Department of MCA

Details of Lesson Plan

S.No.	Particulars	Details
1.	Course Name	Operating System
2.	Course Code	MCA-101
3.	Academic Year	2024-2025
4.	Semester	1st
5.	Number of Lesson plans	40
6.	Faculty Assigned	Sukhmeet Kour

Sukhmeet Kour
Faculty Signature

Lesson Plan No. 1.1	Course Name: Operating System	Course No.: MCA-101

Objectives	**Introduction to Operating System** At the end of the lesson the student shall be able to: a. Understand the basic functions and characteristics of an operating system (OS). b. Explain the historical evolution of operating systems. c. Identify various OS services and user interfaces. d. Describe the basic computer system architecture related to OS.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips showing historical evolution of OS c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** • What operating systems do you use on your devices (e.g., phones, laptops)? • Can you name some popular operating systems? • Have you ever wondered what happens when you turn on your computer? **Concept of Operating System:** https://youtu.be/vBURTt97EkA?list=PLBlnK6fEyqRiVhbXDGLXDk_OQAeuVcp2O **Link:** https://www.youtube.com/watch?v=gxhxcvBuanU **Understanding Operating System** **Link:** https://www.youtube.com/watch?v=fkGCLIQx1MI 2. **Introduction to OS:** • Define an operating system: "An OS is software that acts as an intermediary between the user of a computer and the computer hardware." • Show a simple diagram of how an OS interacts with hardware and software. How Operating System Works?

	Link: https://www.youtube.com/watch?v=GjNp0bBrjmU **Development (30 minutes)** **Historical Evolution of Operating Systems (10 minutes):** ○ Brief overview from early batch systems to modern multi-user and multitasking systems. **Video Reference:** History of Operating Systems **Evolution of OS :** **Link:** https://youtu.be/9TLDCr5pKrc ○ Discuss key milestones: ▪ 1950s: Batch processing systems ▪ 1960s: Time-sharing systems ▪ 1970s: Introduction of personal computers ▪ 1980s: Graphical user interfaces (GUIs) 2. **OS Services and User Interfaces (10 minutes):** ○ Describe OS services such as process management, memory management, file system management, and device management. ○ **Real-time Example:** Illustrate with a running application, explaining how the OS manages resources. ○ Discuss different user interfaces: ▪ Command-Line Interface (CLI) ▪ Graphical User Interface (GUI) ○ **Web Reference:** OS User Interfaces 3. **Computer System Architecture (10 minutes):** • Explain basic components: CPU, memory, I/O devices, and how the OS manages these components. • **Diagram Reference:** Slide showing computer system architecture. **Exercise (5 minutes)** • **Activity:** Pair students and ask them to list the tasks an OS performs when a user opens an application.
Closure	Summarize key points: • Definition and functions of an OS

	• Historical evolution • OS services and user interfaces • Basic computer system architecture **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 1:** Introduction, pp. 1-30 **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 1:** Introduction to Operating Systems, pp. 1-35
Evaluation	• **Reflective Questions:** Allow students to discuss and answer why operating systems are critical and how they have evolved. **Practice Question:** Write a short note on the evolution of operating systems and their impact on modern computing.

Lesson Plan No. 1.2	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Operating System Services** At the end of the lesson the student shall be able to: a. Understand the various services provided by an operating system (OS). b. Explain how these services support system functionality and user experience. c. Provide real-time examples of OS services in action.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips showing OS Services c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** - What tasks do you perform daily on your computer (e.g., opening files, running programs)? - How do you think your operating system helps you with these tasks? **Link :** https://youtu.be/TQWERtMoKbI?list=PLBlnK6fEyqRiVhbXDGLXDk_OQAeuVcp2O **Introduction to Goals & Functions of Operating System:** **Link:** https://youtu.be/ACsLvXuaKxw?list=PLRJP-g0nSX0e0kgIW6bom0kQtx4dyHyZx 2. **Introduction to OS Services:** - Define OS services: "Operating system services provide essential functions for the efficient execution of application programs and management of the system." - Show a simple diagram of the OS architecture highlighting the service layer. **Development (30 minutes)** 1. **Description of OS Services (10 minutes):** **Process Management:**

	▪ Explain how the OS manages processes, including process creation, scheduling, and termination. ▪ **Real-time Example:** Demonstrate with Task Manager or Activity Monitor. ▪ **Web Reference:** Process Management in Operating Systems **Memory Management:** ▪ Describe how the OS allocates and deallocates memory for programs. ▪ **Real-time Example:** Discuss how a running application uses RAM. ▪ **Web Reference:** Memory Management in Operating Systems 2. **File System Management (10 minutes):** • Explain the role of the OS in managing files and directories. • **Real-time Example:** Show how file permissions work in Windows or Unix-based systems. • **Web Reference:** File System Management 3. **Device Management (5 minutes):** • Describe how the OS handles input and output devices. • **Real-time Example:** Show how to check and update device drivers. • **Web Reference:** Device Management 4. **Security and Protection (5 minutes):** • Explain the importance of security services provided by the OS. • **Real-time Example:** Discuss user authentication and file encryption. • **Web Reference:** Security in Operating Systems **Interactive Exercise (5 minutes)** • **Activity:** Pair students and ask them to list OS services they used in the last 24 hours and how these services enhanced their experience.
Closure	Summarize key points: • Definition and importance of OS services

	• Examples of process management, memory management, file system management, device management, and security **Assignment:** Write a detailed description of a day in your life using OS services and how each service contributes to your activities. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 2:** Operating System Structures, pp. 31-65 **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2:** Processes and Threads, pp. 36-85
Evaluation	**Reflective Questions:** Allow students to discuss and answer how OS services are integral to system functionality.

Lesson Plan No. 1.3	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Operating System Users** At the end of the lesson the student shall be able to: a. Identify different types of operating system users. b. Understand the needs and requirements of each user type. c. Explain how operating systems cater to these different user groups.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips illustrating different user interactions c. Chalkboard/Whiteboard
Teaching Development	1. **Pre-Discussion Questions:** • Who are the typical users of an operating system? • Can you think of different environments where operating systems are used (e.g., home, business, servers)? **Link** : https://youtu.be/psDpbWscPuE?list=PLBlnK6fEyqRiVhbXDGLXDk_OQAeuVcp2O 2. **Introduction to OS Users:** • Define operating system users: "Operating system users are the individuals or entities that interact with the computer system through the operating system to perform tasks." • Show a simple diagram of different types of users. **Development (30 minutes)** 1. **Types of Operating System Users (10 minutes):** **End Users:** ▪ Define and describe end users. ▪ **Real-time Example:** Discuss how a typical home user interacts with an OS for tasks like browsing, gaming, and document editing. ▪ **Web Reference:** End Users and Their Needs **System Administrators:** ▪ Define and describe system administrators.

	▪ **Real-time Example:** Explain how system admins manage user accounts, security settings, and software installations. ▪ **Web Reference:** Role of System Administrators **Developers:** ▪ Define and describe developers. ▪ **Real-time Example:** Discuss how developers use an OS to write, test, and deploy applications. ▪ **Web Reference:** Developers and Operating Systems 2. **Needs and Requirements of Each User Type (10 minutes):** **End Users:** ▪ Focus on ease of use, graphical user interfaces, and application support. ▪ **Example:** Discuss popular OS features like taskbars, start menus, and app stores. **System Administrators:** ▪ Focus on system stability, security, and user management tools. ▪ **Example:** Explain how system admins use tools like Control Panel in Windows or Terminal in Linux for system management. **Developers:** ▪ Focus on development environments, access to system resources, and support for programming languages. ▪ **Example:** Discuss how developers use IDEs (Integrated Development Environments) and command-line tools. 3. **Interactive Exercise (5 minutes)** • **Activity:** Pair students and ask them to list OS features they use frequently and classify them according to user type. • Discuss their answers and provide feedback.
Closure	Summarize key points: • Different types of OS users: end users, system administrators, developers.

	- Needs and requirements of each user type. - How OS caters to these user groups. **Reflective Questions:** - Which user type do you identify with the most and why? - How does understanding user types help in designing better operating systems? **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 3:** Operating-System Structures, pp. 66-100 **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3**: Processes, pp. 86-130
Evaluation	**Reflective Questions:** Allow students to discuss and answer how understanding OS user types is important. Question: Write a detailed description of how you interact with an OS in your daily life and categorize yourself as one of the user types discussed.

Lesson Plan No. 1.4	Course Name: Operating System	Course No.: MCA-101

Objectives	**Computer System Architecture** At the end of the lesson the student shall be able to: a. Understand the basic components of a computer system. b. Describe how these components interact within the system architecture. c. Explain the role of the operating system in managing computer system architecture.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips demonstrating system architecture c. Chalkboard/Whiteboard
Teaching Development	1. **Pre-Discussion Questions:** • What do you think are the main components inside a computer? • How do you think these components work together to run applications? **Link :** https://youtu.be/HB4I2CgkcCo **Link to Introduce the Topic:** Computer System Architecture Basics : https://youtu.be/So9SR3qpWsM 2. **Introduction to Computer System Architecture:** • Define computer system architecture: "It is the conceptual design and fundamental operational structure of a computer system." • Show a simple diagram of a computer system highlighting the CPU, memory, and I/O devices. **Development (30 minutes)** 1. **Basic Components of a Computer System (10 minutes):** **Central Processing Unit (CPU):** ▪ Explain the function of the CPU as the brain of the computer.

	▪ **Real-time Example:** Show a task manager displaying CPU usage. ▪ **Web Reference:** CPU and Its Functions **Memory:** ▪ Describe types of memory (RAM, ROM) and their roles. ▪ **Real-time Example:** Discuss how applications use RAM. ▪ **Web Reference:** Memory in Computers **Input/Output Devices:** ▪ Explain the role of I/O devices in a computer system. ▪ **Real-time Example:** Show how a keyboard input is processed. ▪ **Web Reference:** Input and Output Devices 2. **Interaction of Components (10 minutes):** **System Bus:** ▪ Explain the function of the system bus in connecting components. ▪ **Real-time Example:** Diagram showing data flow between CPU, memory, and I/O devices. **Role of the Operating System:** ▪ Describe how the OS manages communication between components. ▪ **Real-time Example:** Show an OS managing hardware resources using system calls. ▪ **Web Reference:** System Calls in OS 3. **Detailed Example (10 minutes):** Walkthrough of a simple process like opening a file, showing the role of each component. **Video Reference:** How Computers Work: File Management
Closure	Summarize key points: ● Components of a computer system: CPU, memory, I/O devices ● Interaction of components via system bus ● Role of the OS in managing the system architecture

	Reflective Questions: - Why is understanding computer system architecture important? - How does the OS enhance the functionality of these components?
	Suggested Readings 1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 1:** Introduction, pp. 1-30 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 1:** Introduction to Operating Systems, pp. 1-35
Evaluation	**Reflective Questions:** Allow students to discuss and answer how understanding system architecture helps in system optimization. **Assignment:** Write a detailed description of how an application runs on a computer, focusing on the interaction of components.

Lesson Plan No. 1.5	Course Name: Operating System	Course No.: MCA-101

Objectives	**Operating System Design, Implementation, and Structure** At the end of the lesson the student shall be able to: a. Understand the design principles of an operating system. b. Explain the implementation techniques used in operating systems. c. Describe the structure of operating systems with real-world examples.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips illustrating OS design and implementation c. Chalkboard/Whiteboard
Teaching Development	1. **Pre-Discussion Questions:** ○ What do you think are the core components of an operating system? ○ How do you think these components are designed and implemented? **Link to Introduce the Topic:** Introduction to Operating System Design : https://youtu.be/t_McsJ1RGQg 2. **Introduction to OS Design, Implementation, and Structure:** ○ Define operating system design: "Operating system design refers to the principles and strategies used to build an OS." ○ Show a simple diagram of OS architecture highlighting design layers. **Development (30 minutes)** 1. **Design Principles of Operating Systems (10 minutes):** ○ **Modularity:** ▪ Explain the importance of modular design in operating systems. ▪ **Real-time Example:** Discuss how Linux uses modularity to load and unload kernel modules. ▪ **Web Reference:** Modular Operating System Design ○ **Simplicity and Efficiency:** ▪ Describe how OS design aims for simplicity and efficiency to enhance performance.

	▪ **Real-time Example:** Show how simple design reduces overhead in embedded systems. ▪ **Web Reference:** Design Principles of Operating Systems 2. **Implementation Techniques (10 minutes):** o **System Calls:** ▪ Explain how system calls serve as the interface between the OS and user programs. ▪ **Real-time Example:** Demonstrate a simple system call in C. ▪ **Web Reference:** System Calls in OS o **Layered Approach:** ▪ Describe the layered approach to OS implementation. ▪ **Real-time Example:** Discuss how layers can improve system organization and security. ▪ **Web Reference:** Layered Operating System 3. **Structure of Operating Systems (10 minutes):** o **Monolithic Structure:** ▪ Explain the monolithic structure where the entire OS operates in kernel mode. ▪ **Real-time Example:** Discuss Unix as an example of a monolithic OS. ▪ **Web Reference:** Monolithic Kernel o **Microkernel Structure:** ▪ Describe the microkernel approach which minimizes kernel size. ▪ **Real-time Example:** Discuss the Mach microkernel used in macOS. ▪ **Web Reference:** Microkernel Architecture **Interactive Exercise (5 minutes)** • **Activity:** Pair students and ask them to outline a basic OS design using either a monolithic or microkernel structure. • Discuss their designs and provide feedback.
Closure	• Summarize key points: o Design principles: modularity, simplicity, efficiency o Implementation techniques: system calls, layered approach o OS structures: monolithic, microkernel • **Reflective Questions:** o Why is modularity important in OS design? o How does the structure of an OS impact its performance and security?
	Suggested Readings

	1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 2:** Operating-System Structures, pp. 66-100 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 1:** Introduction to Operating Systems, pp. 1-35
Evaluation	**Reflective Questions:** Allow students to discuss and answer how different design principles affect OS functionality.

Lesson Plan No. 1.6	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**System Calls and System Programs** At the end of the lesson the student shall be able to: a. Understand what system calls are and their role in operating systems. b. Describe the different types of system calls. c. Understand what system programs are and how they interact with system calls. d. Explain the functions and types of system programs.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips demonstrating system calls and programs c. Chalkboard/Whiteboard
Teaching Development	1. **Pre-Discussion Questions:** ○ What do you think happens when you save a file on your computer? ○ How do you think your computer starts a new program? **Link to Introduce the Topic:** ○ Introduction to System Calls ○ Introduction to System Programs https://youtu.be/lhToWeuWWfw 2. **Introduction to System Calls and System Programs:** ○ Define system calls: "System calls provide an interface between a running program and the operating system." ○ Define system programs: "System programs provide a convenient environment for program development and execution." **Development (30 minutes)** 1. **System Calls (15 minutes):** ○ **Definition and Purpose:** ▪ Explain how system calls allow user-level processes to request services from the kernel. ▪ **Real-time Example:** Saving a file involves system calls like `open()`, `write()`, and `close()`. ○ **Types of System Calls:** ▪ **Process Control:** ▪ Examples: `fork()`, `exit()`

            - **Web Reference:** Process Control System Calls
        - **File Management:**
            - Examples: `open()`, `read()`, `write()`, `close()`
            - **Web Reference:** File Management System Calls
        - **Device Management:**
            - Examples: `ioctl()`, `read()`, `write()`
            - **Web Reference:** Device Management System Calls
        - **Information Maintenance:**
            - Examples: `getpid()`, `alarm()`
            - **Web Reference:** Information Maintenance System Calls
        - **Communication:**
            - Examples: `pipe()`, `shmget()`, `mmap()`
            - **Web Reference:** Communication System Calls

2. **System Programs (15 minutes):**
    - **Definition and Purpose:**
        - Describe how system programs help manage and maintain computer systems.
        - **Real-time Example:** System utilities like task manager, disk cleanup, and text editors.
    - **Types of System Programs:**
        - **File Management:**
            - Examples: file editors, file management utilities
            - **Web Reference:** File Management System Programs
        - **Status Information:**
            - Examples: system monitoring tools
            - **Web Reference:** Status Information System Programs
        - **File Modification:**
            - Examples: text editors
            - **Web Reference:** File Modification System Programs
        - **Programming Language Support:**
            - Examples: compilers, assemblers
            - **Web Reference:** Programming Language Support System Programs
        - **Program Loading and Execution:**
            - Examples: debuggers, loaders
            - **Web Reference:** Program Loading and Execution System Programs

	▪ **Communications:** ▪ Examples: email clients, web browsers ▪ **Web Reference:** Communication System Programs **Interactive Exercise (5 minutes)** **Activity:** Pair students and ask them to outline the steps (system calls) involved in copying a file from one directory to another.
Closure	• Summarize key points: ● System calls: interface between programs and the OS. ● System programs: provide an environment for program development and execution. • **Reflective Questions:** ● How do system calls enhance program functionality? ● Why are system programs essential for user convenience? **Suggested Readings** 1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 2:** Operating-System Structures, pp. 66-100 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 1:** Introduction to Operating Systems, pp. 1-35
Evaluation	**Reflective Questions:** Allow students to discuss and answer how system calls and system programs simplify OS functionality.

Lesson Plan No. 1.7	Course Name: Operating System	Course No.: MCA-101

Objectives	**Virtual Machines and Spooling** At the end of the lesson the student shall be able to: a. Understand the concept and architecture of virtual machines. b. Explain the purpose and functioning of spooling in operating systems. c. Identify real-time applications of virtual machines and spooling.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips illustrating virtual machines and spooling c. Chalkboard/Whiteboard
Teaching Development	1. **Pre-Discussion Questions:** o Have you ever used a virtual machine? o What do you think happens in the background when multiple print jobs are sent to a printer? **Link to Introduce the Topic:** o Introduction to Virtual Machines o Introduction to Spooling **Link :** https://youtu.be/daDbY2iDmU0?list=PLBlnK6fEyqRiVhbXDGLXDk_OQAeuVcp2O 2. **Introduction to Virtual Machines and Spooling:** o Define virtual machines: "A virtual machine is a software emulation of a physical computer." o Define spooling: "Spooling is a process of placing data in a temporary working area for another program to process." **Development (30 minutes)** 1. **Virtual Machines (15 minutes):** o **Definition and Purpose:** ▪ Explain how virtual machines allow multiple OS instances on a single physical machine. ▪ **Real-time Example:** Using VMware or VirtualBox to run different OSs on a PC. ▪ **Web Reference:** Virtual Machines Explained o **Architecture of Virtual Machines:** ▪ **Components:**

	▪ Virtual Hardware: CPU, memory, storage ▪ Hypervisor: Manages and allocates resources to virtual machines. ▪ **Types:** ▪ Type 1 (Bare-metal): Runs directly on hardware (e.g., VMware ESXi). ▪ Type 2 (Hosted): Runs on a host OS (e.g., VMware Workstation, VirtualBox). ▪ **Web Reference:** Hypervisors o **Advantages and Use Cases:** ▪ Isolation, Consolidation, Testing, and Development. ▪ **Real-time Example:** Using virtual machines for software testing across different OS environments. ▪ **Web Reference:** Advantages of Virtual Machines 2. **Spooling (15 minutes):** o **Definition and Purpose:** ▪ Explain how spooling manages data by placing it in a buffer. ▪ **Real-time Example:** Printing documents where multiple print jobs are queued. ▪ **Web Reference:** Spooling o **Functioning of Spooling:** ▪ **Components:** ▪ Input devices, spooler, output devices. ▪ **Steps Involved:** ▪ Data is temporarily held in a spool. ▪ Data is processed and sent to the output device. ▪ **Web Reference:** Spooling Process o **Advantages and Use Cases:** ▪ Efficiency, Resource Sharing, Process Management. ▪ **Real-time Example:** Email systems where messages are queued before being sent. ▪ **Web Reference:** Advantages of Spooling **Interactive Exercise (5 minutes)** **Activity:** Pair students and ask them to outline how spooling works in a print server environment.
Closure	• Summarize key points: ● Virtual machines: Software emulations, types, and uses. ● Spooling: Data management process, advantages, and real-time applications. • **Reflective Questions:** ● How do virtual machines enhance resource utilization? ● Why is spooling important in managing multiple tasks?

	Suggested Readings 1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 5:** Virtual Machines, pp. 200-220 **Chapter 6:** Spooling and Buffering, pp. 230-245 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2:** Virtual Machines, pp. 60-85 **Chapter 3:** Spooling, pp. 90-110
Evaluation	**Reflective Questions:** Allow students to discuss and answer how virtual machines and spooling impact system efficiency.

Lesson Plan No. 2.1	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Introduction to Process Management** At the end of the lesson the student shall be able to: a. Understand the concept of a process in an operating system. b. Differentiate between process and program. c. Explain process states and the state transition diagram. d. Illustrate real-time examples of process management.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips explaining process management. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** **1. Pre-discussion questions:** ○ What do you think happens when you run a program on your computer? ○ Can you differentiate between a process and a program? **2. Introduction to the topic:** ○ Define a process: "A process is a program in execution." ○ Use a simple diagram to show the life cycle of a process. **Development (30 minutes)** 1. **Definition and Concept of Process (10 minutes):** ○ Discuss what a process is and its significance in an OS. ○ Real-time example: Opening a web browser and how it becomes a process. ○ YouTube Reference: What is a Process? 2. **Process States and State Transition Diagram (10 minutes):** ○ Explain the different states of a process (New, Ready, Running, Waiting, Terminated). ○ Show and explain the state transition diagram. ○ Real-time example: Running a program and the state changes it undergoes. ○ Web Reference: Process States 3. **Process Control Block (PCB) (10 minutes):** ○ Describe what a PCB is and its role in process management. ○ Real-time example: PCB details when you open Task Manager on Windows. ○ Video Reference: Process Control Block **Exercise (5 minutes)**

	- Activity: Ask students to pair up and list the states a process goes through when they run a simple program like a text editor.
Closure	Summarize key points: - Definition and concept of a process. - Process states and state transition diagram. - Importance of Process Control Block. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 3: Processes, pp. 101-140.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 45-80.**
Evaluation	- **Reflective questions:** Ask students to discuss why process management is crucial in operating systems. - **Assignment:** Write a short note on the role of process management in operating systems and provide examples from your own usage.

Lesson Plan No. 2.2	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Study of State Models, Process Scheduling** At the end of the lesson the student shall be able to: a. Understand the different process state models. b. Explain process scheduling and its importance. c. Illustrate real-time examples of process scheduling.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips explaining state models and scheduling. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** **1. Pre-discussion questions:** ○ What do you think happens if multiple processes are running on your computer? ○ How does the OS decide which process to run next? **2. Introduction to the topic:** ○ Briefly explain process state models and process scheduling. **Development (30 minutes)** 1. **Process State Models (10 minutes):** ○ Explain the different process state models (e.g., two-state, five-state models). ○ Show and explain diagrams for these models. ○ Real-time example: Multitasking on a computer. ○ YouTube Reference: Process States and Models 2. **Process Scheduling (20 minutes):** ○ Define process scheduling and its objectives. ○ Explain types of schedulers (long-term, short-term, and medium-term). ○ Real-time example: Task switching in a multi-user environment. ○ Video Reference: Process Scheduling ○ Web Reference: Types of Schedulers **Exercise (5 minutes)** • Activity: Ask students to pair up and list the steps the OS might take to switch from one process to another.
Closure	Summarize key points:

	• Process state models. • Process scheduling and its importance. • Types of schedulers **Assignment:** Write a short note on the different process state models and their significance in process management. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 5: CPU Scheduling, pp. 171-210.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3: CPU Scheduling, pp. 101-140.**
Evaluation	**Reflective questions: Ask students to discuss the importance of process scheduling in maintaining system performance.**

Lesson Plan No. 2.3	Course Name: Operating System	Course No.: MCA-101

Objectives	**Job Scheduling** At the end of the lesson the student shall be able to: a. Understand the concept of job scheduling. b. Differentiate between job scheduling and process scheduling. c. Explain job scheduling criteria and policies. d. Illustrate real-time examples of job scheduling.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips explaining job scheduling. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** • What do you think happens when multiple jobs are submitted to the system? • How does the OS manage these jobs? 2. **Introduction to OS Users:** • Define job scheduling and differentiate it from process scheduling. **Development (30 minutes)** 1. **Definition and Concept of Job Scheduling (10 minutes):** • Explain what job scheduling is and its significance in an OS. • Real-time example: Job queue in a print server. • YouTube Reference: Job Scheduling in Operating Systems 2. **Job Scheduling Criteria and Policies (20 minutes):** • Explain different job scheduling criteria (e.g., CPU utilization, throughput, turnaround time). • Discuss various job scheduling policies (e.g., FCFS, SJF). • Real-time example: Scheduling jobs in a batch processing system. • Web Reference: Job Scheduling Policies • Video Reference: Job Scheduling Policies **Interactive Exercise (5 minutes)** • **Activity:** Ask students to pair up and list the advantages and disadvantages of different job scheduling policies.

Closure	Summarize key points: • Definition and concept of job scheduling. • Job scheduling criteria and policies. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Job Scheduling, pp. 211-250.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4: Job Scheduling, pp. 141-180.**
Evaluation	**Reflective Questions :** Ask students to discuss the importance of job scheduling in optimizing system performance.

Lesson Plan No. 2.4	Course Name: Operating System	Course No.: MCA-101

Objectives	**Scheduling Criteria** At the end of the lesson the student shall be able to: a. Understand various scheduling criteria. b. Explain the significance of scheduling criteria in process management. c. Illustrate real-time examples of scheduling criteria.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips explaining scheduling criteria. c. Chalkboard/Whiteboard
Teaching Development	**Introduction(5 minutes)** 1. **Pre-Discussion Questions:** • What factors do you think are important when scheduling processes? • How does the OS decide the order of process execution? 2. Introduction to the topic: • Briefly explain scheduling criteria and their importance. **Development (30 minutes)** 1. **Different Scheduling Criteria (20 minutes):** • Explain different scheduling criteria (e.g., CPU utilization, throughput, turnaround time, waiting time, response time). • Show and explain diagrams for these criteria. • Real-time example: Comparing the performance of different scheduling algorithms. • YouTube Reference: Scheduling Criteria in Operating Systems 2. **Importance of Scheduling Criteria (10 minutes):** • Discuss the importance of each criterion in optimizing system performance. • Real-time example: Choosing a scheduling algorithm for a specific scenario. • Web Reference: Importance of Scheduling Criteria • Video Reference: Importance of Scheduling Criteria **Exercise (5 minutes)** Activity: Ask students to pair up and list the scheduling criteria that would be most important in a real-time system.

Closure	Summarize key points: • Different scheduling criteria. • Importance of scheduling criteria in process management.
	Suggested Readings 1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 7: Scheduling Criteria, pp. 251-290**. 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 5: Scheduling Criteria, pp. 181-220.**
Evaluation	**Reflective Questions:** Ask students to discuss the significance of different scheduling criteria in optimizing system performance. **Assignment:** Write a short note on the importance of scheduling criteria in process management and provide examples from your own usage.

Lesson Plan No. 2.5	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Scheduling Algorithms - FCFS, SJF** At the end of the lesson the student shall be able to: a. Understand the design principles of an operating system. b. Explain the implementation techniques used in operating systems. c. Describe the structure of operating systems with real-world examples.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips illustrating OS design and implementation c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-discussion questions:** o What do you know about scheduling algorithms? o Have you ever considered how processes are scheduled on your computer? 2. **Introduction to the topic:** o Briefly explain FCFS and SJF scheduling algorithms. **Development (30 minutes)** 1. **First-Come, First-Served (FCFS) Scheduling (15 minutes):** o Explain what FCFS is and how it works. o Show and explain a diagram for FCFS scheduling. o Real-time example: Scheduling tasks in a queue based on arrival time. o YouTube Reference: FCFS Scheduling o Web Reference: FCFS Scheduling 2. **Shortest Job First (SJF) Scheduling (15 minutes):** o Explain what SJF is and how it works. o Show and explain a diagram for SJF scheduling. o Real-time example: Scheduling tasks based on their execution time. o YouTube Reference: SJF Scheduling o Web Reference: SJF Scheduling **Exercise (5 minutes)** Activity: Ask students to pair up and compare the advantages and disadvantages of FCFS and SJF.

Closure	Summarize key points: • Working principles and characteristics of FCFS and SJF. • Real-time examples of FCFS and SJF scheduling.
	Suggested Readings 1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 8: FCFS and SJF Scheduling, pp. 291-330.** 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 6: Scheduling Algorithms, pp. 221-260.**
Evaluation	**Reflective Questions:** Ask students to discuss the advantages and disadvantages of FCFS and SJF in different scenarios. **Assignment:** Write a short note on the working principles of FCFS and SJF scheduling algorithms and provide examples from your own usage.

Lesson Plan No. 1.6	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Scheduling Algorithms - Round Robin, Priority** At the end of the lesson the student shall be able to: a. Understand the Round Robin and Priority scheduling algorithms. b. Explain the working principles and characteristics of Round Robin and Priority. c. Illustrate real-time examples of Round Robin and Priority scheduling.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips explaining Round Robin and Priority. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. Pre-discussion questions: • What other scheduling algorithms do you know besides FCFS and SJF? • How do you think priority affects process scheduling? 2. Introduction to the topic: • Briefly explain Round Robin and Priority scheduling algorithms. **Development (30 minutes)** 1. **Round Robin Scheduling (15 minutes):** • Explain what Round Robin is and how it works. • Show and explain a diagram for Round Robin scheduling. • Real-time example: Time-sharing systems in a multi-user environment. • YouTube Reference: Round Robin Scheduling • Web Reference: Round Robin Scheduling 2. **Priority Scheduling (15 minutes):** • Explain what Priority scheduling is and how it works. • Show and explain a diagram for Priority scheduling. • Real-time example: Scheduling tasks based on their priority level. • YouTube Reference: Priority Scheduling

	- Web Reference: Priority Scheduling **Exercise (5 minutes)** Activity: Ask students to pair up and discuss the scenarios where Round Robin and Priority scheduling would be most effective.
Closure	Summarize key points: - Working principles and characteristics of Round Robin and Priority. - Real-time examples of Round Robin and Priority scheduling. **Suggested Readings** 1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 9: Round Robin and Priority Scheduling, pp. 331-370.** 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 7: Advanced Scheduling Algorithms, pp. 261-300.**
Evaluation	**Reflective questions:** Ask students to discuss the advantages and disadvantages of Round Robin and Priority in different scenarios. **Assignment:** Write a short note on the working principles of Round Robin and Priority scheduling algorithms and provide examples from your own usage.

Lesson Plan No. 2.7	Course Name: Operating System	Course No.: MCA-101

Objectives	**Multiple Process Scheduling** At the end of the lesson the student shall be able to: a. Understand the concept of multiple process scheduling. b. Explain various multiple process scheduling techniques. c. Illustrate real-time examples of multiple process scheduling.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips explaining multiple process scheduling. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. Pre-discussion questions: • How do you think multiple processes are scheduled simultaneously? • What challenges might arise in scheduling multiple processes? 2. Introduction to the topic: • Briefly explain multiple process scheduling and its challenges. **Development (30 minutes)** 1. **Multiple Process Scheduling Techniques (20 minutes):** • Explain different multiple process scheduling techniques (e.g., Multilevel Queue Scheduling, Multilevel Feedback Queue Scheduling). • Show and explain diagrams for these techniques. • Real-time example: Scheduling tasks in a multi-user environment. • YouTube Reference: Multiple Process Scheduling • Web Reference: Multiple Process Scheduling Techniques 2. **Challenges in Multiple Process Scheduling (10 minutes):** • Discuss the challenges and solutions in multiple process scheduling. • Real-time example: Handling priority inversion and starvation.

	• Video Reference: Challenges in Multiple Process Scheduling • Web Reference: Challenges in Scheduling **Exercise (5 minutes)** Activity: Ask students to pair up and discuss the best techniques for scheduling multiple processes in a real-time system.
Closure	Summarize key points: • Multiple process scheduling techniques. • Challenges and solutions in multiple process scheduling.
	Suggested Readings 1. **Book:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 10: Multiple Process Scheduling, pp. 371-410.** 2. **Book:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 8: Multiprocessing and Multithreading, pp. 301-340.**
Evaluation	**Reflective questions:** Ask students to discuss the significance of multiple process scheduling in optimizing system performance. **Assignment:** Write a short note on the different multiple process scheduling techniques and their impact on system performance.

Lesson Plan No. 2.9	**Course Name: Operating System (Topic: Case study)**	**Course No.: MCA-101**

Objectives	At the end of the lesson the student shall be able to: a. Apply knowledge of scheduling algorithms to real-world scenarios. b. Analyze case studies of process scheduling in different operating systems.
Teaching Aids (if any)	a. White Board (Chalk and Talk). b. Projector and Laptops c. Case study materials
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Briefly introduce the purpose of case studies in understanding scheduling. - Present a case study of scheduling in a real operating system (e.g., Linux, Windows). - **Development** (30 minutes) **Real-World OS Scheduling** - Present a case study of scheduling in a real operating system (e.g., Linux, Windows). - Analyze the scheduling algorithm used and its impact. - Discuss its advantages and disadvantages. - **Comparative Analysis** - Present a comparative analysis of scheduling in two different operating systems. - Discuss the pros and cons of each approach. - **Exercise** (5 minutes) – - Address any questions from students. - Facilitate a discussion on the case studies presented.
Closure	1. Summarize key takeaways from the case studies.
Evaluation	1. Group discussion on the effectiveness of different scheduling algorithms in real-world scenarios. 2. Ask questions to the students to gauge comprehension.

Lesson Plan No. 3.1	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Process Coordination - Synchronization** At the end of the lesson the student shall be able to: a. Understand the concept of process synchronization. b. Identify problems caused by lack of synchronization. c. Learn techniques to ensure proper synchronization.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Example code snippets
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ Have you ever seen a program where two processes seem to interfere with each other? ○ What do you think happens when multiple processes access shared resources? ○ Definition: Synchronization is a technique to ensure that two or more processes do not interfere with each other while accessing shared resources. **Development (30 minutes)** 2. **Understanding Synchronization (10 minutes)** ○ Define synchronization. ○ Explain critical sections and the need for synchronization. ○ Show example: bank account transactions where two processes access and modify the balance concurrently. 3. **Problems Due to Lack of Synchronization (10 minutes)** ○ Describe race conditions. ○ Real-time Example: Inconsistent bank account balance after concurrent transactions. ○ Show example code without synchronization and explain the issues. 4. **Techniques for Synchronization (10 minutes)** ○ Introduce locks, semaphores, and monitors. ○ Show example code with proper synchronization using semaphores. ○ **Real-time Example:** ○ Discuss different user interfaces:

	▪ Command-Line Interface (CLI) ▪ Graphical User Interface (GUI) ○ **Web Reference:** OS User Interfaces 5. **Computer System Architecture (10 minutes):** ○ Explain basic components: CPU, memory, I/O devices, and how the OS manages these components. ○ **Diagram Reference:** Slide showing computer system architecture. **Exercise (5 minutes)** **Activity:** □ Pair students and provide a problem scenario involving shared resources. □ Ask them to suggest synchronization techniques to solve the problem. Discuss their answers.
Closure	Summarize key points: ● Emphasize the importance of synchronization in concurrent programming. **Web References:**https://www.nesoacademy.org/cs/03-operating-system/ppts/06-process-synchronization **YouTube References:** https://www.youtube.com/watch?v=ph2awKa8r5Y **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 258-278.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 85-112.**
	● **Reflective Questions**:Why is synchronization important in operating systems? Can you think of real-world examples where synchronization is crucial? ● **Assignment**: Write a short essay on a real-world scenario where synchronization is essential and describe the techniques used to achieve it.

Lesson Plan No. 3.2	Course Name: Operating System	Course No.: MCA-101

Objectives	**Race Conditions** At the end of the lesson the student shall be able to: a. Define race conditions. b. Understand how race conditions occur. c. Learn methods to prevent race conditions.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● Have you ever seen an unpredictable output from a program when run multiple times? ● What do you think causes such behavior in concurrent systems? ● **Definition**: A race condition occurs when two or more processes or threads access shared data and try to change it at the same time. **Development (30 minutes)** □ **Understanding Race Conditions (10 minutes)** ● Define race conditions. ● Explain how race conditions occur with examples. ● Show example: Incrementing a counter by multiple threads without proper synchronization. □ **Identifying Race Conditions (10 minutes)** ● Discuss how to detect race conditions. ● Show code examples where race conditions can occur. ● Explain tools and techniques for identifying race conditions (e.g., thread sanitizers). □ **Preventing Race Conditions (10 minutes)** ● Introduce locks, mutexes, and semaphores. ● Show example code with and without race condition prevention techniques. ● Explain the importance of atomic operations.

	Exercise (5 minutes) **Activity:** □ Pair students and provide a problem scenario involving race conditions. □ Ask them to suggest methods to prevent race conditions in the given scenario. □ Discuss their answers.
Closure	Summarize key points: ● Highlight the importance of preventing race conditions for system reliability and correctness. **Web References-** ● https://www.nesoacademy.org/cs/03-operating-system/ppts/06-process-synchronization **YouTube References-** ● https://www.youtube.com/watch?v=7aF0q7NfwfA **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 278-288.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 112-130.**
	● **Reflective Questions**: Can you identify a race condition in the provided code? How would you fix it? ● **Assignment**: Write a short program that demonstrates a race condition and then modify it to prevent the race condition.

Lesson Plan No. 3.3	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Critical Section Problems, Semaphores** At the end of the lesson the student shall be able to: a. Understand the concept of critical sections. b. Learn about critical section problems and their solutions. c. Understand semaphores and their usage in synchronization.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● What happens when multiple threads or processes try to access a shared resource simultaneously? ● Can you think of situations where resource access must be controlled? ● **Definition**: A critical section is a part of a program where shared resources are accessed. It needs to be executed as an atomic action to prevent race conditions. **Development (30 minutes)** ● **Understanding Critical Section Problems (10 minutes)** ○ Define critical section problems. ○ Explain the conditions necessary to avoid critical section problems: Mutual Exclusion, Progress, and Bounded Waiting. ○ Show example scenarios: Printer spooler, shared counters. ● **Semaphores as a Solution (10 minutes)** ○ Define semaphores and types (binary semaphores, counting semaphores). ○ Explain how semaphores can be used to solve critical section problems. ○ Show example code using semaphores to manage access to critical sections. ● **Implementation Examples (10 minutes)** ○ Demonstrate using pseudocode or actual code examples. ○ Explain the semaphore operations: wait (P) and signal (V). ○ Discuss real-time examples: controlling access to a file, managing database connections.. **Exercise (5 minutes)**

	Activity: □ Pair students and provide a code snippet with a critical section problem. □ Ask them to identify the problem and implement a solution using semaphores. □ Discuss their solutions and correct any misconceptions.
Closure	Summarize key points: ● Emphasize the importance of properly managing critical sections to avoid race conditions and ensure data consistency. **YouTube References** ● https://www.youtube.com/watch?v=UtEORPakw5Y ● https://m.youtube.com/live/XDIOC2EY5JE?t=0s **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 288-302.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 130-142.**
	● **Reflective Questions**: What are the three conditions that must be satisfied to avoid critical section problems? How do semaphores help in managing critical sections? ● **Assignment**: Write a program that uses semaphores to solve a critical section problem. Explain how your solution ensures mutual exclusion, progress, and bounded waiting.

Lesson Plan No. 3.4	Course Name: Operating System	Course No.: MCA-101

Objectives	**Bounded Buffer Problem** At the end of the lesson the student shall be able to: a. Understand the bounded buffer (producer-consumer) problem. b. Learn how to implement a solution using semaphores.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● Have you ever used a system where producers generate data and consumers process it? ● What happens if the producer generates data faster than the consumer can process it? ● **Definition**: The bounded buffer problem is a classic synchronization problem where a fixed-size buffer is used by multiple producers and consumers. **Development (30 minutes)** **1. Understanding the Bounded Buffer Problem (10 minutes)** ● Define the bounded buffer problem. ● Explain the roles of producers and consumers. ● Show example scenarios: logging systems, data streaming. 2.**Implementing the Solution (10 minutes)** ● Introduce semaphores for synchronization. ● Explain the use of two semaphores: full and empty. ● Show pseudocode for producer and consumer processes. 3.**Code Example (10 minutes)** ● Demonstrate using actual code examples. ● Explain the semaphore operations in the context of the bounded buffer problem. ● Discuss potential issues and how to resolve them (e.g., deadlocks).

	Exercise (5 minutes) **Activity:** □ Pair students and provide a code snippet with a critical section problem. □ Ask them to identify the problem and implement a solution using semaphores. □ Discuss their solutions and correct any misconceptions.
Closure	Summarize key points: ● Emphasize the importance of properly managing critical sections to avoid race conditions and ensure data consistency. **YouTube References** ● https://www.youtube.com/watch?v=Qx3P2wazwI0 **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 288-302.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 130-142.**
	● **Reflective Questions**: What are the three conditions that must be satisfied to avoid critical section problems? How do semaphores help in managing critical sections? ● **Assignment**: Write a program that uses semaphores to solve a critical section problem. Explain how your solution ensures mutual exclusion, progress, and bounded waiting.

Lesson Plan No. 3.5	Course Name: Operating System	Course No.: MCA-101

Objectives	**Readers-Writers Problem** **Objectives** At the end of the lesson the student shall be able to: a. Understand the readers-writers problem. b. Learn different solutions to the readers-writers problem.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● Can you think of scenarios where multiple readers access data simultaneously but only one writer updates it? ● What issues arise when readers and writers try to access shared data concurrently? ● **Definition**: The readers-writers problem is a classic synchronization problem where multiple readers can access shared data simultaneously, but writers require exclusive access. **Development (30 minutes)** 1. **Understanding the Readers-Writers Problem (10 minutes)** ● Define the readers-writers problem. ● Explain the constraints: multiple readers can read simultaneously, but only one writer can write. ● Show example scenarios: database access, file reading/writing. 2. **First Readers-Writers Problem (10 minutes)** ● Describe the first readers-writers problem: readers have priority over writers. ● Show pseudocode and actual code examples. ● Discuss potential issues: starvation of writers. 3.**Second Readers-Writers Problem (10 minutes)** ● Describe the second readers-writers problem: writers have priority over readers.

	● Show pseudocode and actual code examples.Discuss potential issues: starvation of readers. **Exercise (5 minutes)** **Activity:** □ Pair students and provide a scenario involving readers and writers. ● Ask them to implement a solution using semaphores. ● Discuss their solutions and correct any misconceptions.
Closure	Summarize key points: ● Emphasize the importance of balancing priorities between readers and writers. **YouTube References** ● https://www.youtube.com/watch?v=p2XDhW5INOo **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 310-320.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 150-160.**
	□ **Reflective Questions**: What are the differences between the first and second readers-writers problems? How do you ensure no starvation of readers or writers? □ **Assignment**: Write a program that solves the readers-writers problem with priority given to writers. Explain how your solution ensures fairness and prevents starvation.

Lesson Plan No. 3.6	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Dining Philosophers Problem** **Objectives** At the end of the lesson the student shall be able to: a. Understand the dining philosophers problem. b. Learn different solutions to the dining philosophers problem.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● Have you ever encountered a situation where multiple people need limited resources? ● What issues arise when multiple processes try to access limited resources simultaneously? ● **Definition**: The dining philosophers problem is a classic synchronization problem where philosophers alternately think and eat but need two forks to eat and there are limited forks. **Development (30 minutes)** ● **Understanding the Dining Philosophers Problem (10 minutes)** ○ Define the dining philosophers problem. ○ Explain the constraints: each philosopher needs two forks to eat, but there are only five forks for five philosophers. ○ Show example scenarios: resource allocation in concurrent systems. ● **Naive Solution (5 minutes)** ○ Describe a naive solution where each philosopher picks up the left fork first. ○ Show why this solution leads to deadlock. ○ Demonstrate with pseudocode and explain the problem. ● **Avoiding Deadlock (10 minutes)** ○ Introduce solutions to avoid deadlock, such as allowing at most four philosophers to sit down, or picking up forks only if both are available. ○ Show pseudocode and actual code examples. ○ Explain why these solutions work and discuss potential issues. ● **Optimizing Resource Utilization (5 minutes)** ○ Discuss optimization techniques to ensure efficient resource utilization.

	○ Show example scenarios and code snippets. **Exercise (5 minutes)** **Activity:** ● Pair students and provide a partial implementation of the dining philosophers problem. ● Ask them to complete the implementation with deadlock prevention techniques. ● Discuss their solutions and correct any misconceptions.
Closure	Summarize key points: ● Highlight the importance of efficient resource allocation and deadlock prevention in concurrent systems. **YouTube References** ● https://www.youtube.com/watch?v=FYUi-u7UWgw **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 320-330.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 160-170.**
	● **Reflective Questions**: What are the key challenges in the dining philosophers problem? How do you ensure that no philosopher starves? ● **Assignment**: Implement the dining philosophers problem using your preferred programming language. Ensure that your solution prevents deadlock and starvation, and explain your approach.

Lesson Plan No. 3.7	Course Name: Operating System	Course No.: MCA-101

Objectives	**Deadlocks – Characteristics Objectives** At the end of the lesson the student shall be able to: a. Understand the characteristics of deadlocks. b. Learn how deadlocks occur and their effects on systems.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● Have you ever experienced a situation where your computer seems to freeze and stop responding? ● What do you think causes such behavior in concurrent systems? ● **Definition**: A deadlock is a situation where two or more processes are unable to proceed because each is waiting for the other to release a resource. **Development (30 minutes)** ● **Understanding Deadlocks (10 minutes)** ○ Define deadlocks. ○ Explain the four necessary conditions for deadlock: Mutual Exclusion, Hold and Wait, No Preemption, Circular Wait. ○ Show example scenarios: multiple processes waiting for each other to release resources. ● **Deadlock Example (10 minutes)** ○ Provide a detailed example with pseudocode or actual code demonstrating a deadlock scenario. ○ Explain how the conditions for deadlock are satisfied in the example. ○ Discuss the effects of deadlock on system performance. ● **Detecting Deadlocks (10 minutes)** ○ Introduce methods to detect deadlocks. ○ Show examples of deadlock detection algorithms. ○ Explain the limitations and challenges of deadlock detection. **Exercise (5 minutes)** **Activity:** □ Pair students and provide a code snippet that leads to a deadlock.

	□ Ask them to identify the deadlock and explain how the conditions for deadlock are met. □ Discuss their answers.
Closure	Summarize key points: • Emphasize the importance of understanding deadlock characteristics to design robust concurrent systems. **YouTube References** • https://www.youtube.com/watch?v=UVo9mGARkhQ **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 338-350.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 170-180.**
	□ **Reflective Questions**: What are the four necessary conditions for deadlock? How can you detect a deadlock in a system? □ **Assignment**: Write a program that demonstrates a deadlock scenario. Identify the conditions that lead to the deadlock and suggest possible ways to detect it.

Lesson Plan No. 3.8	Course Name: Operating System	Course No.: MCA-101

Objectives	**Deadlock Prevention, Avoidance, Detection and Recovery** At the end of the lesson the student shall be able to: a. Understand methods to prevent, avoid, detect, and recover from deadlocks. b. Learn different strategies for managing deadlocks in operating systems.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● Have you thought about ways to prevent deadlocks before they occur? ● What strategies can be employed to avoid deadlocks in concurrent systems? ● **Definition**: Deadlock prevention, avoidance, detection, and recovery are techniques used to manage deadlocks and ensure system reliability. **Development (30 minutes)** ● **Deadlock Prevention (10 minutes)** ○ Define deadlock prevention. ○ Explain techniques to prevent deadlocks by negating one of the necessary conditions. ○ Show examples: ordering resource requests, limiting resource holding time. ● **Deadlock Avoidance (10 minutes)** ○ Define deadlock avoidance. ○ Explain how deadlock avoidance uses additional information to ensure safe resource allocation. ○ Show examples: Banker's algorithm. ○ Provide pseudocode for Banker's algorithm and explain how it works. ● **Deadlock Detection and Recovery (10 minutes)** ○ Define deadlock detection. ○ Explain how to detect deadlocks using resource allocation graphs and detection algorithms. ○ Show examples: deadlock detection algorithm pseudocode. ○ Discuss recovery techniques: process termination, resource preemption.

	Exercise (5 minutes) **Activity:** □ Pair students and provide a scenario where a deadlock can occur. □ Ask them to suggest methods to prevent, avoid, detect, and recover from the deadlock. □ Discuss their answers and correct any misconceptions.
Closure	Summarize key points: ● Highlight the importance of combining prevention, avoidance, detection, and recovery techniques to manage deadlocks effectively. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 350-370.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 180-200.**
	□ **Reflective Questions**: What are the differences between deadlock prevention and avoidance? How can you recover from a deadlock situation?

Lesson Plan No. 3.9	Course Name: Operating System	Course No.: MCA-101

Objectives	**Banker's Algorithm** At the end of the lesson the student shall be able to: a. Understand the Banker's algorithm for deadlock avoidance. b. Learn how to implement the Banker's algorithm.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Chalkboard/Whiteboard c. Video clip
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ● Have you heard of the Banker's algorithm in the context of resource allocation? ● Why is it important to ensure that a system is in a safe state? ● **Definition**: The Banker's algorithm is a deadlock avoidance algorithm that tests for safe resource allocation. **Development (30 minutes)** ● **Understanding the Banker's Algorithm (10 minutes)** ○ Define the Banker's algorithm. ○ Explain the concepts of safe state and unsafe state. ○ Show example scenarios: resource allocation in a banking system. ● **Steps of the Banker's Algorithm (10 minutes)** ○ Describe the steps involved in the Banker's algorithm: Request, Allocation, Safety Check. ○ Provide pseudocode for the Banker's algorithm. ○ Show detailed examples with step-by-step explanations. ● **Implementing the Banker's Algorithm (10 minutes)** ○ Demonstrate using actual code examples. ○ Explain how to handle resource requests and perform safety checks. ○ Discuss potential issues and how to resolve them. **Exercise (5 minutes)** **Activity:** □ Pair students and provide a set of resource allocation scenarios. ● Ask them to apply the Banker's algorithm to determine if the system is in a safe state.

	● Discuss their solutions and correct any misconceptions.
Closure	Summarize key points: Emphasize the importance of the Banker's algorithm in ensuring safe resource allocation and avoiding deadlocks. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 6: Process Synchronization, pp. 370-380.** **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 2: Processes and Threads, pp. 200-210.**
	□ **Reflective Questions**: What is the purpose of the Banker's algorithm? How does it ensure that a system remains in a safe state? □ **Assignment**: Write a program that implements the Banker's algorithm. Provide detailed comments explaining each part of the code and how it ensures safe resource allocation

Lesson Plan No. 4.1	**Course Name: Operating System**	**Course No.:MCA-101**

Objectives	**Introduction to Memory Management** At the end of the lesson the student shall be able to: a. Understand the basic concepts and functions of memory management. b. Explain why memory management is essential in an operating system. c. Identify different memory management techniques. d. Describe the basic memory management in OS.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on memory management: c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is memory management in an OS? ○ Why is it important to manage memory efficiently? **Link:** https://youtu.be/TAk822Wz4x4?si=t03F8YtoxU_QxU3S https://www.youtube.com/watch?v=fkGCLIQx1MI 2. **Introduction to Management Memory:** ○ Define Memory management: "Memory management is the functionality of an operating system which handles or manages primary memory." ○ Show a simple diagram of how memory is allocated to processes. **Development (30 minutes)** 3. **Basic Concepts of Memory Management (10 minutes):** ○ Explain RAM, cache, and virtual memory. **Video Reference:** Introduction to Memory Management **Link:** https://youtu.be/TAk822Wz4x4?si=t03F8YtoxU_QxU3S ○ Discuss key milestones: Discuss how the OS tracks memory usage. ○ **Real-time Example:** Demonstrate with a simple application how memory allocation and deallocation work. 4. **Memory Management Techniques (10 minutes):**

	○ Describe the importance of memory management in operating systems. ○ Discuss the role of memory management unit (MMU). ○ Explain the differences and relationship between logical and physical addresses. ○ **Real-time Example:** Use an application that runs multiple processes to show memory allocation. ○ Discuss different techniques. ▪ Fixed and dynamic partitioning. ▪ Paging and segmentation. ○ **Web Reference:** Memory Management Overview 5. **Memory Management (10 minutes):** ○ Explain RAM, cache, and virtual memory. ○ Discuss how the OS tracks memory usage. ○ **Diagram Reference:** Show how memory is allocated to processes. **Exercise (5 minutes)** 6. **Activity:** Pair students and ask them to explain how memory is managed when running multiple applications. 7. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Definition and functions of Memory management. • Importance of Memory management. • Basic concepts and techniques • Basic Memory Management Techniques **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 8:** Memory Management, pp. 331-370. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3:** Memory Management, pp. 105-147
Evaluation	• **Reflective Questions:** Discuss why efficient memory management is critical. **Assignment:** Write a short note on different memory management techniques.

Lesson Plan No. 4.2	**Course Name: Operating System**	**Course No.:MCA-101**

Objectives	**Logical and Physical Address Space** At the end of the lesson the student shall be able to: a. Differentiate between logical and physical addresses. b. Understand the concept of address translation. c. Explain how the OS handles address space.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on address translation c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What do you understand by logical and physical addresses? ○ Why do we need address translation? **Link:** (https://www.youtube.com/watch?v=1DoIrf9Evf8) 2. **Introduction to Logical and physical address:** ○ Define logical and physical addresses: ○ Logical Address: "An address generated by the CPU." ○ Physical Address: "An address seen by the memory unit." ○ Show a simple example of address translation **Development (30 minutes)** 3. **Basic Concepts of Address Space Concepts (10 minutes):** ○ Explain logical and physical addresses. ○ Discuss how logical addresses are translated to physical addresses. **Video Reference:** Address Translation Techniques **Link**: https://youtu.be/TAk822Wz4x4?si=t03F8YtoxU_QxU3S ○ **Real-time Example:** Demonstrate with an example how an address is translated. 4. **Differences and Relationship (10 minutes):** ○ Explain the logical address (generated by CPU) and physical address (seen by memory unit). ○ Discuss their relationship and how they interact. ○ **Real-time Example:** Use an application that runs multiple processes to show memory allocation and address translation.

	o Discuss different techniques. ▪ Fixed and dynamic partitioning. ▪ Paging and segmentation. o **Web Reference:** Address Translation Techniques 5. **Techniques (10 minutes):** o Explain Fixed and Dynamic Partitioning. o Discuss Paging and Segmentation o Describe these techniques and how they help in memory management. o **Diagram Reference:** Show a simple example of address translation. **Exercise (5 minutes)** 6. **Activity:** Ask students to map a given set of logical addresses to physical addresses. 7. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Difference between logical and physical addresses. • Importance of Memory management. • Techniques for address translation **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 8:** Memory Management, pp. 358-360. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3:** Memory Management, pp. 183-185
Evaluation	• **Reflective Questions:** Discuss the importance of address translation. **Assignment:** Write a note on how the OS handles address space translation.

Lesson Plan No. 4.3	Course Name: Operating System	Course No.:MCA-101

Objectives	**Contiguous and Non-Contiguous Memory Allocation** At the end of the lesson the student shall be able to: a. Understand the difference between contiguous and non-contiguous memory allocation. b. Explain the advantages and disadvantages of each allocation method. c. Describe how the OS implements these allocation techniques.
Teaching Aids (if any)	a. Slides with diagrams and examples b. Video clips on memory allocation methods c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What are contiguous and non-contiguous memory allocations? ○ Can you think of any examples where these might be used? ○ Explain in detail Memory Allocation Methods. **Link:** https://www.youtube.com/watch?v=7-2yDNlBDDY 2. **Introduction to contiguous and non-contiguous memory allocations:** ○ Define contiguous and non-contiguous memory allocations? ○ Contiguous Memory Allocation: "Allocating consecutive blocks of memory." ○ Non-Contiguous Memory Allocation: "Allocating scattered memory blocks." ○ Show a simple example of both methods. **Development (30 minutes)** 3. **Basic Concepts of Memory Allocation Methods (10 minutes):** ○ Explain contiguous and non-contiguous memory allocations. ○ Discuss advantages and disadvantages of each method. ○ **Video Reference:** contiguous and non-contiguous memory allocations. **Link**: https://youtu.be/TAk822Wz4x4?si=t03F8YtoxU_QxU3S 4. **Real-time Example:** Demonstrate with an application how each allocation method works. 5. **Differences and Relationship (10 minutes):**

	○ Explain contiguous and non-contiguous allocation. ○ Discuss their relationship and how they interact. ○ **Real-time Example:** Demonstrate with an application how each allocation method works. ○ Discuss different techniques. ▪ Fixed and dynamic partitioning. ▪ Paging and segmentation. ○ **Web Reference:** Memory Allocation Techniques 6. **Techniques (10 minutes):** ○ Explain Fixed and Dynamic Partitioning. ○ Discuss Paging and Segmentation ○ Describe these techniques and how they help in memory management. ○ **Diagram Reference:** Show an application that uses both allocation methods and discuss its performance. **Exercise (5 minutes)** 7. **Activity:** Pair students and ask them to explain the advantages and disadvantages of each method. 8. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Difference between contiguous and non-contiguous allocation. • Advantages and disadvantages of each method. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 9:** Memory Management Strategies, pp. 372-382. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3:** Memory Management, pp. 155-158
Evaluation	• **Reflective Questions:** Discuss why an OS might choose one method over another. **Assignment:** Write a note on the implementation of these techniques in modern operating systems.

Lesson Plan No. 4.4	Course Name: Operating System	Course No.:MCA-101

Objectives	**Paging and Structure of Page Table** At the end of the lesson the student shall be able to: a. Understand the concept of paging. b. Explain the structure of a page table. c. Describe how the OS uses paging to manage memory.
Teaching Aids (if any)	a. Slides with diagrams and examples b. Video clips on paging c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is paging in an operating system? ○ Why do we need a page table? **Link:** (https://www.youtube.com/watch?v=1DoIrf9Evf8) 2. **Introduction to Paging:** ○ Define Paging: Paging: "A memory management scheme that eliminates the need for contiguous allocation of physical memory." ○ Show a simple example of paging and a page table. **Development (30 minutes)** 3. **Basic Concepts of Paging (10 minutes):** ○ Explain the concept of paging. ○ Discuss the role of the page table. **Video Reference:** Page Table Structure **Link**: https://youtu.be/TAk822Wz4x4?si=t03F8YtoxU_QxU3S ○ **Real-time Example:** Demonstrate with an example how paging works 4. **Page Table Structure (10 minutes):** ○ Explain the Single-level and multi-level page tables. ○ Discuss Inverted page tables. ○ **Real-time Example:** Show a program and trace its page table entries. ○ **Web Reference:** Paging and Page Table Structure 5. **Real- time Examples (10 minutes):** ○ Explain Single-level and multi-level page tables.

	○ Show a program and trace its page table entries. ○ **Diagram Reference:** Show a simple example of paging and a page table. **Exercise (5 minutes)** 6. **Activity:** Ask students to explain how a given logical address is translated using a page table. 7. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Concept of paging and its importance • Structure and function of page tables **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 8:** Memory Management, pp. 360-372. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3:** Memory Management, pp. 185-195
Evaluation	• **Reflective Questions:** Discuss the advantages of using paging. **Assignment:** Write a note on different types of page tables and their use cases.

Lesson Plan No. 4.5	Course Name: Operating System	Course No.:MCA-101

Objectives	**Segmentation and Demand Paged Memory Management** At the end of the lesson the student shall be able to: a. Understand the concept of segmentation. b. Explain the demand-paged memory management. c. Describe Demand Paged Memory Management.
Teaching Aids (if any)	a. Slides with diagrams and examples b. Video clips segmentation c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is segmentation in an operating system? ○ Why do we need a page table? ○ How does segmentation differ from paging? **Link:** https://archive.nptel.ac.in/courses/106/106/106106144/ 2. **Introduction to Segmentation:** ○ Define Segmentation: "Segmentation is a memory management scheme that supports the user view of memory." ○ Show a simple example of segmentation. **Development (30 minutes)** 3. **Basic Concepts of Segmentation (10 minutes):** ○ Explain the concept of segmentation. ○ Diagram showing segmentation. **Video Reference:** Segmentation. **Link**: https://www.youtube.com/watch?v=GskGMGQ_wlg ○ **Real-time Example:** Allocation of memory segments in a program. 4. **Demand Paged Memory Management (10 minutes):** ○ Explain the concept of demand paging. ○ Diagram showing demand paging. ○ **Real-time Example:** Describe the process of loading a page into memory on demand. ○ **Web Reference:** Demand Paging

	5. **Real- time Examples (10 minutes):** ○ Using segmentation in large applications like databases. ○ Show a program and trace its page table entries. ○ **Diagram Reference:** Show a simple example of segmentation **Exercise (5 minutes)** 6. **Activity:** Simulate a demand paging system in class using paper and markers. 7. Group discussion on page faults and recovery mechanisms.
Closure	Summarize key points: • Concept of segmentation and its importance • Structure and function of page tables **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 8:** Memory Management, pp. 411-418. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3:** Memory Management, pp. 195-210
Evaluation	• **Reflective Questions:** Discuss the advantages of using segmentation. **Assignment:** Emphasize the importance of understanding memory management.

Lesson Plan No. 4.6	**Course Name: Operating System**	**Course No.:MCA-101**

Objectives	**Page Replacement** At the end of the lesson the student shall be able to: a. Understand the need for page replacement in memory management. b. Learn different page replacement algorithms. c. Discuss the performance of various algorithms.
Teaching Aids (if any)	a. Slides with diagrams and examples b. Video clips on Page Replacement c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ Define page replacement in the context of demand paging. ○ Introduce the concept of page replacement and its necessity. ○ How does segmentation differ from paging? **Link:** https://archive.nptel.ac.in/courses/106/106/106106144/ 2. **Introduction to Page Replacement:** ○ Definition and need for page replacement. ○ Real-time example: Managing memory in a gaming application. **Development (30 minutes)** 3. **Basic Concepts of Page Replacement Algorithms (10 minutes):** ○ Explain the concept of Page Replacement Algorithms ○ Diagram showing Page Replacement Algorithms. **Video Reference:** Page Replacement Algorithms. **Link**: https://www.youtube.com/watch?v=GskGMGQ_wlg ○ **Real-time Example:** How different algorithms affect performance in file systems. 4. **Implementation and Challenges (10 minutes):** ○ Practical aspects of implementing page replacement algorithms. ○ Example: Page replacement in embedded systems. ○ **Real-time Example:** Managing memory in a gaming application. ○ **Web Reference:** Page Replacement 5. **Real- time Examples (10 minutes):**

	○ Comparison of page replacement algorithms using a simulation. ○ Show a program Page replacement in embedded systems. ○ **Diagram Reference:** Show a simple example of page replacement. **Exercise (5 minutes)** 6. **Activity:** Simulate a demand paging system in class using paper and markers. 7. Group discussion on algorithm performance in various scenarios.
Closure	Summarize key points: • Concept of page replacement and its importance • Structure and function of page replacement **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 9:** Memory Management, pp. 361-380. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 8:** Virtual Memory Management, pp. 300-320
Evaluation	• **Reflective Questions:** Discuss the advantages of using page replacement. **Assignment:** Emphasize the importance of understanding memory management.

Lesson Plan No. 4.7	Course Name: Operating System	Course No.: MCA-101

Objectives	**Allocation of Frames** At the end of the lesson the student shall be able to: a. Definition and purpose of frame allocation. b. Real-time example: Frame allocation in virtual memory systems. c. Discuss the Frame allocation.
Teaching Aids (if any)	a. Slides with diagrams and examples b. Video clips on Allocation of Frames c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ Define Frame Allocation. ○ How frame allocation affects performance. ○ Introduce the concept of Frame allocation in virtual memory systems. **Link:** https://archive.nptel.ac.in/courses/106/106/106106144/ 2. **Introduction to Frame Allocation:** ○ Definition and need for frame allocation. ○ Real-time example: Frame allocation in virtual memory systems. **Development (30 minutes)** 3. **Basic Concepts of Frame Allocation (10 minutes):** ○ Explain the concept of Frame Allocation. ○ Diagram showing Allocation strategies used in different types of applications. **Video Reference:** Frame Allocation **Link**: https://www.youtube.com/watch?v=GskGMGQ_wlg ○ **Real-time Example:** How frame allocation affects performance. 4. **Implementation and Challenges (10 minutes):** ○ Practical challenges in frame allocation. ○ Example: Frame allocation in embedded systems. ○ **Real-time Example:** Frame allocation in virtual memory systems. ○ **Web Reference:** Frame Allocation

	5. **Real- time Examples (10 minutes):** ○ How frame allocation affects performance. ○ Impact of frame allocation on a database system. ○ **Diagram Reference:** Show a simple example of frame allocation. **Exercise (5 minutes)** 6. **Activity:** Simulate a demand paging system in class using paper and markers. 7. Group discussion on algorithm performance in various scenarios.
Closure	Summarize key points: • Concept of frame allocation and its importance. • Structure and function of frame allocation. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 9:** Memory Management, pp. 381-400. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 8:** Virtual Memory Management, pp. 340-360
Evaluation	• **Reflective Questions:** Discuss the advantages of using frame allocation. **Assignment:** Emphasize the importance of understanding memory management.

Lesson Plan No. 4.8	**Course Name: Operating System**	**Course No.:MCA-101**

Objectives	**Thrashing & Swapping** At the end of the lesson the student shall be able to: a. Understand the concepts of thrashing, swapping. b. Explain the causes and effects of thrashing in an operating system. c. Describe the mechanisms of swapping and how it helps manage memory.
Teaching Aids (if any)	a. Slides with diagrams and examples b. Video clips on thrashing, swapping c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What do you know about thrashing in an OS? ○ How does swapping help in memory management? ○ Introduce the concept of Thrashing. **Link:** https://youtu.be/IyWaK8pbN6A?si=XeqpG9la-kl2tEB- 2. **Introduction to Thrashing & Swapping:** ○ Definition Thrashing: "Thrashing occurs when a system spends more time swapping pages in and out of memory than executing instructions." ○ Discuss the causes of thrashing: High degree of multiprogramming, insufficient RAM, and high page fault rate. ○ Explain the effects of thrashing: Decreased system performance, increased CPU overhead. **Development (30 minutes)** 3. **Basic Concepts of Swapping (10 minutes):** ○ Define swapping: "Swapping is the process of moving processes between main memory and disk to ensure that each process gets the necessary resources." ○ Discuss how swapping helps manage memory and increases the degree of multiprogramming. ○ Diagram Show a simple demonstration of swapping using an OS simulator. **Video Reference:** Swapping in Operating System

	Link: https://youtu.be/mJqGKsJncco?si=M_K8r5D3Jj-1UyYp ○ **Real-time Example:** Demonstrate with a system simulation showing high CPU usage due to excessive paging. 4. **Implementation and Challenges (10 minutes):** ○ Explain the effects of thrashing: Decreased system performance, increased CPU overhead. ○ **Real-time Example:** Show a simple demonstration of swapping using an OS simulator. ○ **Web Reference:** Swapping 5. **Real- time Examples (10 minutes):** ○ Explain how overlays are used to manage memory in constrained environments. ○ Role of overlays in memory management ○ **Diagram Reference:** Show a simple example of swapping. **Exercise (5 minutes)** 6. **Activity** Pair students and ask them to explain how cache memory improves the performance of a simple application. 7. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Definition and effects of thrashing • Mechanism and importance of swapping • Role of overlays in memory management **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 8:** Memory Management, pp. 331-370. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3:** Virtual Memory Management, pp. 105-147
Evaluation	• **Reflective Questions:** Discuss why efficient memory management is critical. **Assignment:** Write a short note on how cache memory impacts system performance.

Lesson Plan No. 4.9	Course Name: Operating System	Course No.: MCA-101

Objectives	**Overlays, and Cache Memory** At the end of the lesson the student shall be able to: a. Understand the role of overlays in memory management. b. Explain the importance and functioning of cache memory. c. Describe the mechanisms of cache memory and how it helps manage memory.
Teaching Aids (if any)	a. Slides with diagrams and examples b. Video clips on Overlays & Cache Memory c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ Have you heard about overlays in memory management? ○ How does Cache memory help in memory management? ○ What is cache memory and why is it important? **Link:** https://youtu.be/Quj-Goz4VMA?si=PTuPjGUR8vPzAqQH 2. **Introduction to Overlays & Cache Memory:** ○ Define overlays: "Overlays are a technique used to fit large programs into smaller memory by loading only the necessary sections into memory." ○ Discuss the historical context and the importance of overlays in early computing. ○ Explain how overlays are used to manage memory in constrained environments. **Development (30 minutes)** 3. **Basic Concepts Cache Memory (10 minutes):** ○ Define cache memory: "Cache memory is a small, high-speed memory located close to the CPU to reduce the average time to access data from the main memory." ○ Discuss the importance of cache memory in improving system performance. ○ Diagram Show a diagram explaining the cache hierarchy. **Video Reference:** Cache Memory in Operating System **Link:** https://youtu.be/y5rKivsKyZA?si=etZ-MfxMqYoaEZ7l

	○ **Real-time Example:** Demonstrate how cache memory speeds up data access using a simulation. 4. **Implementation and Challenges (10 minutes):** ○ Explain how overlays are used to manage memory in constrained environments. ○ **Real-time Example:** Show a simple demonstration of swapping using an OS simulator. ○ **Web Reference:** Cache memory 5. **Real- time Examples (10 minutes):** ○ Explain how cache memory is used to manage memory in constrained environments. ○ Role of cache in memory management ○ **Diagram Reference:** Show a simple example of cache memory. **Exercise (5 minutes)** 6. **Activity** Pair students and ask them to explain how cache memory improves the performance of a simple application. 7. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Definition and effects of overlays. • Mechanism and importance of cache memory. • Role of overlays in memory management **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 8:** Memory Management, pp. 348-350. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 3:** Virtual Memory Management, pp. 105-147
Evaluation	• **Reflective Questions:** Discuss why efficient memory management is critical. **Assignment:** Write a short note on how cache memory impacts system performance.

Lesson Plan No. 5.1	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Introduction to File Systems** At the end of the lesson the student shall be able to: a. Understand the basic concepts and functions of file systems. b. Describe file concepts, structures, types, and access methods. c. Explore directory structures.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on file systems. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is a file system? ○ Why is it essential for an operating system to manage files? ○ Example: Ask students to name different types of file systems they have encountered (e.g., NTFS, FAT32, ext4). **Link: https://youtu.be/0LtuQhNFFe0?si=gV5ZD1fkV_fAhh5S** 2. **Introduction to File Concept and Structure:** ○ Define what a file is: "A file is a named collection of related information that is recorded on secondary storage." ○ **Real-time Example:** Discuss different file types such as text files, binary files, executable files, etc. **Development (30 minutes)** 3. **Basic Concepts of File Access Methods (10 minutes):** ○ Explain Sequential access: accessing files in a sequential order. ○ Direct access: accessing files at a specific location. **Video Reference:** Introduction to File Access Methods **Link: https://youtu.be/volw53pUD3Y?si=O8trSTVpw-SLuUyO** ○ Discuss Key Concept: Discuss how the OS tracks disk usage. ○ **Real-time Example:** Compare reading a book (sequential access) vs. looking up a word in a dictionary (direct access). 4. **Directory Structure (10 minutes):** ○ Describe the Single-level directory, two-level directory, tree-structured directory, acyclic-graph directory, general graph directory. ○ Discuss different file types such as text files, binary files, executable files, etc.

	○ Compare the structure of directories to the organization of folders on a computer. ○ **Real-time Example:** Compare the structure of directories to the organization of folders on a computer. 5. **File Concept and Structure (10 minutes):** ○ Explain File Concept and File Structure. ○ Discuss different file types such as text files, binary files, executable files, etc. **Exercise (5 minutes)** **Think -Pair Activity:** Pair students and ask them to classify various file systems they know into different types and access methods.
Closure	Summarize key points: ● The concept of files and their structures. ● Various file types and access methods. ● Different directory structures. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 11:** File-System Implementation, pp. 463-508. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4:** File System, pp. 131-174
Evaluation	**Reflective Questions:** ● Discuss why managing file systems efficiently is critical.? ● Discuss about the different file access methods and directory structures?

Lesson Plan No. 5.2	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Allocation Methods** At the end of the lesson the student shall be able to: a. Understand different file allocation methods. b. Explore contiguous, linked, and indexed allocation methods. c. Explore allocation methods.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on allocation methods. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is file allocation? ○ Why do different allocation methods exist? **Link: https://www.youtube.com/watch?v=J6wVO4pvUCw** 2. **Introduction to Allocation Methods:** ○ Define Contiguous Allocation: "Files are stored in contiguous blocks." ○ **Real-time Example:** Compare to a bookshelf where books are placed next to each other without gaps. **Development (30 minutes)** 3. **Basic Concepts of Contiguous Allocation (10 minutes):** ○ Explain files are stored in contiguous blocks. ○ Direct access: accessing files at a specific location. **Video Reference:** Contiguous Allocation **Link:** https://www.youtube.com/watch?v=JVHQ8y1yzV0 ○ Discuss contiguous allocation. Discuss how the OS tracks memory usage. ○ **Real-time Example:** Compare to a bookshelf where books are placed next to each other without gaps. 4. **Linked Allocation (10 minutes):** ○ Describe the Linked Allocation. ○ Discuss different allocation methods. ○ Explain each file is a linked list of disk blocks. ○ **Real-time Example:** Compare to a treasure hunt where each clue leads to the next.

	○ **Web Reference:** https://www.geeksforgeeks.org/file-allocation-methods/ ○ 5. **Indexed Allocation (10 minutes):** ○ Explain Indexed Allocation. ○ Compare to an index in a book that directs you to the correct page. ○ **Diagram Reference:** Discuss different allocation methods. **Exercise (5 minutes)** 6. **Activity:** Pair students and ask them to draw diagrams representing each allocation method. 7. Discuss their classifications and correct any misconceptions.
Closure	Summarize key points: • Different file allocation methods: contiguous, linked, indexed. • Advantages and disadvantages of each method. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 11:** File-System Implementation, pp. 509-537. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4:** File System, pp. 175-212.
Evaluation	• **Reflective Questions:** Discuss the pros and cons of each allocation method. **Assignment:** Write a short note comparing the three allocation methods.

Lesson Plan No. 5.3	Course Name: Operating System	Course No.: MCA-101

Objectives	**Free- Space management** At the end of the lesson the student shall be able to: a. Understand the different methods of free-space management. b. Explain how bit vector, linked list, and grouping methods work. c. Compare the efficiency and use cases of each method.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on demonstrating free-space management techniques. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is free-space management in an operating system? ○ Why is it important to manage free space efficiently? **Link: https://www.youtube.com/watch?v=Kfrv8QMfznU** 2. **Introduction to free-space management:** ○ Define free-space management: "Free-space management is the process of managing the available disk space in a storage system." ○ Show a simple diagram of free-space management. ○ **Real-time Example:** Use a chalkboard to illustrate how blocks are linked. **Development (30 minutes)** 3. **Basic Concepts of Bit Vector (10 minutes):** ○ Explain what a bit vector is and how it is used to manage free space. ○ Show a diagram of a bit vector. **Video Reference:** Bit Vector **Link:** https://www.youtube.com/watch?v=Kfrv8QMfznU ○ **Real-time Example:** Demonstrate a bit vector with an example of disk blocks being allocated and deallocated. 4. **Linked List (10 minutes):** ○ Explain the concept of linked list free-space management. ○ Show a diagram of how a linked list can manage free space. ○ **Real-time Example:** Use a chalkboard to illustrate how blocks are linked.

	○ **Web Reference:** https://www.youtube.com/watch?v=irGdM3iIS54 5. **Grouping (10 minutes):** ○ Explain the grouping method of free-space management. ○ Show a diagram of how grouping works. ○ Real-time Example: Illustrate grouping with an example on the whiteboard. **Exercise (5 minutes)** 1. **Activity:** Pair students and ask them to explain which method they find most efficient and why. 2. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Different methods of free-space management. • Efficiency and use cases of each method. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 11:** File-System Implementation, pp. 459-463. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4:** File System, pp. 245-252.
Evaluation	• **Reflective Questions:** Discuss why choosing the right free-space management method is critical. **Assignment:** Write a short note comparing bit vector, linked list, and grouping methods.

Lesson Plan No. 5.4	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Disk Structure, Disk Scheduling** At the end of the lesson the student shall be able to: a. Understand the structure of disks and how data is stored. b. Learn different disk scheduling algorithms. c. Compare the efficiency and scenarios of each disk scheduling method.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on demonstrating disk structure and scheduling. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is the basic structure of a disk in an operating system? ○ Why is disk scheduling important? **Link: https://www.youtube.com/watch?v=Iqs5G-Uz9gg** 2. **Introduction to disk structure:** ○ Define disk structure: "Disk structure refers to the physical and logical organization of data on a disk." ○ Show a simple diagram of a disk structure. ○ **Real-time Example:** Demonstrate data storage on a disk. **Development (30 minutes)** 3. **Basic Concepts of Disk Structure (10 minutes):** ○ Explain the basic components of a disk (platters, tracks, sectors). ○ Show a diagram of a disk structure. **Video Reference:** Disk structure **Link:** https://youtu.be/s9G7K8VCzk?si=52vH98e40xDGSZq4 ○ **Real-time Example:** Demonstrate data storage on a disk. 4. **Disk Scheduling Algorithms (10 minutes):** ○ Explain the First-Come, First-Served (FCFS) scheduling algorithm. ○ Show a diagram and real-time example.

	○ **Real-time Example:** Show a diagram and real-time example. ○ **Web Reference:** https://www.youtube.com/watch?v=P_dA8VGJjA8 5. **LOOK and C-LOOK (10 minutes):** ○ Explain the LOOK and Circular LOOK (C-LOOK) scheduling algorithms. ○ Show a diagram of how grouping works. **Exercise (5 minutes)** 1. **Activity:** Pair students and ask them to choose a disk scheduling algorithm and explain its advantages and disadvantages. 2. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Structure of disks. • Different disk scheduling algorithms. • Efficiency and use cases of each method. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 11:** File-System Implementation, pp. 464-470. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4:** File System, pp. 431-440.
Evaluation	• **Reflective Questions:** Discuss why choosing the right disk scheduling method is critical. **Assignment:** Write a short note comparing different disk scheduling algorithms.

Lesson Plan No. 5.5	Course Name: Operating System	Course No.: MCA-101

Objectives	**Disk Management, Disk Formatting** At the end of the lesson the student shall be able to: a. Understand the concepts of disk management and formatting. b. Learn the steps involved in disk formatting. c. Compare different disk management techniques.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on demonstrating disk management and formatting. c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is disk management in an operating system? ○ Why is disk formatting important? **Link: https://www.youtube.com/watch?v=NiefQwuNtjg** 2. **Introduction to disk management:** ○ Define disk management: "Disk management involves the tasks of managing and maintaining disks and the data stored on them." ○ Show a simple diagram of a disk management. ○ **Real-time Example:** Demonstrate data storage on a disk. **Development (30 minutes)** 3. **Basic Concepts of Disk Management (10 minutes):** ○ Explain the basic concepts of disk management. ○ Show a diagram of disk partitions and volumes. **Video Reference:** Disk management **Link:** https://www.youtube.com/watch?v=vU8ZLyJIVs0 ○ **Real-time Example:** Demonstrate disk management using disk management tools (e.g., Windows Disk Management). 4. **Disk Formatting (20 minutes):** ○ Explain the steps involved in disk formatting. ○ Show a diagram the formatting process. ○ **Real-time Example:** Demonstrate disk formatting using formatting tools.

	○ **Web Reference:** https://www.youtube.com/watch?v=P_dA8VGJjA8 **Exercise (5 minutes)** 1. **Activity:** Pair students and ask them to explain the importance of disk management and formatting. 2. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Concepts of disk management. • Steps involved in disk formatting. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 11:** File-System Implementation, pp. 470-473. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4:** File System, pp. 440-445.
Evaluation	• **Reflective Questions:** Discuss why disk management and formatting are critical for system performance. **Assignment:** Write a short note on the steps involved in disk formatting.

Lesson Plan No. 5.6	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**Swap Space Management** At the end of the lesson the student shall be able to: a. Understand the concept of swap space in an operating system. b. Learn how swap space is managed. c. Compare the efficiency and use cases of swap space management techniques.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on demonstrating swap space management c. Chalkboard/Whiteboard
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is swap space in an operating system? ○ Why is swap space important? **Link:** https://www.youtube.com/watch?v=MAy4-3ppj6Y 2. **Introduction to Swap Space Management:** ○ Define swap space: "Swap space is a portion of a hard disk used as an extension of RAM." ○ Show a simple diagram of a swap space. ○ **Real-time Example:** Demonstrate swap space management using system settings. **Development (30 minutes)** 3. **Basic Concepts of Swap Space Management (10 minutes):** ○ Explain the concept of swap space. ○ Show a diagram of how swap space is used. **Video Reference:** Swap Space Management **Link:** https://www.youtube.com/watch?v=vU8ZLyJIVs0 ○ **Real-time Example:** Demonstrate swap space management using system settings. 4. **Techniques of Swap Space Management (20 minutes):** ○ Explain different swap space management techniques. ○ Show diagrams and real-time examples. ○ **Real-time Example:** Demonstrate disk formatting using formatting tools.

	○ **Web Reference:** **https://www.youtube.com/watch?v=P_dA8VGJjA8** **Exercise (5 minutes)** 1. **Activity:** Pair students and ask them to explain the importance of swap space. 2. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Concepts of swap space management. • Different techniques of swap space management. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 11:** File-System Implementation, pp. 473-480. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4:** File System, pp. 445-455.
Evaluation	• **Reflective Questions:** Discuss why disk management and formatting are critical for system performance. **Assignment:** Write a short note on the steps involved in disk formatting.

Lesson Plan No. 5.7	**Course Name: Operating System**	**Course No.: MCA-101**

Objectives	**RAID Structure** At the end of the lesson the student shall be able to: a. Understand the concept of RAID (Redundant Array of Independent Disks). b. Explain the different levels of RAID and their characteristics. c. Compare the advantages and disadvantages of various RAID levels. d. Identify the appropriate use cases for different RAID configurations.
Teaching Aids (if any)	a. Slides with diagrams and definitions b. Video clips on RAID structures c. Chalkboard/Whiteboard d. Computer with RAID simulation software (if available)
Teaching Development	**Introduction (5 minutes)** 1. **Pre-Discussion Questions:** ○ What is RAID in the context of an operating system? ○ Why is RAID used in data storage systems? **Link: https://www.youtube.com/watch?v=MAy4-3ppj6Y** 2. **Introduction to RAID Structure :** ○ Define RAID: "RAID (Redundant Array of Independent Disks) is a data storage virtualization technology that combines multiple physical disk drive components into one or more logical units for data redundancy, performance improvement, or both." ○ Show a simple diagram of a RAID setup. ○ **Real-time Example:** Demonstrate swap space management using system settings. Use a whiteboard to demonstrate how data is divided and written across multiple disks in RAID 0. **Development (30 minutes)** 3. **Basic Concepts of RAID 1 (10 minutes):** ○ Explain RAID 1, its configuration, and its purpose (data redundancy). ○ Show a diagram of data mirroring across two disks. **Video Reference:** RAID Structure **Link:** https://www.youtube.com/watch?v=vU8ZLyJIVs0

	○ **Real-time Example:** Illustrate how parity is calculated and stored using an example on the chalkboard. 4. **Techniques of RAID 5, RAID 6 (20 minutes):** ○ Explain RAID 6, its configuration, and its purpose (improved redundancy with double parity). ○ Show diagrams and real-time examples. ○ **Real-time Example:** Demonstrate disk formatting using formatting tools. ○ **Web Reference:** **https://www.youtube.com/watch?v=P_dA8VGJjA8** **Exercise (5 minutes)** 1. **Activity:** Pair students and ask them to explain the importance of RAID. 2. Discuss their answers and correct any misconceptions.
Closure	Summarize key points: • Concept and purpose of RAID. • Different techniques of swap space management. **Suggested Readings** **Book 1:** "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. **Chapter 11:** File-System Implementation, pp. 473-480. **Book 2:** "Modern Operating Systems" by Andrew S. Tanenbaum. **Chapter 4:** File System, pp. 445-455.
Evaluation	• **Reflective Questions:** Discuss why disk management and formatting are critical for system performance. **Assignment:** Write a short note on the steps involved in disk formatting.