Third Year Even Semester  
Operating Systems

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Resources:**

1. [Modern Operating Systems (4th Edition)](https://www.pdfdrive.com/modern-operating-systems-4th-edition-e175928348.html) - Tanenbaum
2. [Operating System Concepts](https://www.pdfdrive.com/operating-system-concepts-e158325105.html) (10th edition) - Silberschatz
3. [William Stallings-Operating System](https://www.pdfdrive.com/william-stallings-operating-system-e19674757.html)
4. <https://classroom.google.com/c/NTIzNDgwNzc2NDJa>
5. <https://classroom.google.com/c/MTA5MTIyNjQ3NjM2>
6. [Relocation register | Article about relocation register by The Free Dictionary](https://encyclopedia2.thefreedictionary.com/relocation+register)
7. [Kernel (operating system)](https://en.wikipedia.org/wiki/Kernel_(operating_system))
8. [Abraham Silberschatz-Operating System Concepts](https://www.pdfdrive.com/abraham-silberschatz-operating-system-concepts-e45162438.html) - 9th edition
9. [What is Memory Management | static and dynamic loading in OS](http://easyconcept.in/what-is-memory-management-static-and-dynamic-loading-in-os/)
10. [University - Google Drive](https://drive.google.com/drive/folders/1kpmEwcuAcUSkSqxyM8fBIbXYtwFFneEI?usp=sharing)

**Introduction to Operating System**

**Operating System Concepts, Its role in Computer Systems**

**What is an operating system?[1] – (1.1)**

* 1.1.1: The Operating System as an Extended Machine
* 1.1.2: The Operating System as a Resource Manager

**What Operating Systems Do**[**2] – (1.1)**

* 1.1.1: User View
* 1.1.2: System View
* 1.1.3: Defining Operating Systems

**Kernel[7]:** The kernel is a computer program at the core of a computer's operating system with complete control over everything in the system. It is the "portion of the operating system code that is always resident in memory". It facilitates interactions between hardware and software components. On most systems, it is one of the first programs loaded on startup (after the bootloader). It handles the rest of startup as well as input/output requests from software, translating them into data-processing instructions for the central processing unit. It handles memory and peripherals like keyboards, monitors, printers, and speakers.

**Operating System Concepts[1] - (1.5)**

* 1.5.1: Processes
* 1.5.2: Address Spaces
* 1.5.3: Files
* 1.5.5: Protection

**Computer System Structure**

**Computer Hardware Review[1] - (1.3)**

* 1.3.6: Booting The Computer

**Building and Booting an Operating System[2] - (2.9)**

* 2.9.2: System Boot

**Fundamentals of Different types of Computer System**

**History of Operating Systems[1] – (1.2)**

**The Operating System Zoo[1] - (1.4)**

**Operating System Structure and Operation, Protection and Security**

**Operating-System Operations[2] - (1.4)**

* 1.4.2: Dual-Mode and Multimode Operation

**Resource Management[2] - (1.5)**

* 1.5.1: Process Management
* 1.5.2: Memory Management
* 1.5.3:File-System Management

**Security and Protection[2] - (1.6)**

**System Calls[1] - (1.6)**

* 1.6.1: System Calls for Process Management
* 1.6.2: System Calls for File Management

**Operating-System Services[2] - (2.1)**

**System Calls[2] - (2.3)**

* 2.3.3: Types of System Calls
  + 2.3.3.1: Process Control
  + 2.3.3.2: File Management
  + 2.3.3.6: Protection

**System Services[2] - (2.4)**

**Operating System Structure[1] - (1.7)**

* 1.7.1: Monolithic Systems
* 1.7.2: Layered Systems
* 1.7.3: Microkernels
* 1.7.5: Virtual Machines

**Operating-System Structure[2] - (2.8)**

* 2.8.1: Monolithic Structure
* 2.8.2: Layered Approach
* 2.8.3: Microkernels
* 2.8.5: Hybrid Systems

**Distributed Operating System**

* [Distributed Operating Systems - Introduction](https://www.ics.uci.edu/~cs230/lectures/DistributedOSintro.pdf)
* [Distributed operating system - Wikipedia](https://en.wikipedia.org/wiki/Distributed_operating_system)
* [Distributed Operating System - javatpoint](https://www.javatpoint.com/distributed-operating-system)

**Process Management**

**Process Concept, Model and Implementation, Process State**

**Process Concept[2] - (3.1)**

* 3.1.1: The Process
* 3.1.2: Process State
* 3.1.3: Process Control Block
* 3.1.4: Thread

**Processes[1] - (2.1)**

* 2.1.1: The Process Model
* 2.1.2: Process Creation
* 2.1.3: Process Termination
* 2.1.4: Process Hierarchies
* 2.1.5: Process States
* 2.1.6: Implementation of Processes
* 2.1.7: Modeling Multiprogramming

**Process Table**

* [Process Table](https://www.cs.unc.edu/~dewan/242/s07/notes/pm/node3.html)

**Process Address Space**

* [W4118: Process and Address Space](https://www.cs.columbia.edu/~junfeng/12sp-w4118/lectures/l04-proc.pdf)
* [Process Address Space in Operating System (OS)](https://prepinsta.com/operating-systems/process-address-space/)

**Threads[1] - (2.2)**

* 2.2.1: Thread Usage
* 2.2.2: The Classical Thread Model
* 2.2.3: POSIX Threads

**Overview[2] - (4.1)**

* 4.1.1: Motivation
* 4.1.2: Benefits

**Process Scheduling**

**Process Scheduling[2] - (3.2)**

* 3.2.1: Scheduling Queues
* 3.2.2: CPU Scheduling
* 3.2.3: Context Switch

**Scheduling[1] - (2.4)**

* 2.4.1: Introduction to Scheduling
* 2.4.2: Scheduling in Batch Systems

**Inter-Process Communication (IPC)**

**Operations on Processes[2] - (3.3)**

* 3.3.1: Process Creation
* 3.3.2: Process Termination

**Interprocess Communication[2] - (3.4)**

**IPC in Shared-Memory Systems[2] - (3.5)**

**IPC in Message-Passing Systems[2] - (3.6)**

**Examples of IPC Systems[2] - (3.7)**

* 3.7.1:POSIX Shared Memory
* 3.7.4: Pipes
  + 3.7.4.1: Ordinary Pipes
  + [A simple program on pipe](https://github.com/rmShoeb/university-courses/blob/master/3-2/CSE%203202/lab%2005%20-%20online/f-two.cpp)

**Interprocess Communication[1] - (2.3)**

* 2.3.1: Race Conditions
* 2.3.2: Critical Regions
* 2.3.3: Mutual Exclusion with Busy Waiting
* 2.3.4: Sleep and Wakeup
* 2.3.5: Semaphores
* 2.3.6: Mutexes
* 2.3.7: Monitor
* 2.3.8: Message Passing

**Synchronization Tools[2] - (Chapter - 6)**

* 6.1: Background
* 6.2: The Critical-Section Problem
* 6.3: Peterson’s Solution
* 6.5: Mutex Locks
* 6.6: Semaphores
  + 6.6.1: Semaphore Usage
  + 6.6.2: Semaphore Implementation
* 6.7: Monitors
  + 6.7.1: Monitor Usage
  + 6.7.2: Implementing a Monitor using Semaphore
  + 6.7.3: Resuming Processes within a Monitor

**Classic Problem of Synchronization[2] - (7.1)**

* 7.1.3: The Dining-Philosophers Problem

**Classical IPC Problems[1] - (2.5)**

* 2.5.1: The Dining Philosophers Problem

**Multiprocessing and Timesharing**

**Interaction between Process and Operating System**

**CPU Scheduling**

* [What is Burst time, Arrival time, Exit time, Response time, Waiting time, Turnaround time, and Throughput?](https://afteracademy.com/blog/what-is-burst-arrival-exit-response-waiting-turnaround-time-and-throughput)
* [Process Scheduling and Process Scheduler](https://tutorialwing.com/process-scheduling-and-process-schedular/)

**Scheduling[1] - (2.4)**

* 2.4.1: Introduction to Scheduling
* 2.4.2: Scheduling in Batch Systems
* 2.4.3: Scheduling in Interactive Systems

**Basic Concepts[2] - (5.1)**

* 5.1.1: CPU-I/O Burst Cycle
* 5.1.2: CPU Scheduler
* 5.1.3: Preemptive and Nonpreemptive Scheduling
* 5.1.4: Dispatcher

**Scheduling Criteria[2] - (5.2)**

**Scheduling Algorithms[2] - (5.3)**

* 5.3.1: First-Come, First-Served Scheduling
* 5.3.2: Shortest-Job-First Scheduling
* 5.3.3: Round-Robin Scheduling
* 5.3.4: Priority Scheduling
* 5.3.6: Multilevel Feedback Queue Scheduling

**Memory Management**

**Why Do we need Memory Management?[9]:** Using memory management, the operating system analyzes the computer’s memory to check how much free space is left in it and how a new program and its data should be stored in those free spaces. Through this, the operating system ensures that the programs that have completed their execution, free up the space occupied by them in the memory so that the free space remains in the memory. Using memory management, it is ensured that the data of each computer program is stored by dividing it into separate memory blocks, so that the data of one program does not create any problem by unnecessarily affecting the data of other programs. It helps the program executing in the CPU of the computer to establish a relationship with the data stored in the memory.

For a program to be executed by the computer’s CPU. It is necessary to store both that program and the data used by it by bringing it to the computer’s primary memory (i.e. RAM). But it is not necessary that there is always free space in RAM to store a program. Many times the memory space of RAM is misused for various reasons such as fragmentation etc. In multiprogramming, many programs are being executed simultaneously. So it is very common for the memory space of RAM to run out completely. To avoid this type of situation, operating systems use techniques like swapping, makes space for new programs in primary memory. All this work also comes under memory management.

**Advantages of memory management in operating systems[9]:**

* This helps in using them in a better way without wasting the space of memory.
* It helps in multiprogramming the CPU by storing as many programs and their data as possible in the primary memory.
* Due to efficient memory management, the speed of accessing the data stored in the memory by the CPU increases. So that the computer improves its performance by providing the output as quickly as possible.
* Using this, the operating system checks the suffix memory space as soon as the execution of a program is completed. Here arrangements are made to free the memory space occupied by that program and store a new program at that place.
* It improves the performance of the computer’s memory so that it can complete its work at a higher speed.
* Memory management protects the data of a program and prevents unauthorized communication of one program’s data with another program’s data and keeps them isolated from each other.

[Static Vs. Dynamic Loading in Operating System (OS)](https://prepinsta.com/operating-systems/static-vs-dynamic-loading/)

[Static vs. Dynamic Linking in Operating System (OS)](https://prepinsta.com/operating-systems/static-vs-dynamic-linking/)

[Fragmentation in Operating System (OS) | Prepinsta](https://prepinsta.com/operating-systems/fragmentation/)

[Difference Between Segmentation and Paging in Operating System (OS)](https://prepinsta.com/operating-systems/difference-between-segmentation-and-paging/)

**Relocation register[6]:** A hardware element that holds a constant to be added to the address of each memory location in a computer program running in a multiprogramming system, as determined by the location of the area in memory assigned to the program.

**Memory Management[1] - (Chapter - 3)**

* 3.1: No Memory Abstraction
* 3.2: A Memory with Abstraction: Address Spaces
  + 3.2.1: The Notion of an Address Space
  + 3.2.2: Swapping
  + 3.2.3: Managing Free Memory
* 3.3: Virtual Memory
  + 3.3.1: Paging
  + 3.3.2: Page Tables
  + 3.3.3: Speeding Up Paging
  + 3.3.4: Page Table for Large Memories
* 3.4: Page Replacement Algorithms
  + 3.4.1: The Optimal Page Replacement Algorithm
  + 3.4.2: The Not Recently Used Page Replacement Algorithm
  + 3.4.3
  + 3.4.4
  + 3.4.5
  + 3.4.6

**Segmentation[8] - (8.4)**

* 8.4.1: Basic Method
* 8.4.2: Segmentation Hardware

**Main Memory[2] - (Chapter - 9)**

* 9.1: Background
  + 9.1.1: Basic Hardware
  + 9.1.2: Address binding
  + 9.1.3: Logical Versus Physical Address Space
  + 9.1.4: Dynamic Loading
  + 9.1.5: Dynamic Linking and Shared Libraries
* 9.2: Contiguous Memory Allocation
  + 9.2.1: Memory Protection
  + 9.2.2: Memory Allocation
  + 9.2.3: Fragmentation
* 9.3: Paging
  + 9.3.1: Basic Method
  + 9.3.2: Hardware Support
  + 9.3.3
  + 9.3.4
* 9.4: Structure of the Page Table
  + 9.4.1
* 9.5: Swapping
  + 9.5.1: Standard Swapping
  + 9.5.2: Swapping with paging

**Virtual Memory[2] - (Chapter - 10)**

* 10.1: Background
* 10.2
  + 10.2.1: Basic Concepts
  + 10.2.3: Performance of Demand Paging
* 10.3
* 10.4: Page Replacement
  + 10.4.1: Basic Page Replacement
  + 10.4.2: FIFO Page Replacement
  + 10.4.3: Optimal Page Replacement
  + 10.4.4: LRU Page Replacement

**File Systems**

**FS Services**

**Disk Space Management**

**Directory and Data Structures**

**Deadlocks and Case Study**

**Modeling**

**Detection and Recovery**

**Prevention and Avoidance**

**Case Study of Some Operating Systems**

**Others**

**Introduction to the Different Smart Device Operating System and their Usage**