CS-22 : Operating System Concept with Unix/Linux

Prepared By : Lathiya_Harshal.

(1)(A)

(1) Define Round Robin:

Round Robin (RR) is a **CPU scheduling algorithm** where each process is assigned a fixed time slot (also called **time quantum**) in a cyclic order. If a process doesn't finish in its time slice, it is moved to the back of the queue.
👉 *It is simple, fair, and suitable for time-sharing systems.*

(2) Give any four names of OS types:

1. **Batch Operating System**
2. **Time-Sharing Operating System**
3. **Real-Time Operating System (RTOS)**
4. **Distributed Operating System**

(3) Define Virtual Memory:

Virtual Memory is a **memory management technique** that allows the execution of processes that may not be completely in the main memory. It uses **disk space as an extension of RAM**, enabling more programs to run simultaneously by **swapping** data between RAM and disk.
👉 *It gives an illusion of a large main memory.*

(4) Define SJN:

SJN (Shortest Job Next), also known as **SJF (Shortest Job First)**, is a **non-preemptive CPU scheduling algorithm** in which the **process with the shortest execution time is selected next**.
👉 *It minimizes average waiting time but may cause starvation for longer processes.*

(B)

(1) Define User Point of View Operating System:

From the **user's point of view**, an operating system is a **convenient interface** that allows interaction with the computer hardware without needing to know the details of how the hardware works.
👉 The OS provides a **friendly environment** to run applications, manage files, and execute commands efficiently and safely.

Example: In Windows or Linux, the user can open programs, manage files, and connect to the internet using a graphical or command-line interface — all controlled by the OS.

(2) Define Feature Point of View Operating System:

From the **feature point of view**, an operating system is a **collection of software routines** that manage system resources like **CPU, memory, I/O devices, and files**.
👉 It ensures **multiprogramming**, **multitasking**, **security**, **process scheduling**, and **hardware abstraction** so that multiple applications and users can operate efficiently.

Key features include: Process management, memory management, file systems, device drivers, and user interfaces.

(C)

(1) Explain Virtual Memory with Segmentation

🔹 Definition of Virtual Memory:

Virtual memory is a **memory management technique** that allows execution of processes **without requiring the entire process to be in the main memory (RAM)** at once. It creates an **illusion of a large continuous memory** using a combination of **RAM and disk space**.

🔹 Why Virtual Memory?

- To **run large applications** that do not fit into RAM.
- To **allow multitasking** by giving each process its own virtual address space.
- To **optimize RAM usage** by loading only needed parts of a process.

🔹 Segmentation in Virtual Memory:

Segmentation is a technique where a process is divided into **logical units** called **segments**:

- Each segment represents a logical block (e.g., code, data, stack).
- Each segment has its own **base address and length**.
- Segments can be of **variable size**, unlike pages.

When using segmentation for virtual memory:

- **Logical address = Segment number + Offset**
- The OS maintains a **Segment Table** for each process containing:
    - **Base**: Starting physical address of the segment
    - **Limit**: Length/size of the segment

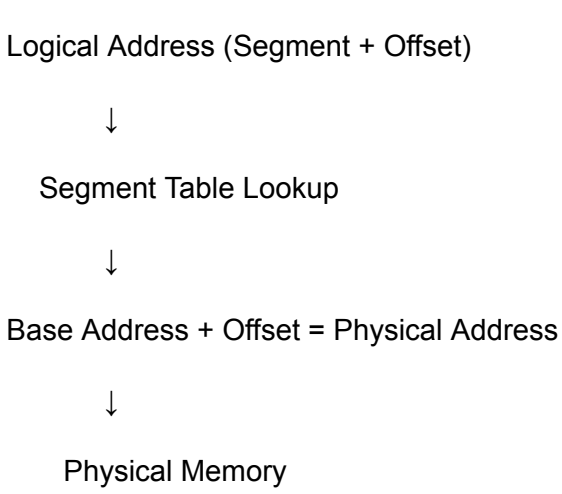
🔹 How Virtual Memory Works with Segmentation:

1. The **CPU generates logical addresses**.
2. The **MMU (Memory Management Unit)** uses the **segment table** to translate logical addresses to physical addresses.
3. If the segment is not present in RAM, a **page fault** occurs, and the **OS loads the segment from disk** (swap space) into RAM.

🔹 Advantages:

- Supports **modularity** and logical separation.
- Protects memory by restricting access to segments.
- Efficient use of memory.

- **Diagram:**



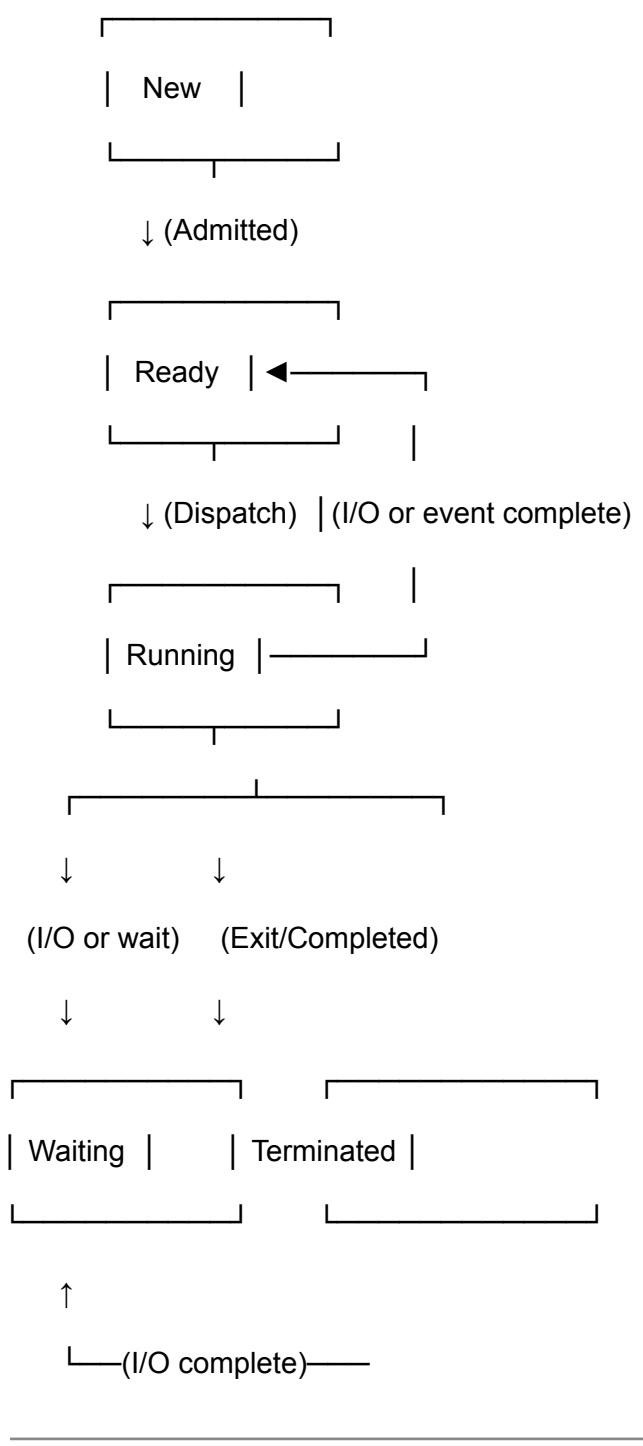
---

(2) Draw Process State Transition Diagram

- **Definition:**

A process during its lifetime goes through **various states**, such as new, ready, running, waiting, and terminated. The **Process State Transition Diagram** shows how a process **moves between these states** based on events like scheduling, I/O, or completion.

- **Diagram:**



🔹 States Explained:

- **New**: Process is being created.
- **Ready**: Process is ready to run, waiting for CPU.
- **Running**: CPU is executing the process.
- **Waiting**: Process is waiting for I/O or event.
- **Terminated**: Process has finished execution.

🔹 Transitions:

- **Admitted**: OS moves a new process to ready.
- **Dispatch**: CPU scheduler selects a ready process to run.
- **I/O or Event Wait**: Running process needs to wait, so it moves to waiting.
- **I/O Complete**: Waiting process is ready again.
- **Exit**: Process finishes and goes to terminated.

✅ Conclusion:

The Process State Transition Diagram helps in understanding **how the OS manages processes** and **ensures smooth execution** by scheduling and transitioning based on events and resource availability.

(D)

✅ (1) Explain Virtual Memory & Physical Memory

🔹 Physical Memory:

- **Definition**: Physical memory refers to the **actual RAM (Random Access Memory)** installed in a computer system.
- It is **limited in size** and is directly accessible by the CPU.
- Stores **currently executing programs**, their data, and the operating system kernel.
- Once RAM is full, new data cannot be stored unless some memory is freed or **virtual memory** is used.

Example: If your computer has 8 GB RAM, that's the physical memory.

🔹 Virtual Memory:

- **Definition**: Virtual memory is a **memory management technique** that uses **disk space** (usually called the **swap file** or **page file**) to **extend RAM**.
- It gives the illusion of having **more memory than physically available**.
- Managed by the **Operating System**, it loads only the required parts of a program into RAM.
- It uses techniques like **paging** and **segmentation** to allocate memory.

Example: If an application needs 12 GB memory but only 8 GB RAM is available, 4 GB can be handled via virtual memory on disk.

🔹 Comparison Table:

Feature	Physical Memory	Virtual Memory
Location	RAM (Hardware)	Hard Disk (Software-managed)
Speed	Very fast	Slower (due to disk access)
Size	Limited to installed RAM	Can be much larger
Access	Direct by CPU	Indirect (needs translation)
Managed By	Hardware & OS	OS using page tables & memory mapping

---

✅ Conclusion:

Physical memory is essential for performance, while **virtual memory** improves flexibility and allows running large or multiple programs even with limited RAM.

✅ (2) Explain Functions of Operating System

(5 Marks)

An **Operating System (OS)** acts as an **interface between the user and hardware** and provides an environment for executing programs. Its **main functions** include:

🔹 1. Process Management

- Manages processes in the system: creation, execution, and termination.
- Handles **process scheduling** using algorithms (FCFS, SJF, RR, etc.).
- Manages **context switching** and **inter-process communication (IPC)**.

🔹 2. Memory Management

- Allocates and deallocates memory space to programs in use.
- Maintains memory **maps** and tables.
- Handles **paging**, **segmentation**, and **virtual memory**.

🔹 3. File System Management

- Organizes files in directories.
- Manages file creation, deletion, reading, writing, and permissions.
- Ensures **data security and consistency**.

🔹 4. Device Management

- Manages all **I/O devices** (keyboard, printer, disk, etc.).
- Uses device drivers for hardware interaction.
- Performs **buffering**, **spooling**, and **interrupt handling**.

🔹 5. Security and Protection

- Protects system resources from unauthorized access.
- Provides **authentication**, **authorization**, and **encryption**.
- Isolates processes to prevent interference or memory corruption.

🔹 6. User Interface Management

- Provides **CLI (Command-Line Interface)** or **GUI (Graphical User Interface)**.
- Allows users to interact with the system easily.

🔹 7. Networking and Communication

- Supports protocols and APIs for network communication.
- Manages **networked processes**, sharing, and data transfer.

---

2 (A)

(1) List out types of files in Unix:

Unix supports the following **types of files**:

1. **Regular Files** – Contain text, data, or program instructions
2. **Directory Files** – Contain names of other files and directories
3. **Character Device Files** – For devices like keyboards, mice
4. **Block Device Files** – For devices like hard disks, CD-ROMs
5. **FIFO (Named Pipes)** – For inter-process communication
6. **Socket Files** – For network communication
7. **Symbolic Links** – References to other files

---

(2) Use of `mkdir` Command:

The `mkdir` (make directory) command is used to **create a new directory** in Unix/Linux.

📌 **Syntax:**

mkdir [directory_name]

📌 **Example:**

mkdir myfolder

This will create a directory named `myfolder` in the current location.

---

(3) Use of `pwd` Command:

The `pwd` (print working directory) command is used to **display the full absolute path** of the **current working directory**.

📌 **Syntax:**

pwd

📌 **Example Output:**

/home/mihir/documents

---

(4) Use of `chgrp` Command:

The `chgrp` (change group) command is used to **change the group ownership** of a file or directory.

📌 **Syntax:**

chgrp [group_name] [file_name]

📌 **Example:**

chgrp developers project.txt

This will change the group of `project.txt` to `developers`.

🔹 (B) Answer in Brief (2 marks each)

(1) Explain Types of Shell:

A **shell** is a command-line interpreter that allows users to interact with the operating system. Common **types of shells** in Unix/Linux include:

1. **Bourne Shell (sh)** – Original Unix shell.
2. **Bash Shell (bash)** – Enhanced Bourne Again SHell; default in most Linux systems.
3. **C Shell (csh)** – Syntax similar to C programming.
4. **Korn Shell (ksh)** – Combines features of sh and csh.
5. **Z Shell (zsh)** – Advanced features like spell check, auto-complete.

---

(2) Explain VI Modes:

`vi` is a powerful text editor in Unix with **three modes**:

1. **Command Mode** – Default mode; for navigating and editing (delete, copy, paste).
2. **Insert Mode** – For typing and editing text. Activated by pressing `i`, `a`, or `o`.
3. **Last Line Mode (Ex Mode)** – For saving, quitting, and searching (activated by `:`).

🔹 (C) Answer in Brief (3 marks each)

(1) Explain Types of Files in Unix:

Unix supports multiple types of files:

1. **Regular Files** – Contain data, text, or executable code.
2. **Directory Files** – Contain a list of files and directories.
3. **Character Device Files** – Represent character-based devices (e.g., keyboard).
4. **Block Device Files** – Represent block storage (e.g., hard disks).
5. **FIFO Files (Named Pipes)** – For inter-process communication.
6. **Socket Files** – For network communication.
7. **Symbolic Links** – Pointers to other files (like shortcuts).

---

(2) Explain Any Three Features of Unix:

1. **Multitasking** – Can run multiple tasks simultaneously.
2. **Multiuser Capability** – Many users can access the system at the same time.
3. **Portability** – Easily runs on different hardware platforms.
4. **Security** – Offers file permissions and user/group access control.
5. **Hierarchical File System** – Organizes files in a tree-like structure.

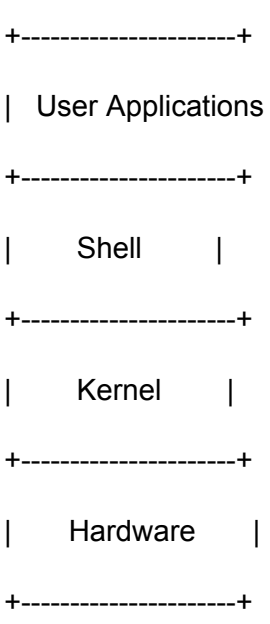
🔹 (D) Write a Note on (5 marks each)

(1) Explain Unix Architecture:

Unix Architecture is composed of the following layers:

1. **Hardware** – Physical devices like CPU, RAM, I/O devices.
2. **Kernel** – Core part of Unix; manages hardware, memory, processes, and file system.
3. **Shell** – Interface between user and kernel; interprets commands.
4. **Utilities/Commands** – Tools and programs used for operations (e.g., ls, cp).
5. **Applications** – User-developed or system-provided apps.

Diagram:



(2) Explain Any 5 File & Directory Commands with Example:

1. **`ls`** – Lists contents of a directory
   👉 `ls -l` (long format)

2. **`cd`** – Changes current working directory
   👉 `cd /home/user`

3. **`mkdir`** – Creates a new directory
   👉 `mkdir myfolder`

4. **`rm`** – Removes files or directories
   👉 `rm file.txt` or `rm -r myfolder`

5. **`touch`** – Creates a new empty file
   👉 `touch notes.txt`

6. **`pwd`** – Shows present working directory
   👉 `pwd`

---

(3)(A)

(1) Define Telnet:

Telnet is a **network protocol** used to establish a **remote connection** to another computer over a TCP/IP network.
👉 It allows users to **log into remote systems** and access them as if they were local, typically using command-line interface.

(2) Define VI Editor:

`vi` (Visual Editor) is a **text editor in Unix/Linux** used for creating and editing files.
👉 It operates in **three modes**: Command mode, Insert mode, and Last line mode, and is known for its **efficiency and keyboard-based editing**.

(3) Use of `touch` Command:

The `touch` command is used to:

- **Create an empty file**, or
- **Update the access and modification timestamp** of an existing file.

📌 **Example:**

touch file.txt

(4) Define Mounting File System:

Mounting a file system means **attaching an additional file system** (like a USB drive or another partition) to a **specific directory** in the main file system.
👉 This makes the new file system **accessible under the existing directory structure**.

📌 **Example:**

mount /dev/sdb1 /mnt/usb

(B)

(1) Explain Any Five Shell Keywords:

Shell **keywords** are **reserved words** used in shell scripting. They have special meaning and cannot be used as variable names. Here are five commonly used keywords:

1. if

Used to start a conditional statement.
📌 *Example:* `if [ $a -gt $b ]`

2. then

Used with `if` to define the block of code to execute if the condition is true.
📌 *Example:*

```
if [ $a -gt $b ]

then

  echo "A is greater"

fi
```

3. else

Defines alternative action if `if` condition fails.
📌 *Example:* `else echo "B is greater"`

4. fi

Ends the `if` block (reverse of `if`).
📌 *Example:* Ends the above if-else structure.

5. while

Used to create a loop that runs as long as the condition is true.
📌 *Example:*

```
while [ $i -lt 5 ]

do

  echo $i

  i=$((i+1))

done
```

✅ (2) List Out Process Related Commands:

Here are **important Unix/Linux commands** related to **process management**:

1. `ps` – Displays current running processes.

📌 *Example:* `ps aux`

2. `top` – Real-time display of running processes and system resource usage.

📌 *Example:* `top`

3. `kill` – Sends signal to terminate a process.

📌 *Example:* `kill 1234` (kills process with PID 1234)

4. `nice` – Starts a process with a given priority.

📌 *Example:* `nice -n 10 myscript.sh`

5. `bg` / `fg` – Moves a process to background/foreground.

📌 *Example:* `fg` brings background job to foreground.

(C)

(1) Mounting vs Demounting File System

🔹 Mounting File System:

- **Definition**: Mounting is the process of **making a file system accessible** by attaching it to a directory (mount point) in the existing file system hierarchy.
- After mounting, users can **access the contents** of the device (e.g., USB, CD-ROM, other partitions).
- The `mount` command is used.

📌 **Example**:

mount /dev/sdb1 /mnt/usb

This mounts the device `/dev/sdb1` to the `/mnt/usb` directory.

🔹 Demounting File System:

- **Definition**: Demounting (unmounting) is the process of **safely removing a mounted file system**, ensuring all data is written back and the device can be removed.
- Prevents **data corruption or loss**.
- The `umount` command is used.

📌 **Example**:

umount /mnt/usb

🔸 Difference Table:

Feature	Mounting	Demounting
Purpose	Attaches file system	Detaches file system
Command Used	`mount`	`umount`
Access	Makes files accessible	Stops access to files
When Required	Before using a device	Before removing a device

---

(2) Explain Screen Control Command

🔹 What is a Screen in Unix/Linux?

The `screen` **command** is a terminal multiplexer. It allows you to:

- Run multiple shell sessions in one terminal window.
- **Detach** from a session and **resume it later**, even after logout.
- Useful for running long background tasks.

🔹 Common Screen Control Commands:

Start a new screen session:

```
screen
```

Detach from screen session (keep it running in background):

```
Ctrl + A, then press D
```

List all screen sessions:

```
screen -ls
```

Reattach to a session:

```
screen -r [session_id]
```

Exit screen session:
Type `exit` inside the screen or press `Ctrl + D`.

🔸 Use Case Example:

You start a long-running Python script using:

screen

python myscript.py

Then detach (`Ctrl+A, D`) and log out. Later, reattach using:

screen -r

(D)

(1) Explain Conditional Statements with Example: `if`, `then`, `elif`, `else`, `fi`

🔹 What are Conditional Statements in Shell?

Conditional statements are used to **make decisions** in a shell script based on certain conditions. The `if-elif-else-fi` structure allows branching logic based on multiple conditions.

🔹 Syntax:

```
if [ condition1 ]
then
   # Code block if condition1 is true
elif [ condition2 ]
then
   # Code block if condition2 is true
else
   # Code block if all above conditions are false
fi
```

🔹 Example:

```
#!/bin/bash
echo "Enter your marks:"
read marks
if [ $marks -ge 90 ]
then
  echo "Grade: A+"
elif [ $marks -ge 75 ]
then
  echo "Grade: A"
elif [ $marks -ge 60 ]
then
  echo "Grade: B"
else
  echo "Grade: C or Fail"
fi
```

🔹 Explanation:

- The script asks for marks.
- Based on the value, it checks multiple conditions using `if`, `elif`, and `else`.
- The block ends with `fi`.

✅ Conclusion:

`if-elif-else` is useful for checking **multiple conditions sequentially**, and `fi` is used to end the structure.

✅ (2) Write a Program to Take Three Values from the User and Calculate Their Addition

🔹 Shell Script:

```
#!/bin/bash


echo "Enter first number:"
read num1
echo "Enter second number:"
read num2
echo "Enter third number:"
read num3
# Addition
sum=$((num1 + num2 + num3))
echo "The sum of $num1, $num2, and $num3 is: $sum"
```

🔹 Explanation:

- The script prompts the user to enter 3 values.
- `read` is used to take input from the user.
- `$((...))` is used to perform arithmetic addition.
- The final result is printed using `echo`.

🔸 Sample Output:

```
Enter first number:
10
Enter second number:
20
Enter third number:
30
The sum of 10, 20, and 30 is: 60
```

---

(4)(A)

1) Full Form of GNU:

👉 **GNU** stands for **"GNU's Not Unix"**
It is a **free and open-source operating system** developed as an alternative to Unix.

(2) Full Form of GPL:

👉 **GPL** stands for **"General Public License"**
It is a **free software license** that allows users to run, study, modify, and share software.

(3) Where is X-Config File?

👉 The **X-configuration file** (used for configuring the X Window System) is typically located at:
📁 `/etc/X11/xorg.conf`

> Note: In modern systems, it may be auto-generated and not manually edited.

(4) What is Window Manager in Linux?

👉 A **Window Manager** in Linux is a system software that **controls the placement and appearance of windows** within a GUI.
It handles:

- Opening/closing/moving windows
- Window borders and title bars
- Desktop themes and effects

Examples: `Metacity`, `Openbox`, `Fluxbox`, `i3`, `Xfwm`.

(B)

✅(1) Explain Freeware:

Freeware is software that is **available for use at no cost**, but its **source code is not available** for modification or redistribution.
It is **proprietary software** offered for free, usually by the developer or company.

📌 **Examples**: Adobe Acrobat Reader, Skype, WinRAR (free version)

- 🔹 Difference from open source: Freeware is free to use, but **not free to modify or share**.

✅ (2) Explain Two Features of Linux:

1. Multitasking:

Linux allows **multiple programs** to run at the same time efficiently. For example, you can play music, browse the web, and download files simultaneously.

2. Multiuser Support:

Multiple users can log in and use a Linux system **at the same time** without interfering with each other's work or data.

> Other notable features include **security, stability, portability, and open-source nature.**

(C)

✅ (1) Define GRUB (GRand Unified Bootloader):

GRUB stands for GRand Unified Bootloader.
It is the default bootloader used in most Linux distributions to load and manage multiple operating systems.

🔹 Key Points:

- **GRUB allows users to choose between multiple installed OSes (e.g., Linux and Windows).**
- **It loads the Linux kernel and passes control to it.**
- **It supports command-line mode and graphical menus for boot options.**

🔹 GRUB Configuration File:

- **Located at: `/boot/grub/grub.cfg` (or `/etc/default/grub` for settings)**
- **Automatically generated using the `update-grub` command.**

📌 **Example Entry in GRUB Menu:**

```
menuentry 'Ubuntu' {
  set root=(hd0,1)
  linux /vmlinuz root=/dev/sda1 ro quiet splash
  initrd /initrd.img
}
```

✅ (2) Define LILO Configuration (Linux Loader):

LILO stands for Linux Loader.
It is an older bootloader used in Linux systems to load the operating system into memory.

🔹 Key Points:

- **LILO was widely used before GRUB became standard.**
- **It does not support dynamic configuration; any changes require reinstallation using `lilo` command.**
- **Suitable for systems with only Linux or limited dual-boot setups.**

🔹 LILO Configuration File:

- **File: `/etc/lilo.conf`**

After editing the file, run:

```
sudo lilo
```

🔹 Sample Configuration:

```
boot=/dev/sda
prompt
timeout=50
default=linux
image=/boot/vmlinuz
  label=linux
  root=/dev/sda1
  read-only
```

🟢 **Note: GRUB is more flexible and modern, while LILO is mostly obsolete but still found in legacy systems.**

(D)

✅ (1) Difference between CGI and GUI

Feature	CGI (Common Gateway Interface)	GUI (Graphical User Interface)
Full Form	Common Gateway Interface	Graphical User Interface
Purpose	Enables web servers to run scripts and generate dynamic web content	Provides a visual interface for user interaction with software
User Interaction	No direct user interface; runs on the server side	Users interact using windows, icons, menus, and buttons
Technology Used	Server-side scripting (Perl, Python, Shell, etc.)	Desktop environments (GNOME, KDE) or graphical applications
Execution Environment	Web server (like Apache, Nginx)	Local computer or operating system
Example	A script that processes a web form and sends an email	File Explorer, Calculator, or Web Browser in Linux

✅ Summary:

- **CGI is used in web development for server-side communication.**
- **GUI is used on desktops for a user-friendly software experience.**

✅ (2) Explain History of Linux

🔹 Origins:

- **Linux was created by Linus Torvalds, a Finnish student, in 1991.**
- **Inspired by MINIX (a small Unix-like system used for teaching), Linus wanted to develop a free and open-source operating system kernel.**

🔹 **Key Milestones:**

1. **1991 – Linus posted the first Linux kernel (version 0.01) on the internet.**
2. **1992 – Linux was released under the GNU General Public License (GPL), allowing free use and modification.**
3. **1993–1994 – First Linux distributions (Slackware, Debian, Red Hat) were released.**
4. **2000s–Present – Linux becomes the backbone of servers, cloud computing, Android, and embedded systems.**

🔹 **Why Linux is Popular:**

- **It is free, open-source, stable, and secure.**
- **Backed by large communities and organizations (Red Hat, Canonical, IBM).**
- **Used in servers, supercomputers, Android phones, and IoT devices.**

✅ **Summary:**

Linux evolved from a personal project to one of the most widely used and influential operating systems in the world, known for its flexibility, speed, and open nature.

---

(5)

(A) Objective Type Questions (1 mark each)

(1) Full Form of GNOME:

👉 **GNOME stands for GNU Network Object Model Environment**
It's a popular desktop environment for Unix/Linux systems.

---

(2) Define FTP:

👉 **FTP (File Transfer Protocol) is a standard network protocol used to transfer files between a client and a server over a TCP/IP-based network.**

---

(3) Define WINE:

👉 **WINE stands for Wine Is Not an Emulator.**
It is a compatibility layer that allows running Windows applications on Linux/Unix systems.

---

(4) Define Firewall:

👉 **A firewall is a security system that monitors and controls incoming and outgoing network traffic based on predefined security rules.**
It helps to block unauthorized access while permitting legitimate communication.

(B)

(1) Explain KDE Control Panel:

The KDE Control Panel (also known as System Settings) is the central configuration hub for the KDE desktop environment.
It allows users to manage:

- **Appearance (themes, icons, fonts)**
- **Hardware (printers, display, input devices)**
- **Network settings**
- **User management**
  It is graphical and user-friendly, often launched with the command `systemsettings`.

---

(2) Explain GNOME Control Panel:

The GNOME Control Panel (or Settings) is a central interface in GNOME desktop for configuring the system.
It includes settings for:

- **Display, Keyboard, Mouse**
- **User Accounts**
- **Privacy and Security**
- **Network and Bluetooth**

Accessible via the "Settings" menu in GNOME, it aims for simplicity and ease-of-use.

(C)

(1) Explain Installation & Configuration of WINE:

Installation:

On Debian/Ubuntu-based systems:

sudo apt update

sudo apt install wine64

Configuration:

- **Run `winecfg` to create a Wine prefix (acts like a virtual Windows environment).**
- **Configure Windows version, graphics, audio, etc.**

Running a Windows Program:

wine setup.exe

WINE helps in running .exe files and supports many common Windows applications.

---

(2) Explain DNS Services:

DNS (Domain Name System) is a naming system that translates domain names into IP addresses.

Function:

- **Converts URLs like `www.example.com` → `93.184.216.34`**
- **Works through a hierarchy: Root DNS → TLD DNS → Authoritative DNS**

Types of DNS Records:

- **A Record: Maps domain to IPv4 address**
- **MX Record: Mail server info**
- **CNAME: Canonical name alias**

DNS Server Software:

- **`bind9` is the most commonly used DNS server in Linux.**

✅ (D) Write a Note on (Any 1 out of 2) — 5 Marks

(1) Write a Note on SAMBA Server:

Samba is a free software suite that allows file and print sharing between Linux/Unix and Windows systems using the SMB/CIFS protocol.

Features:

- **Share directories and printers with Windows clients**
- **Can act as a domain controller**
- **Supports user authentication and permissions**

Installation:

sudo apt install samba

Configuration:

Edit `/etc/samba/smb.conf` to define shared folders:

[shared]

path = /home/user/shared

read only = no

guest ok = yes

Restart service:

sudo systemctl restart smbd

---

(2) How to Install & Manage Apache Server:

Apache is one of the most widely used web servers on Linux.

Installation (Ubuntu):

sudo apt update

sudo apt install apache2

Basic Commands:

- **Start Apache:** `sudo systemctl start apache2`
- **Enable on boot:** `sudo systemctl enable apache2`
- **Check status:** `sudo systemctl status apache2`

Configuration:

- **Default config file:** `/etc/apache2/apache2.conf`
- **Document root (default site):** `/var/www/html`

To edit homepage:

sudo nano /var/www/html/index.html

- **Apache supports modules like PHP, SSL, and can serve dynamic web pages.**