overide overide  4096 May 31 20:46 Pictures
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Public
drwxrwxr-x  2 overide overide  4096 May 31 19:49 scripts
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Templates
drwxrwxr-x  2 overide overide  4096 Feb 14 11:22 test
drwxr-xr-x  2 overide overide  4096 Mar 11 13:27 Videos
drwxrwxr-x  2 overide overide  4096 Sep  1  2016 xdm-helper
overide@Atul-HP:~$
```

Linux - Shell

- Functions of a Shell
  - Command Execution – Users can run commands to manage files, processes, and system settings
  - Scripting – Shell scripts automate repetitive tasks using loops, conditions, and variables
  - Process Management – Users can start, stop, and monitor system processes
  - File Manipulation – Commands like ls, cp, mv, and rm help manage files and directories
  - User Interaction – Provides an environment for executing user commands and displaying outputs

Foundations of Computer Science



```
overide@Atul-HP:~$ ls -l
total 212
drwxrwxr-x  5 overide overide  4096 May 19 03:45 academy
drwxrwxr-x  4 overide overide  4096 May 27 18:20 acadview_demo
drwxrwxr-x 12 overide overide  4096 May  3 15:14 anaconda3
drwxr-xr-x  6 overide overide  4096 May 31 16:49 Desktop
drwxr-xr-x  2 overide overide  4096 Oct 21  2016 Documents
drwxr-xr-x  7 overide overide 40960 Jun  1 13:09 Downloads
-rw-r--r--  1 overide overide  8980 Aug  8  2016 examples.desktop
-rw-rw-r--  1 overide overide 45005 May 28 01:40 hs_err_pid1971.log
-rw-rw-r--  1 overide overide 45147 Jun  1 03:24 hs_err_pid2006.log
drwxr-xr-x  2 overide overide  4096 Mar  2 10:22 Music
drwxrwxr-x 21 overide overide  4096 Dec 25 00:13 Mydata
drwxrwxr-x  2 overide overide  4096 Sep 20  2016 newbie
drwxrwxr-x  5 overide overide  4096 Dec 20 22:44 nltk_data
drwxr-xr-x  4 overide overide  4096 May 31 20:46 Pictures
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Public
drwxrwxr-x  2 overide overide  4096 May 31 19:49 scripts
drwxr-xr-x  2 overide overide  4096 Aug  8  2016 Templates
drwxrwxr-x  2 overide overide  4096 Feb 14 11:22 test
drwxr-xr-x  2 overide overide  4096 Mar 11 13:27 Videos
drwxrwxr-x  2 overide overide  4096 Sep  1  2016 xdm-helper
overide@Atul-HP:~$
```

Linux - Shell

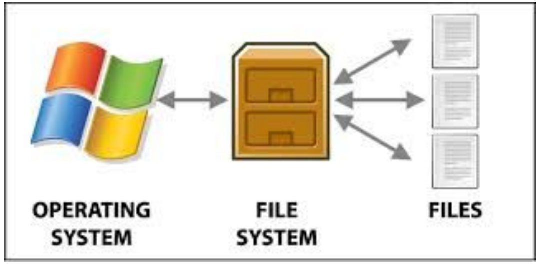
- Functions of a Shell
  - Command Execution – Users can run commands to manage files, processes, and system settings
  - Scripting – Shell scripts automate repetitive tasks using loops, conditions, and variables
  - Process Management – Users can start, stop, and monitor system processes
  - File Manipulation – Commands like ls, cp, mv, and rm help manage files and directories
  - User Interaction – Provides an environment for executing user commands and displaying outputs

Linux - Bash (Bourne Again Shell)

- The default shell in Ubuntu
- It is widely used because of its powerful scripting capabilities, user-friendly features, and compatibility with older Bourne shell scripts
- Current shell Ubuntu can be verified using the following command:

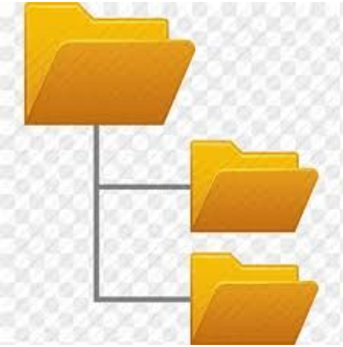
```
shell
```

Foundations of Computer Science



Secondary Memory - Data Storage

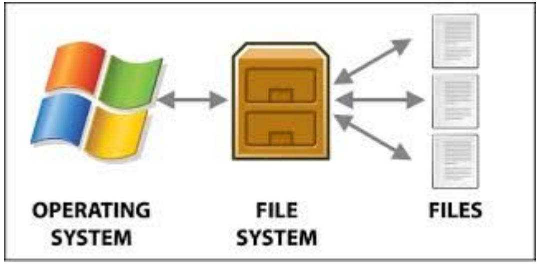
- We have learned that secondary storage is used to store data and programs permanently
- In any operating system, files and folders (also called directories) help organize and store data efficiently



Secondary Memory - Data Storage

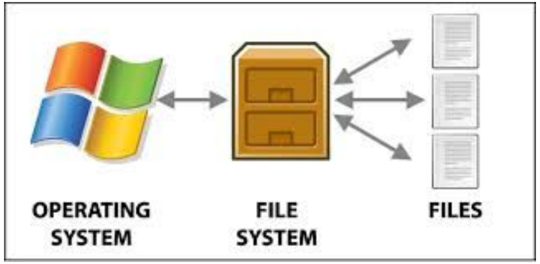
- A folder (or directory) is like a container that holds files and other folders

- It helps in organizing files systematically
  - Parent Folder – A folder that contains other folders or files
  - Subfolder – A folder inside another folder



Secondary Memory - Data Storage

- A file system is the way an operating system organizes and manages files and folders on a storage device (like a hard drive or USB)
- It defines how data is stored, accessed, and retrieved
- Examples of file systems include:
  - NTFS (Windows)
  - FAT32 (USB drives, older Windows systems)
  - ext4 (Linux-based systems)



Secondary Memory - Data Storage

- A file system is the way an operating system organizes and manages files and folders on a storage device (like a hard drive or USB)
- It defines how data is stored, accessed, and retrieved
- Examples of file systems include:
  - NTFS (Windows)
  - FAT32 (USB drives, older Windows systems)
  - ext4 (Linux-based systems)

Linux Directory Structure

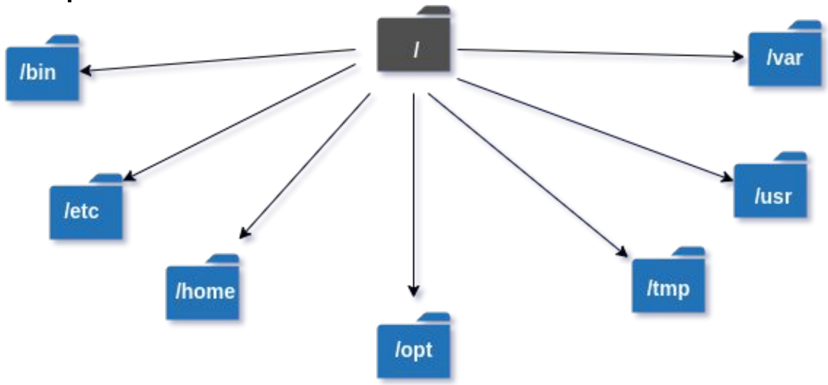
- In Linux/Unix operating system everything is a file - directories, files, and devices such as mouse, keyboard, and printer are files

- Types of files in Linux are:
  - General Files
    - It is also called ordinary files
    - It may be an image, video, program, or simple text file
    - They can be in ASCII or Binary format
    - It is the most commonly used file in the Linux system

Linux Directory Structure

- In Linux/Unix operating system everything is a file - directories, files, and devices such as mouse, keyboard, and printer are files

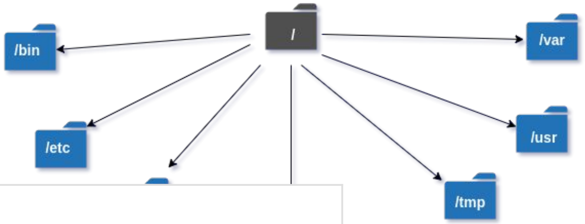
- Types of files in Linux are:
  - Directory Files – These types of files are a warehouse for other file types. It may be a directory file within a directory (subdirectory)
  - Device Files – In Linux devices such as CD-ROM, and hard drives are represented as files. For example, /dev/sda1, and /dev/sda2



Linux Directory Structure

- In Linux/Unix, operating system files are stored in a tree-like structure starting with the root directory
- The exact details of the directory structure may vary according the release of the OS

Linux Directory Structure



- The common top-level directories associated with the root directory are:

Directories	Description
/bin	binary or executable programs.
/etc	system configuration files.
/home	home directory. It is the default current directory.
/opt	optional or third-party software.
/tmp	temporary space, typically cleared on reboot.
/usr	User related programs.
/var	log files.

Linux Commands for files and directories - cd

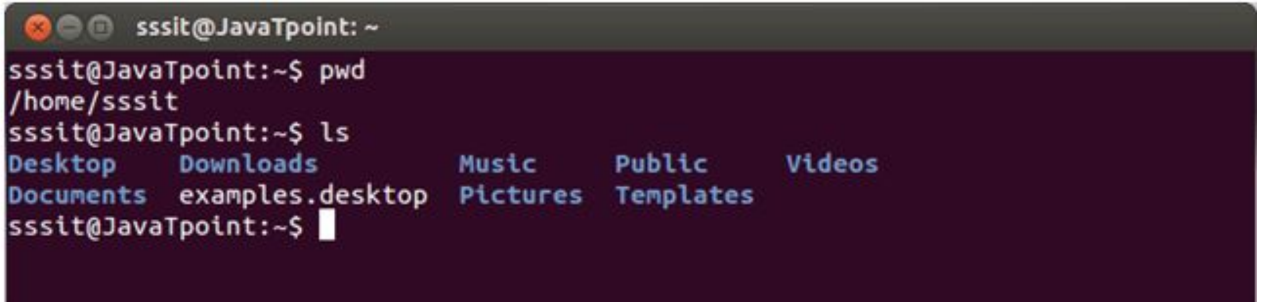
- The cd command is used to change the current working directory
  - cd [directory]
  - To change to a specific directory: cd /path/to/directory
  - To change to Home Directory: cd
  - To change to Parent Directory: cd ..
  - To change to a subfolder in the current directory: cd subdirectory

Linux Commands for files and directories cd

- The cd command is used to change the current working directory
  - cd [directory]
  - To change to a specific directory: cd /path/to/directory
  - To change to Home Directory: cd
  - To change to Parent Directory: cd ..
  - To change to the Previous Directory: cd -
  - To change to any folder: cd relative/path/to/directory
  - To change to a subfolder in the current directory: cd subdirectory

Linux Commands for files and directories: ls

- The ls command is used to list files and directories in a directory. It provides information about the files and directories, such as names, sizes, permissions, and timestamps.
  - To list files in the current directory: ls



```
sssit@JavaTpoint:~$ pwd
/home/sssit
sssit@JavaTpoint:~$ ls
Desktop    Downloads         Music     Public     Videos
Documents  examples.desktop  Pictures  Templates
sssit@JavaTpoint:~$
```

Linux Commands for files and directories: ls

- The ls command is used to list files and directories in a directory. It provides information about the files and directories, such as names, sizes, permissions, and timestamps.

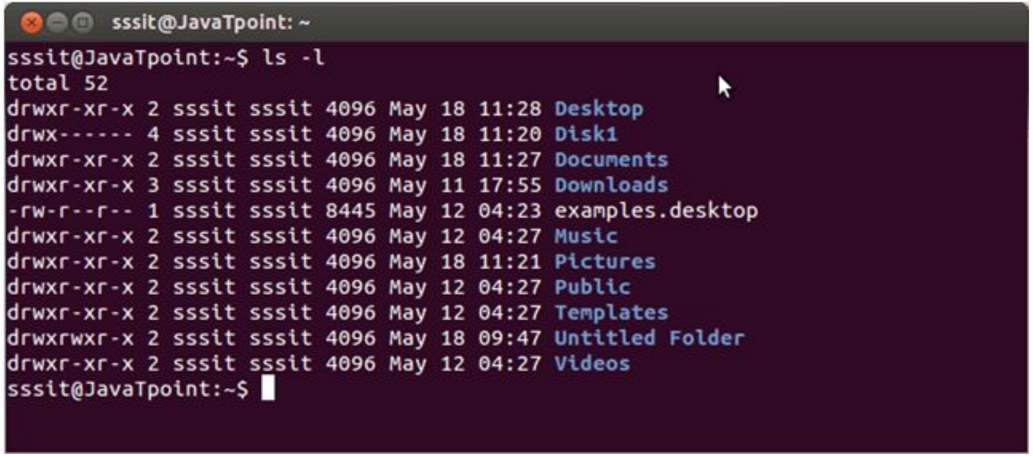
  - To list files with detailed information such as file permissions, number of links, owner, group, size, and modification time: ls -l

Columns in the output indicate specific things:

Column 1: File permission

Column 2: The number of links to the file

Column 3 & 4; The owner and group information



```
sssit@JavaTpoint:~$ ls -l
total 52
drwxr-xr-x 2 sssit sssit 4096 May 18 11:28 Desktop
drwx------ 4 sssit sssit 4096 May 18 11:20 Disk1
drwxr-xr-x 2 sssit sssit 4096 May 18 11:27 Documents
drwxr-xr-x 3 sssit sssit 4096 May 11 17:55 Downloads
-rw-r--r-- 1 sssit sssit 8445 May 12 04:23 examples.desktop
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Music
drwxr-xr-x 2 sssit sssit 4096 May 18 11:21 Pictures
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Public
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Templates
drwxrwxr-x 2 sssit sssit 4096 May 18 09:47 Untitled Folder
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Videos
sssit@JavaTpoint:~$
```

Linux Commands for files and directories: ls

- The ls command is used to list files and directories in a directory. It provides information about the files and directories, such as names, sizes, permissions, and timestamps.

    - To list files with detailed information such as file permissions, number of links, owner, group, size, and modification time: ls -l

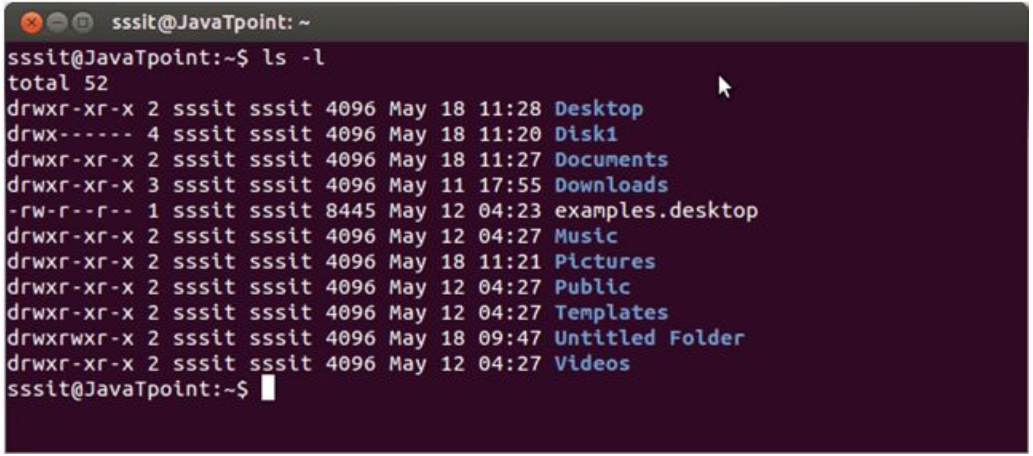
Columns in the output indicate specific things:

Column 5: Size of the file in bytes

Column 6: The date and time on

which the file was recently modified

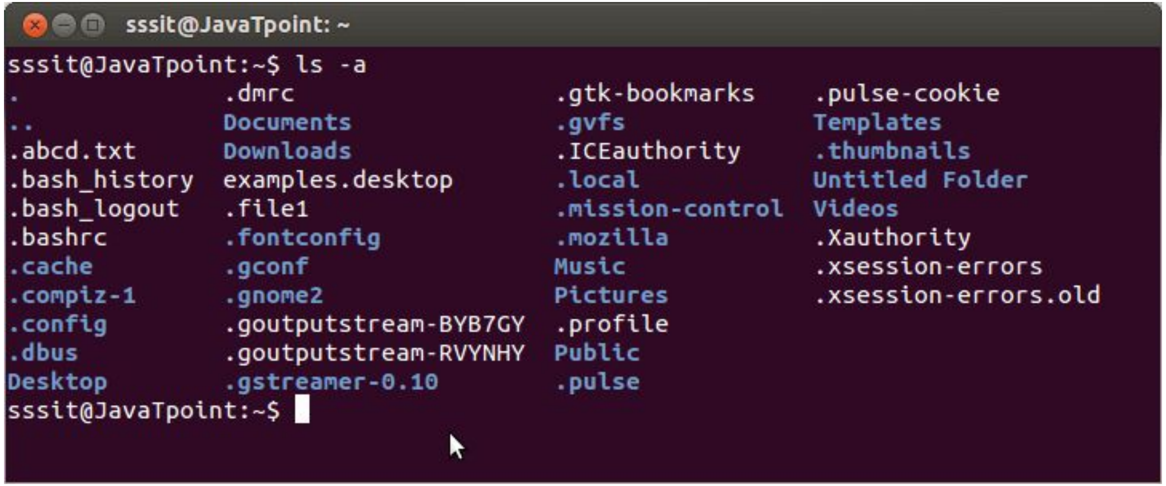
Column 7: File or directory name



```
sssit@JavaTpoint:~$ ls -l
total 52
drwxr-xr-x 2 sssit sssit 4096 May 18 11:28 Desktop
drwx------ 4 sssit sssit 4096 May 18 11:20 Disk1
drwxr-xr-x 2 sssit sssit 4096 May 18 11:27 Documents
drwxr-xr-x 3 sssit sssit 4096 May 11 17:55 Downloads
-rw-r--r-- 1 sssit sssit 8445 May 12 04:23 examples.desktop
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Music
drwxr-xr-x 2 sssit sssit 4096 May 18 11:21 Pictures
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Public
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Templates
drwxrwxr-x 2 sssit sssit 4096 May 18 09:47 Untitled Folder
drwxr-xr-x 2 sssit sssit 4096 May 12 04:27 Videos
sssit@JavaTpoint:~$ 
```

Linux Commands for files and directories: ls

- The ls command is used to list files and directories in a directory. It provides information about the files and directories, such as names, sizes, permissions, and timestamps.
    - To listing all Files (Including Hidden). Names of hidden files begin with a dot: ls -a



sssit@JavaTpoint: ~

```
sssit@JavaTpoint:~$ ls -a
.                .dmrc                    .gtk-bookmarks     .pulse-cookie
..               Documents                .gvfs              Templates
.abcd.txt        Downloads                .ICEauthority      .thumbnails
.bash_history    examples.desktop         .local             Untitled Folder
.bash_logout     .file1                   .mission-control   Videos
.bashrc          .fontconfig              .mozilla           .Xauthority
.cache           .gconf                   Music              .xsession-errors
.compiz-1        .gnome2                  Pictures           .xsession-errors.old
.config          .goutputstream-BYB7GY    .profile
.dbus            .goutputstream-RVYNHY    Public
Desktop          .gstreamer-0.10          .pulse
sssit@JavaTpoint:~$ 
```

Linux Commands for files and directories ls

- The ls command is used to list files and directories in a directory. It provides information about the files and directories, such as names, sizes, permissions, and timestamps.
    - To list all files (including hidden files) with detailed information: ls -al

Linux Commands for files and directories: ls

- The ls command is used to list files and directories in a directory. It provides information about the files and directories, such as names, sizes, permissions, and timestamps.
    - To list files with detailed information sorted based on the time of modification, with the newest files first: ls -lt

Linux Commands for files and directories: cp

- The cp command is used copy files or directories from one location to another
  - Copy a file from the current location to another directory: cp source-file.txt /path/to/destination/
  - Copy a multiple files from the current location to another directory: cp source-file.txt source-file2.txt source-file3.txt /path/to/destination/
  - Copy a directory and its contents in from the current location to another directory: cp -r source-directory/ /path/to/destination/
  - Copy a file from the current location to another directory, preserving its original file attributes, timestamps, and permissions: cp -a source-file.txt /path/to/destination/
  - Copy a file from the current location to another directory, confirming with the user whether to overwrite the target file or not: cp -i source-file.txt /path/to/destination/

Linux Commands for files and directories: cp

- The cp command is used copy files or directories from one location to another
  - Copy a file from the current location to another directory, forcing the system to overwrite overwrite the target file, if it exists: cp -f source-file.txt /path/to/destination/
  - Copy a file from the current location to another directory, displaying the progress of copy by listing the files being copied: cp -v source-file.txt /path/to/destination/
  - Copy a file from the current location to another directory. Copying will be done only if the source file is the latest: cp -u source-file.txt /path/to/destination/

Linux Commands for files and directories: rm

- The rm command is used to remove or delete files and directories
  - Remove a file residing in the current folder: rm file.txt
  - Remove many files residing in the current folder: rm file1.txt file2.txt
  - Remove a directory and its contents: rm -r directory-to-be-removed
  - Remove a file residing in the current folder without a confirmation: rm -f file.txt
  - Remove a file residing in the current folder with confirmation prompt:rm -i file.txt
  - Remove files residing in a directory whose names match a pattern (for eg. all files with their file name extensions as txt): rm /path/to/directory/*.txt

Linux Commands for files and directories: rmdir

- The rmdir command is specifically designed for deleting directories that contain no files or subdirectories
    - Remove an empty directory: rmdir empty_directory/
    - Remove many files empty directories: rmdir dir1/ dir2/ dir3/
    - Remove a parent directory if its become empty after removing a child directory under the parent: rmdir -p parent_directory/child_directory/
    - Remove the specified directory and its parent directory if they become empty after removing the specified directory: rmdir -r empty_directory/
    - Remove the files in verbose mode (which shows the directories as they are removed.): rmdir -v empty_directory/

Linux Commands for files and directories: mkdir

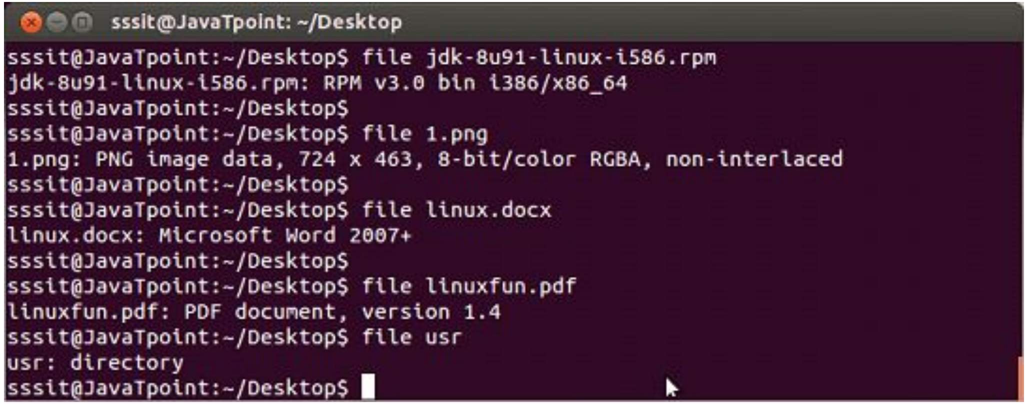
- The mkdir command is used to create directories
  - Create a single directory under the currently directory: mkdir directory_name
  - Create a multiple directories under the currently directory: mkdir dir1 dir2 dir3
  - Create a directory structure with a parent child relationship: mkdir -p parent_directory/child_directory
  - Create directories with specific permissions: mkdir -m 755 directory_name
  - Create directories with a verbose (output displaying a message for each directory created): mkdir -v dir1 dir2 dir3

Linux Commands for files and directories: pwd

- The pwd stands for print working directory
- It is used to display the current working directory, which is the directory you are currently located in within the file system
  - To display the current working directory: pwd

Linux Commands for files and directories: file

- The file is used to determine the type of a file or the nature of a file's content. It examines the file's header, a specific portion of the file, to identify its format
  - To check the type of file: file filename
  - To check the type of multiple files: file filename1 filename2
  - To get the MIME type information in addition to the file type: file -i filename
  - To check the types of files in a directory: file /path/to/directory/*
  - To get the verbose version of file type: file -v /path/to/directory/*



```
sssit@JavaTpoint:~/Desktop$ file jdk-8u91-linux-i586.rpm
jdk-8u91-linux-i586.rpm: RPM v3.0 bin i386/x86_64
sssit@JavaTpoint:~/Desktop$
sssit@JavaTpoint:~/Desktop$ file 1.png
1.png: PNG image data, 724 x 463, 8-bit/color RGBA, non-interlaced
sssit@JavaTpoint:~/Desktop$
sssit@JavaTpoint:~/Desktop$ file linux.docx
linux.docx: Microsoft Word 2007+
sssit@JavaTpoint:~/Desktop$
sssit@JavaTpoint:~/Desktop$ file linuxfun.pdf
linuxfun.pdf: PDF document, version 1.4
sssit@JavaTpoint:~/Desktop$ file usr
usr: directory
sssit@JavaTpoint:~/Desktop$
```

Linux Commands for files and directories; cat

- The cat command is used to concatenate and display the content of one or more files
    - Display the content of a file: cat filename
    - Concatenate multiple files and display as a single file: cat file1 file2 file3
    - Redirect output to a new file: cat file1 > newfile (Writes contents of file1 to newfile. If the target file exists, it will be overwritten)
    - Append contents to an existing file:cat file2 >> targetfile
    - Display line numbers along with the content: cat -n filename
    - Create a new file accepting input from the user: cat > newfile (Press CTRL + D to finish off the data entry and save the contents to the file)
    - Combine files with line numbers: cat -n file1 file2 > combinedfile (To combine the contents of file1 and file2 adding line numbers, and to write the output to the combinedfile)

Common administrative tasks: Obtaining Supervisor Privileges

- Supervisor privileges often referred to as "root" or "superuser" privileges, can be obtained using the sudo command
- The sudo command allows a permitted user to execute a command as the superuser or another user, as specified by the security policy configured in the /etc/sudoers file
  - To execute a command with superuser privileges, the command should be prefixed by sudo command
    - sudo apt update
  - The user will be prompted to enter the user password. This is the password associated with your user account
  - To execute a command as another user: sudo -u username command
  - The users who can vail sudo command is configured in the /etc/sudoers file. We can edit this file using the visudo command, which is designed to prevent syntax errors:
    - sudo visudo

Common administrative tasks: Obtaining Supervisor Privileges

- Supervisor privileges can also be attained using su command
- The su (substitute user) command is used to switch to another user account, including the superuser (root) account, to execute commands with the privileges of that user
- To switch to the Root User: su (System will ask for the root password, once we execute the command)
- To execute a specific command with root privileges: su -c 'command'
  - su -c 'ls /root'
- To switch to another user: su user-name
- To return to the original user account: exit
- When using su to switch to the root user, user need to know the root password
- su command does not require configuration in the /etc/sudoers file like the sudo command

User Management in Linux

- Each user is an entity with a ID allotted by the system
- The ID 0 is assigned to the root user and the IDs 1 to 999 (both inclusive) are assigned to the system users and the ids for local user begins from 1000 onwards
- Information about the users in the system is kept in /etc/passwd
- To get the ID assigned to a user, use id <user name>

User Management - users groups

- A user belong to a group
- A group is created to represent a specific class of users with certain rights and privileges
- Later any user can be added to an existing group - all the rights and privileges owned by the group will be enjoyed by the user belongs to that group too!
- Information about user groups are maintained in /etc/group and /etc/gshadow files on your system
- To add a user group, use groupadd command
  - groupadd <name of the group>
- To add a user group with a specific ID
  - groupadd <name of the group> -g <group ID>

User Management - users groups - adding members

- To get a list of users that belongs to a particular group,
    - getent group <group name>
- To add an existing user to a group
    - usermod -a -G <group name> <user name>
- To add an existing user to multiple groups
    - usermod -a -G <group name1>,<group name2>,<group name3> <user name>
- To add a new user to a group
    - useradd -G <group name> <user>
- To add a new user to multiple groups
    - useradd -G <group name1>,<group name2>,<group name3> <user>

User Management - users groups - deleting members

- To delete a user from a group
  - usermod -G <group name> <user name>

User Management - users groups - Deleting Groups

- To delete a group
  - groupdel <group name>
- This command will not affect the members of the group

User Management - Creating Users

- To create a user
  - useradd <user name>
- When a new user is created,, by default a new home directory is made for the user. By default, the directory name is the username of the new user. If you want your user to have a home directory with some other name, you can use the -d flag in the useradd command
  - useradd -d <home/path-to-the-folder> <user name>
- If the home directory for the user is not required at the time of creation, it can be done as follows
  - useradd -M <user name>

User Management - Deleting Users

- To delete a user
  - userdel <user name>
- To delete a user along with its home folder
  - userdel -r <user name>

User Management - Listing users that are logged in

- users
  - Displays only the usernames of logged-in users
- whoami
  - Displays the username of the current user
- who
  - This displays a list of users currently logged in, along with details like terminal session and login time
- w
  - Shows more details, including what each user is doing
- last
  - Shows login history of users

man - Ubuntu Manuals!

- Man pages (short for manual pages) are built-in help documents in Linux, including Ubuntu
- They provide detailed information about commands, configuration files, and system calls
- To view the user manual for a command, we have to use the man command

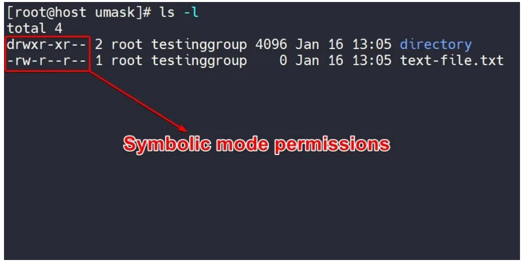
  man <command>

  Eg:- man ls

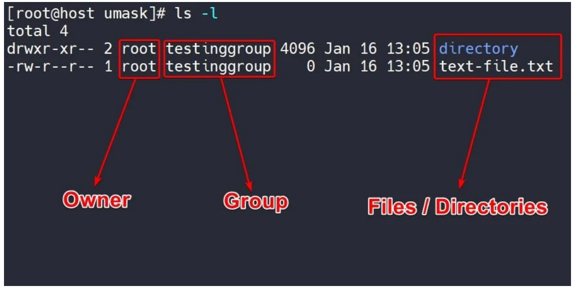
- A typical man page contains:
  - NAME – The command name and a short description
  - SYNOPSIS – The command's syntax and usage
  - DESCRIPTION – A detailed explanation of what the command does
  - OPTIONS – The available options and flags for the command
  - EXAMPLES – How to use the command in practice
  - SEE ALSO – Related commands or documentation

umask

- umask (short for user file-creation mode mask) is used by Linux systems to set default permissions for newly created files and directories
- This command is very useful when there are multiple users that are creating new files and directories, particularly in a shared environment
- System administrators can ensure that with properly set umask values, different users will make files with secure permissions by default



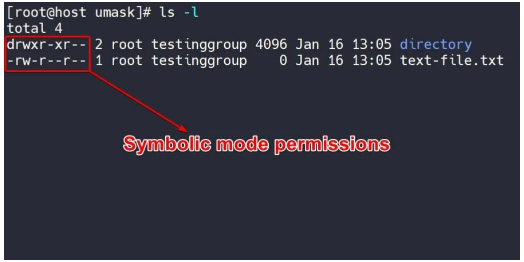
```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```



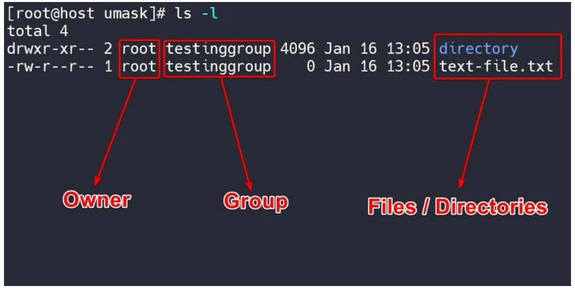
```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```

User Permissions

- Linux based operating systems have a set of properties that are used to define who is allowed to read, write, or execute specific files or directories
- There are three categories called permissions classes to which these permissions apply:
  - User: by default is the owner or creator of a file or folder. The ownership of the new file defaults to this user
  - Group: the set of users that share the same access level or permissions to a file or folder.
  - Other: defined as any user not included in the previous two categories. These users have not created a file or folder, nor do they belong to a specific user group. This group includes everyone not identified as a user or as being part of a user group



```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```



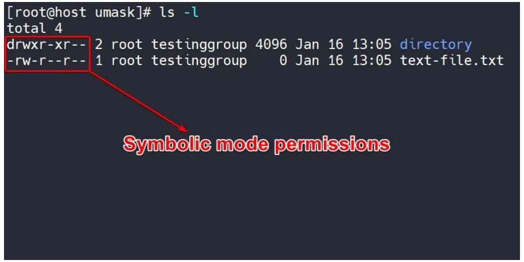
```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```

User Permissions

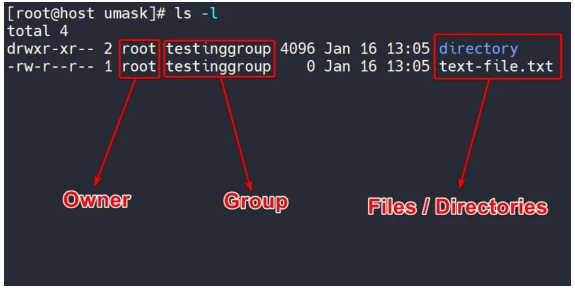
- Each permission has a numeric value:
  - Read (r) = 4
  - Write (w) = 2
  - Execute (x) = 1

Examples

- 644 - Owner can read/write, group & others can read
- 755 - Owner can read/write/execute, others can read/execute
- 777 - Everyone can read/write/execute (not recommended)



```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```



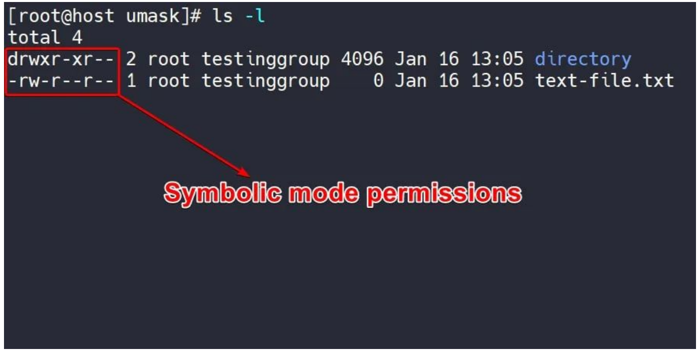
```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```

Setting User Permissions - chmod

- chmod command is used to set user permissions for a file

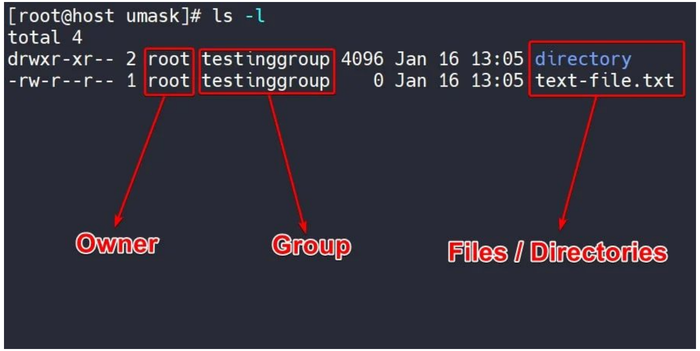
Examples

- chmod 644 filename - Owner can read/write, group & others can read
- chmod 755 filename - Owner can read/write/execute, others can read/execute
- chmod 777 filename - Everyone can read/write/execute (not recommended)



```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```

Symbolic mode permissions



```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```

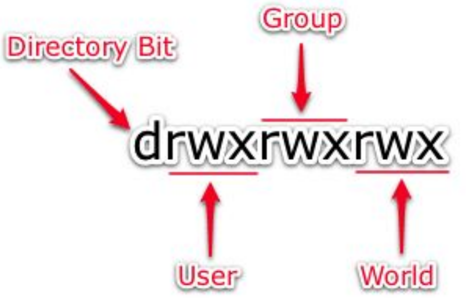
Owner

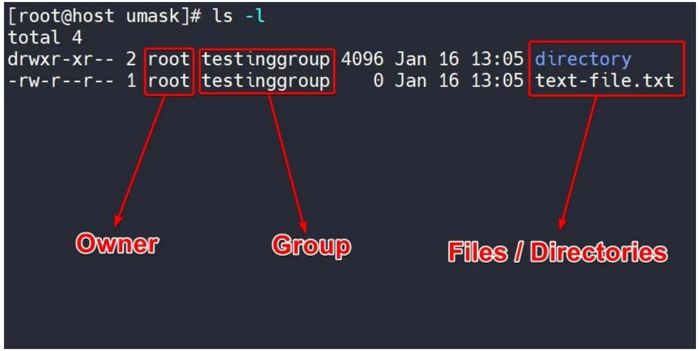
Group

Files / Directories

umask

- Linux based operating systems have a set of properties that are used to define who is allowed to read, write, or execute specific files or directories
- There are three categories called permissions classes to which these permissions apply:
  - User: by default is the owner or creator of a file or folder. The ownership of the new file defaults to this user
  - Group: the set of users that share the same access level or permissions to a file or folder.
  - Other: defined as any user not included in the previous two categories. These users have not created a file or folder, nor do they belong to a specific user group. This group includes everyone not identified as a user or as being part of a user group





```
[root@host umask]# ls -l
total 4
drwxr-xr-- 2 root testinggroup 4096 Jan 16 13:05 directory
-rw-r--r-- 1 root testinggroup    0 Jan 16 13:05 text-file.txt
```

umask

- The default umask value is currently set to the octal value of 022
- To view the default value umask, we have to type in umask on the terminal
- If the umask is set to 022, the permissions for files and directories are made as follows:
  - Subtract that value 022 from the default permissions Linux sets for files and directories
    - It is 666 for files and 777 for directories
  - Hence the new files and directories will have their access permissions as follows
    - New files: 666 - 022 = 644
    - New directories: 777 - 022 = 755

======