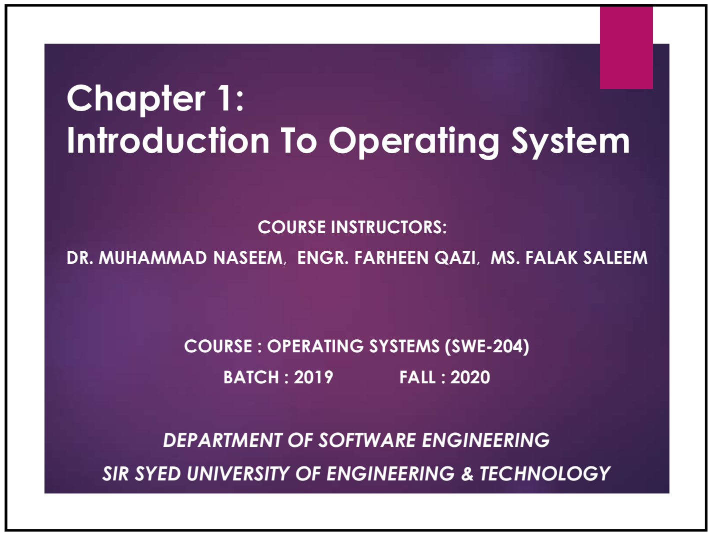


Chapter 1: Introduction To Operating System

COURSE INSTRUCTORS:

DR. MUHAMMAD NASEEM, ENGR. FARHEEN QAZI, MS. FALAK SALEEM

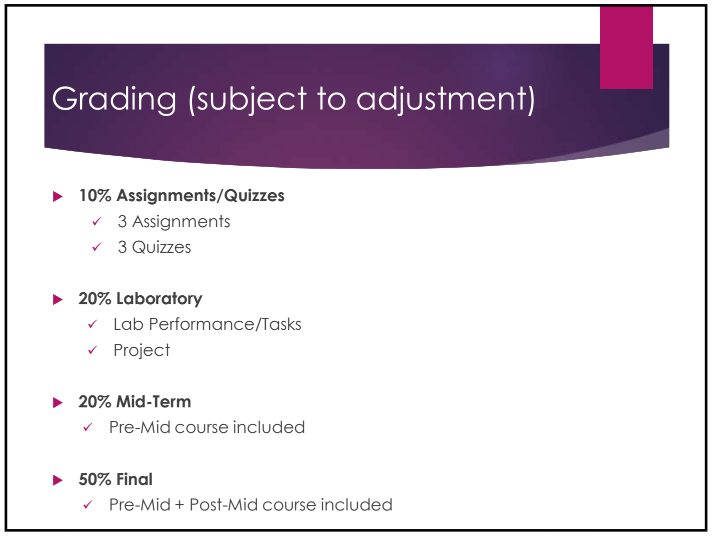
COURSE : OPERATING SYSTEMS (SWE-204)

BATCH : 2019 FALL : 2020

DEPARTMENT OF SOFTWARE ENGINEERING

SIR SYED UNIVERSITY OF ENGINEERING & TECHNOLOGY

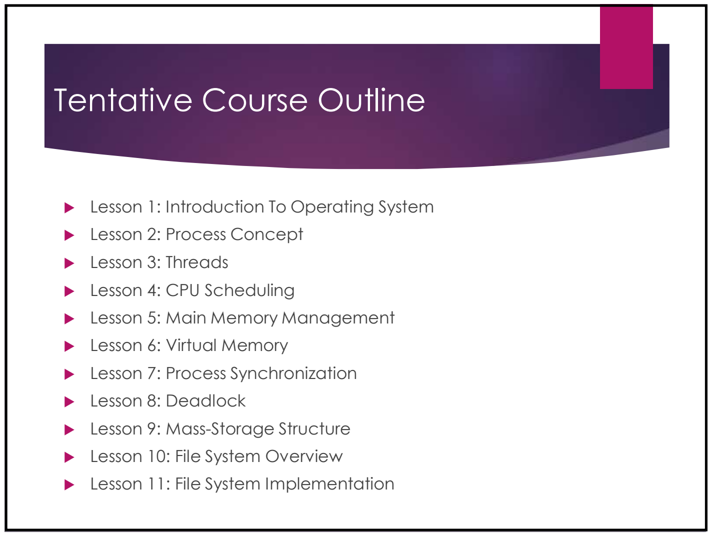
1



Grading (subject to adjustment)

- **10% Assignments/Quizzes**
  - ✓ 3 Assignments
  - ✓ 3 Quizzes
- **20% Laboratory**
  - ✓ Lab Performance/Tasks
  - ✓ Project
- **20% Mid-Term**
  - ✓ Pre-Mid course included
- **50% Final**
  - ✓ Pre-Mid + Post-Mid course included

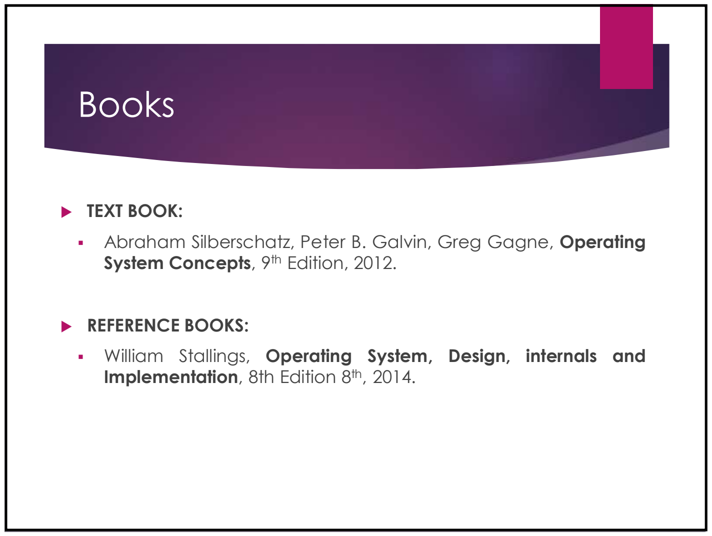
2



Tentative Course Outline

- Lesson 1: Introduction To Operating System
- Lesson 2: Process Concept
- Lesson 3: Threads
- Lesson 4: CPU Scheduling
- Lesson 5: Main Memory Management
- Lesson 6: Virtual Memory
- Lesson 7: Process Synchronization
- Lesson 8: Deadlock
- Lesson 9: Mass-Storage Structure
- Lesson 10: File System Overview
- Lesson 11: File System Implementation

3



Books

- **TEXT BOOK:**
  - Abraham Silberschatz, Peter B. Galvin, Greg Gagne, **Operating System Concepts**, 9th Edition, 2012.
- **REFERENCE BOOKS:**
  - William Stallings, **Operating System, Design, internals and Implementation**, 8th Edition 8th, 2014.

4

Today's Agenda

- What Operating Systems Do
- Computer System Structure
- Operating-System Operations
- Operating-System Structure
- Operating System Components
- System Calls
- Transition from User to Kernel Mode
- Process & Job Control
- Communication Models
- Proprietary VS Open-Source Operating Systems
- Virtual Machine (VM)
- Emerging Trends in Computing

5

Objectives

- To describe the basic organization of computer systems
- To provide a grand tour of the major components of operating systems
- To give an overview of the many types of computing environments
- To explore several open-source operating systems

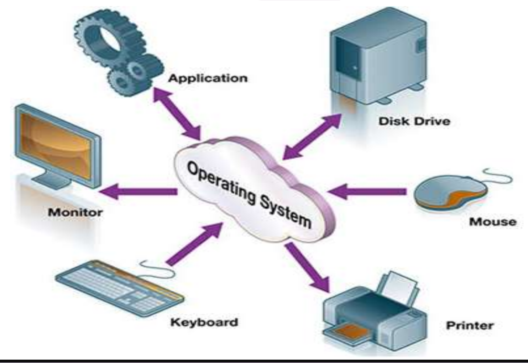
6

What is an Operating System?

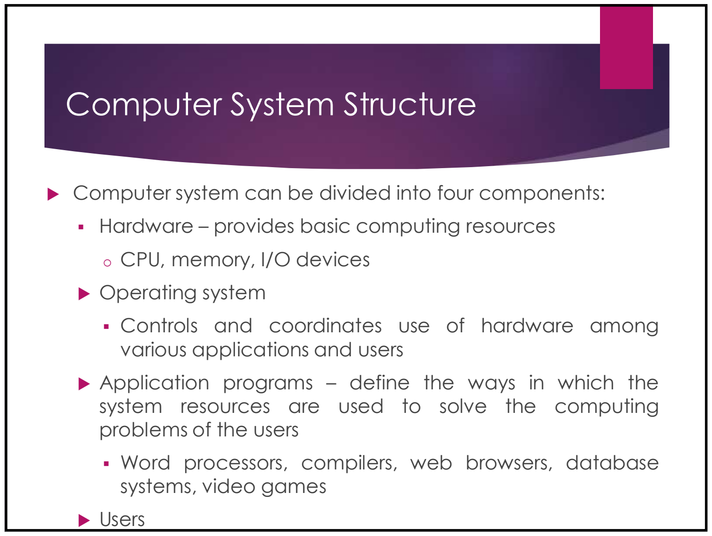
- A program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner

7

(Cont.)



8

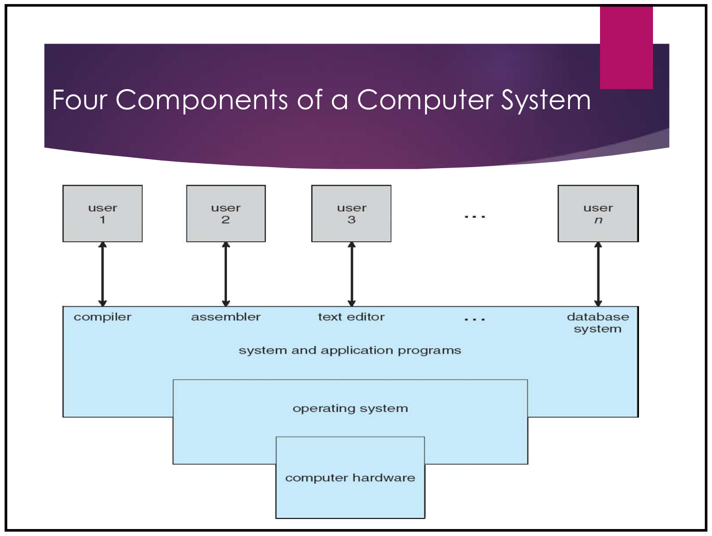


Computer System Structure

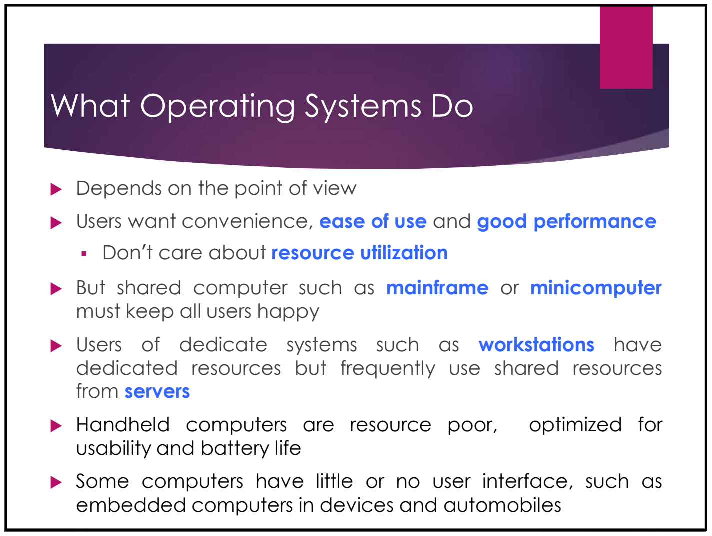
- Computer system can be divided into four components:
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs – define the ways in which the system resources are used to solve the computing problems of the users
    - Word processors, compilers, web browsers, database systems, video games
  - Users

9

Four Components of a Computer System



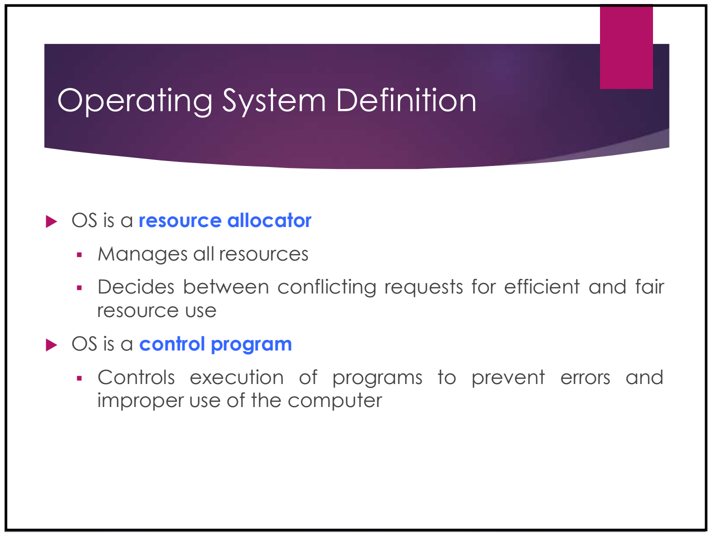
10



What Operating Systems Do

- Depends on the point of view
- Users want convenience, **ease of use** and **good performance**
  - Don't care about **resource utilization**
- But shared computer such as **mainframe** or **minicomputer** must keep all users happy
- Users of dedicate systems such as **workstations** have dedicated resources but frequently use shared resources from **servers**
- Handheld computers are resource poor, optimized for usability and battery life
- Some computers have little or no user interface, such as embedded computers in devices and automobiles

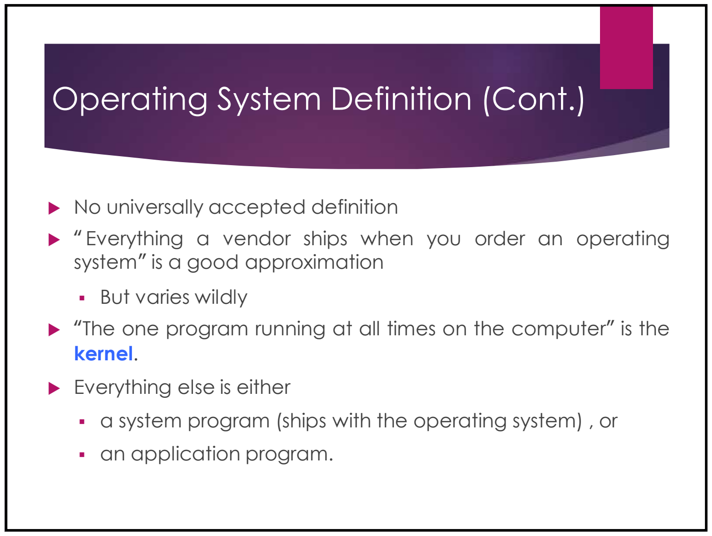
11



Operating System Definition

- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer

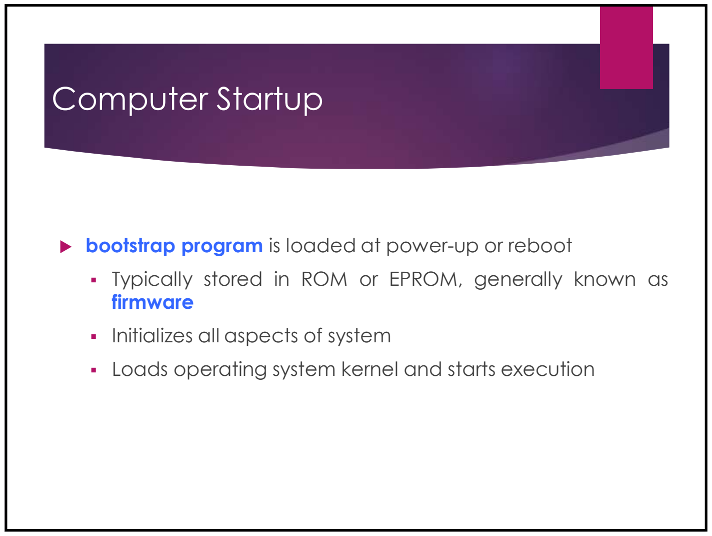
12



Operating System Definition (Cont.)

- No universally accepted definition
- "Everything a vendor ships when you order an operating system" is a good approximation
  - But varies wildly
- "The one program running at all times on the computer" is the **kernel**.
- Everything else is either
  - a system program (ships with the operating system) , or
  - an application program.

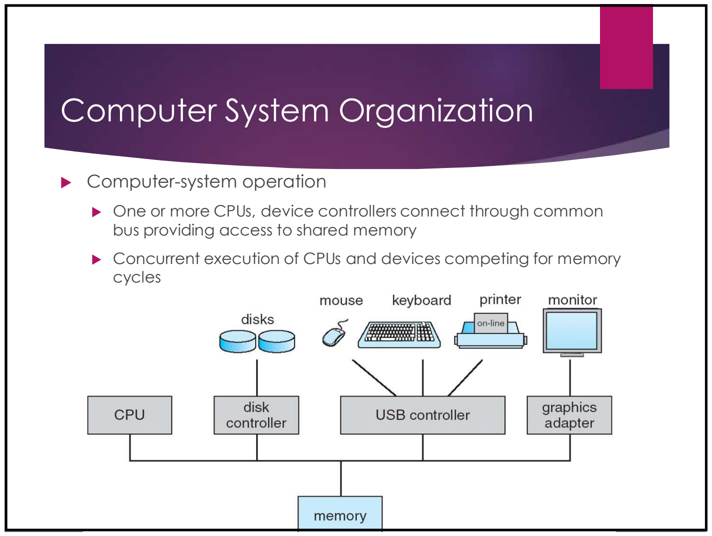
13



Computer Startup

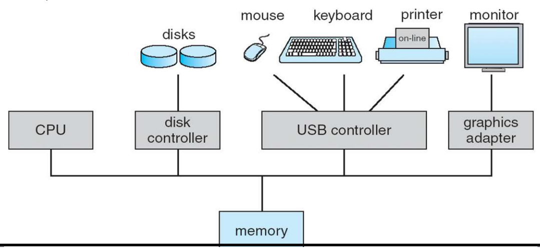
- **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware**
  - Initializes all aspects of system
  - Loads operating system kernel and starts execution

14

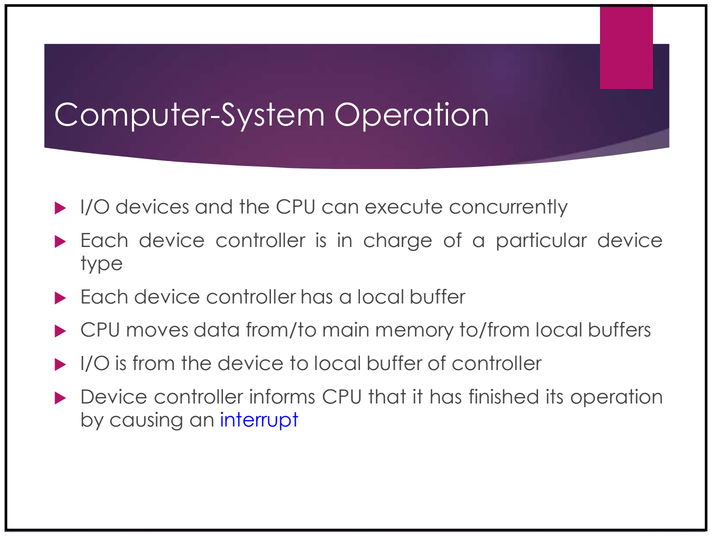


Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



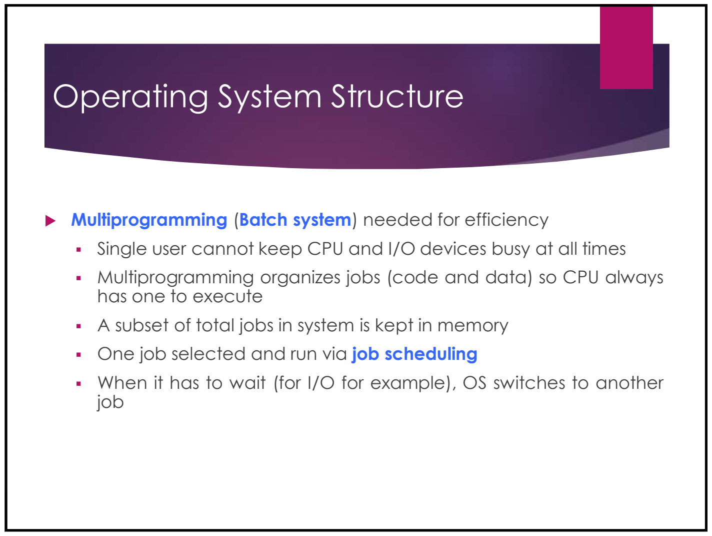
15



Computer-System Operation

- I/O devices and the CPU can execute concurrently
- Each device controller is in charge of a particular device type
- Each device controller has a local buffer
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller
- Device controller informs CPU that it has finished its operation by causing an interrupt

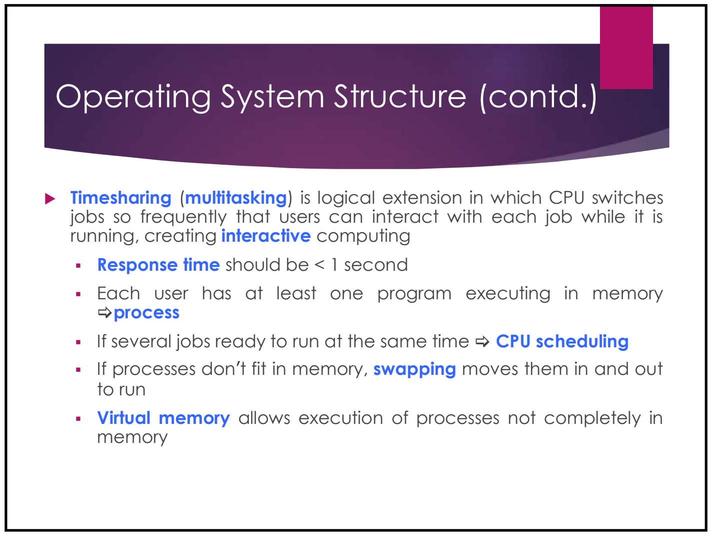
16



Operating System Structure

- **Multiprogramming** (**Batch system**) needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via **job scheduling**
  - When it has to wait (for I/O for example), OS switches to another job

17

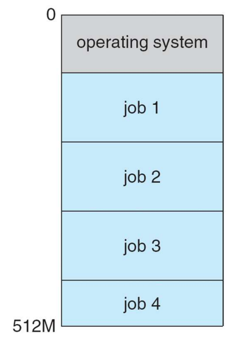


Operating System Structure (contd.)

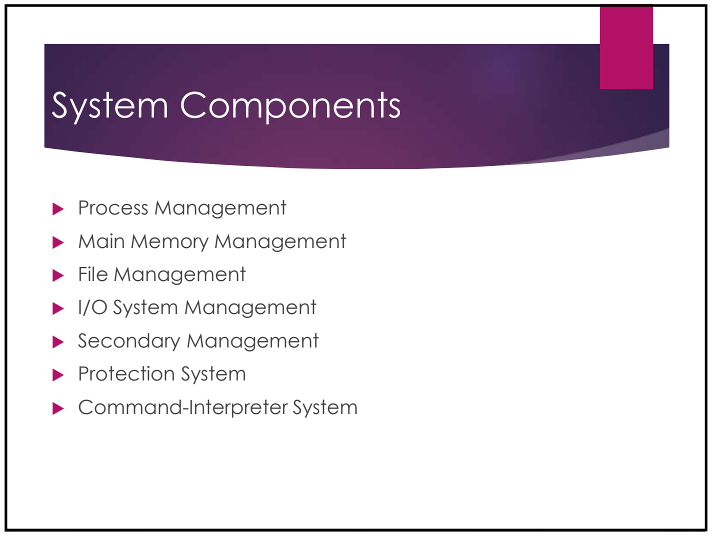
- **Timesharing** (**multitasking**) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨**process**
  - If several jobs ready to run at the same time ⇨ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

18

Memory Layout for Multiprogrammed System



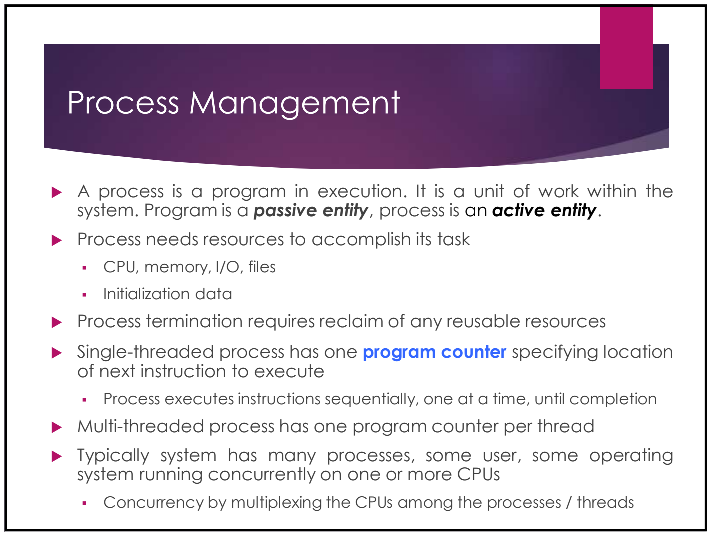
19



System Components

- Process Management
- Main Memory Management
- File Management
- I/O System Management
- Secondary Management
- Protection System
- Command-Interpreter System

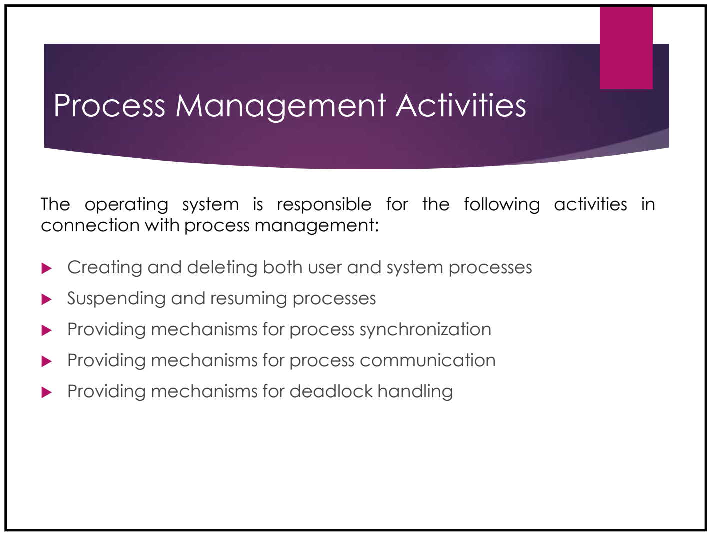
20



Process Management

- A process is a program in execution. It is a unit of work within the system. Program is a ***passive entity***, process is an ***active entity***.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources
- Single-threaded process has one **program counter** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the processes / threads

21

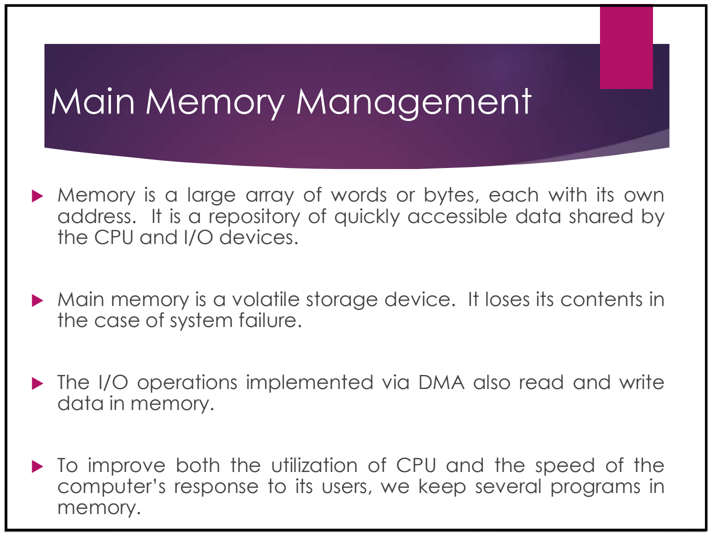


Process Management Activities

The operating system is responsible for the following activities in connection with process management:

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

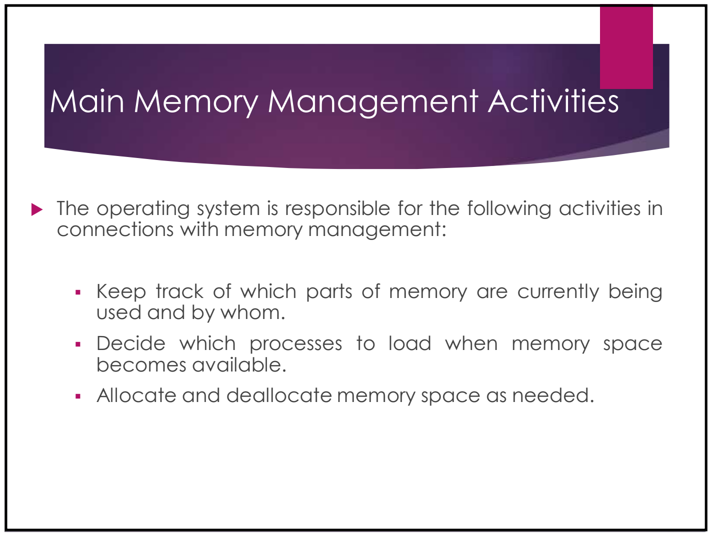
22



Main Memory Management

- Memory is a large array of words or bytes, each with its own address. It is a repository of quickly accessible data shared by the CPU and I/O devices.
- Main memory is a volatile storage device. It loses its contents in the case of system failure.
- The I/O operations implemented via DMA also read and write data in memory.
- To improve both the utilization of CPU and the speed of the computer's response to its users, we keep several programs in memory.

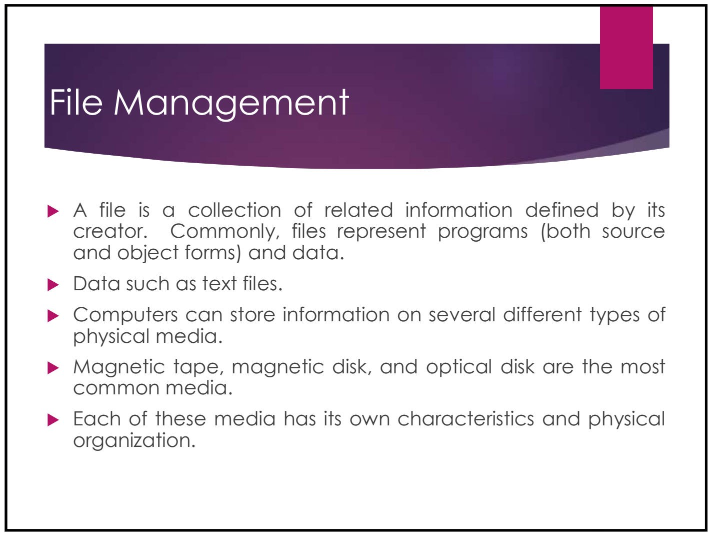
23



Main Memory Management Activities

- The operating system is responsible for the following activities in connections with memory management:
  - Keep track of which parts of memory are currently being used and by whom.
  - Decide which processes to load when memory space becomes available.
  - Allocate and deallocate memory space as needed.

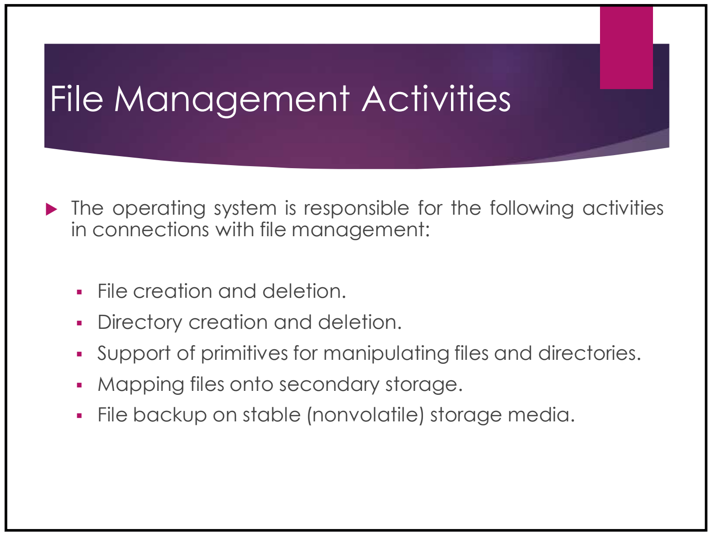
24



File Management

- A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data.
- Data such as text files.
- Computers can store information on several different types of physical media.
- Magnetic tape, magnetic disk, and optical disk are the most common media.
- Each of these media has its own characteristics and physical organization.

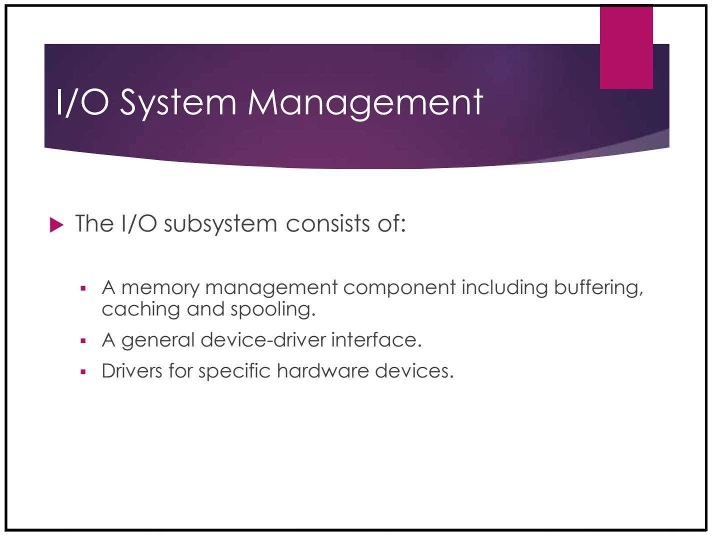
25



File Management Activities

- The operating system is responsible for the following activities in connections with file management:
  - File creation and deletion.
  - Directory creation and deletion.
  - Support of primitives for manipulating files and directories.
  - Mapping files onto secondary storage.
  - File backup on stable (nonvolatile) storage media.

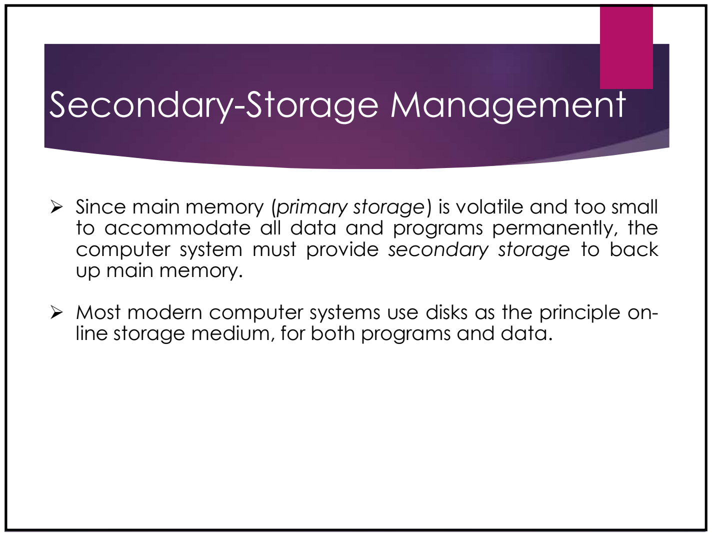
26



I/O System Management

- The I/O subsystem consists of:
  - A memory management component including buffering, caching and spooling.
  - A general device-driver interface.
  - Drivers for specific hardware devices.

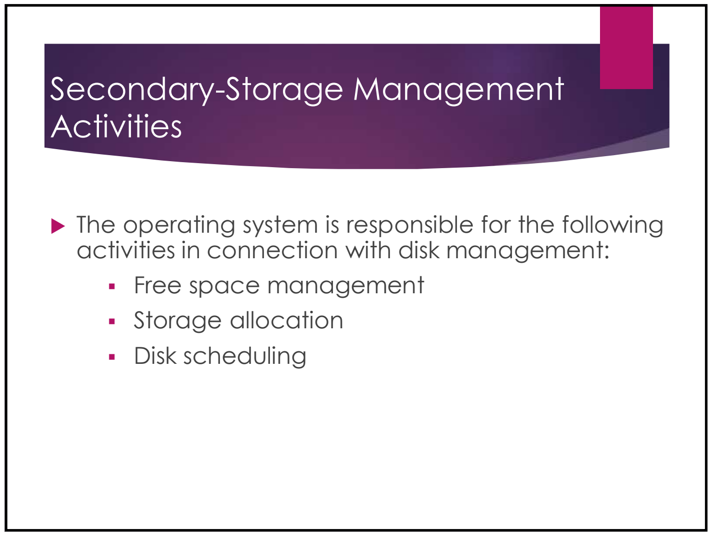
27



Secondary-Storage Management

- Since main memory (*primary storage*) is volatile and too small to accommodate all data and programs permanently, the computer system must provide *secondary storage* to back up main memory.
- Most modern computer systems use disks as the principle on-line storage medium, for both programs and data.

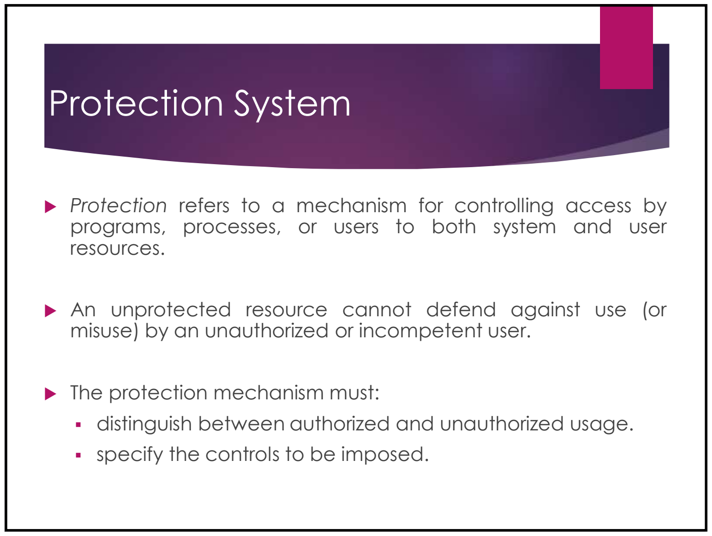
28



Secondary-Storage Management Activities

- The operating system is responsible for the following activities in connection with disk management:
  - Free space management
  - Storage allocation
  - Disk scheduling

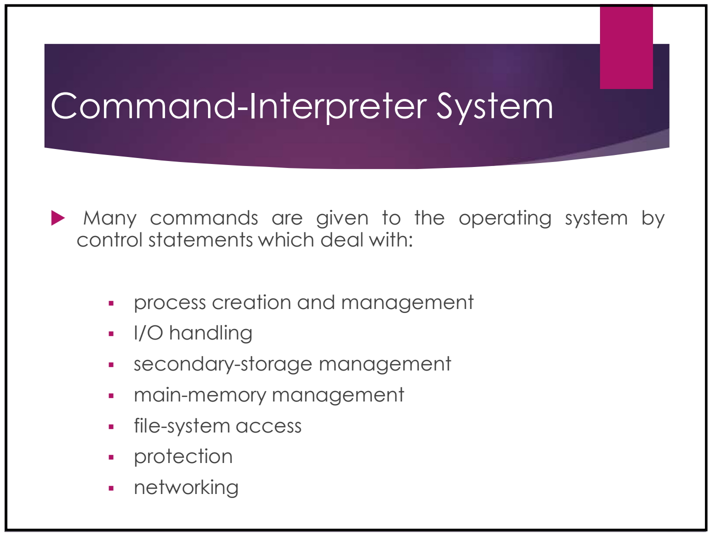
29



Protection System

- *Protection* refers to a mechanism for controlling access by programs, processes, or users to both system and user resources.
- An unprotected resource cannot defend against use (or misuse) by an unauthorized or incompetent user.
- The protection mechanism must:
  - distinguish between authorized and unauthorized usage.
  - specify the controls to be imposed.

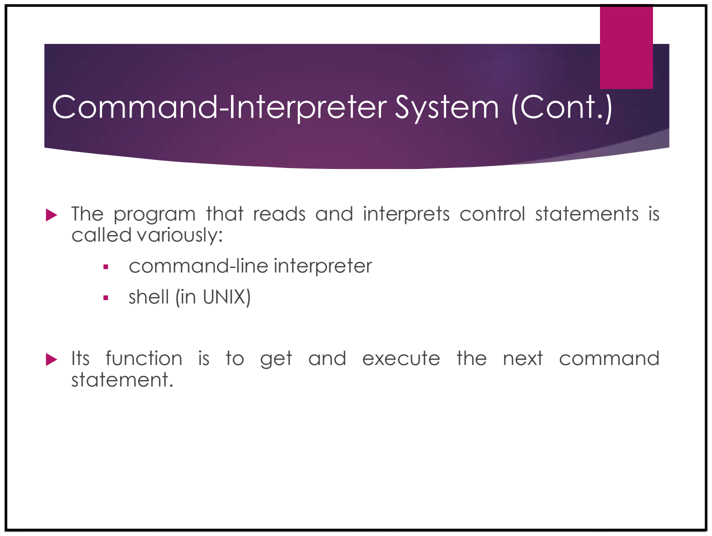
30



Command-Interpreter System

- Many commands are given to the operating system by control statements which deal with:
  - process creation and management
  - I/O handling
  - secondary-storage management
  - main-memory management
  - file-system access
  - protection
  - networking

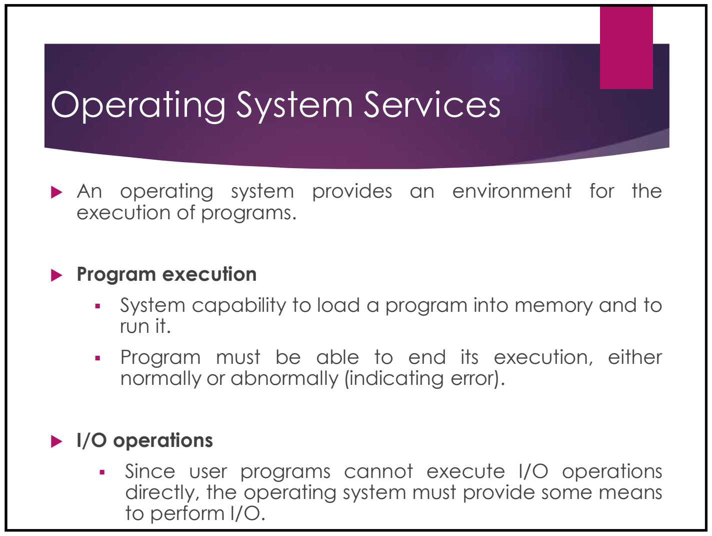
31



Command-Interpreter System (Cont.)

- The program that reads and interprets control statements is called variously:
  - command-line interpreter
  - shell (in UNIX)
- Its function is to get and execute the next command statement.

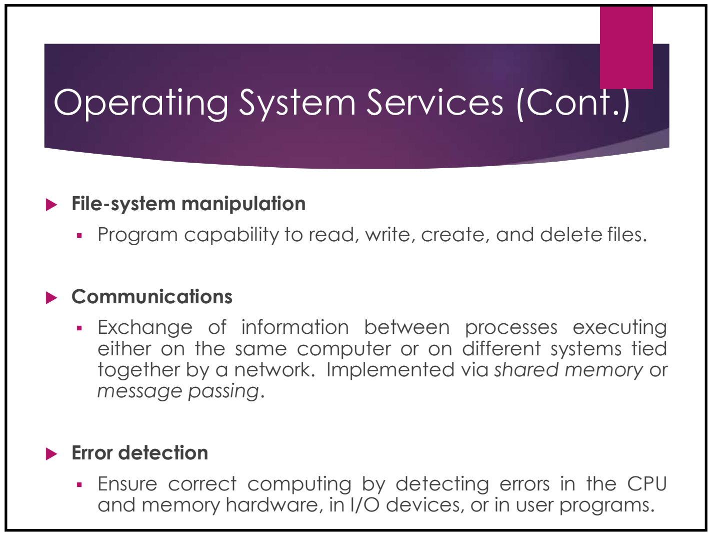
32



Operating System Services

- An operating system provides an environment for the execution of programs.
- **Program execution**
  - System capability to load a program into memory and to run it.
  - Program must be able to end its execution, either normally or abnormally (indicating error).
- **I/O operations**
  - Since user programs cannot execute I/O operations directly, the operating system must provide some means to perform I/O.

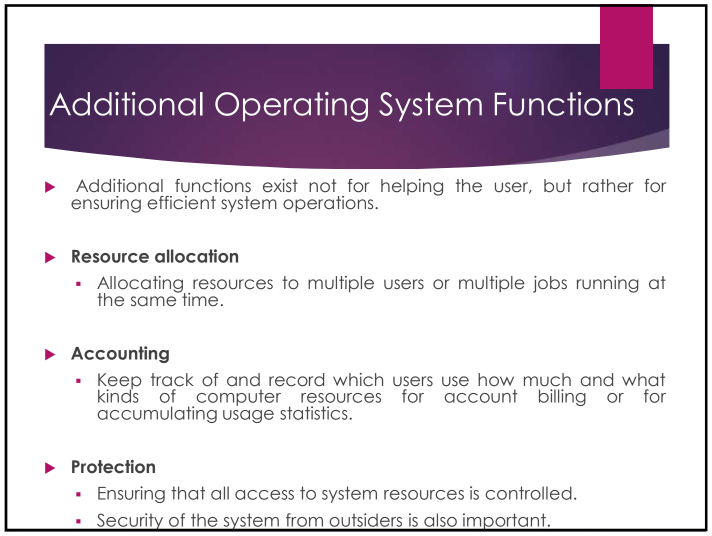
33



Operating System Services (Cont.)

- **File-system manipulation**
  - Program capability to read, write, create, and delete files.
- **Communications**
  - Exchange of information between processes executing either on the same computer or on different systems tied together by a network. Implemented via *shared memory* or *message passing*.
- **Error detection**
  - Ensure correct computing by detecting errors in the CPU and memory hardware, in I/O devices, or in user programs.

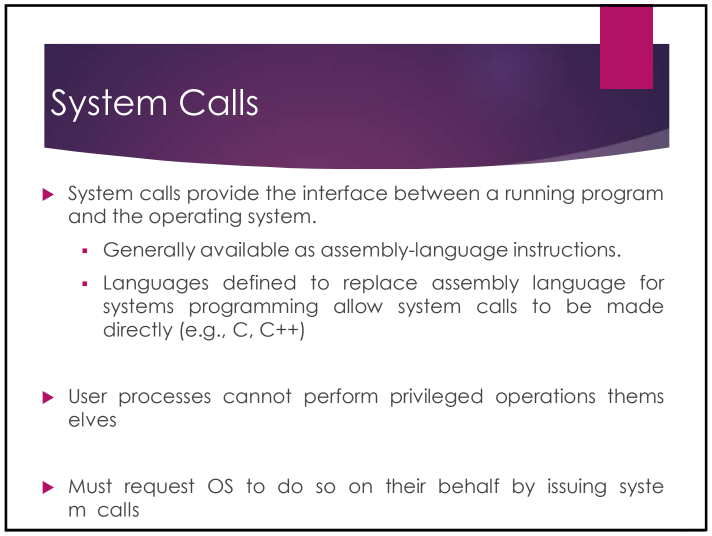
34



Additional Operating System Functions

- Additional functions exist not for helping the user, but rather for ensuring efficient system operations.
- **Resource allocation**
  - Allocating resources to multiple users or multiple jobs running at the same time.
- **Accounting**
  - Keep track of and record which users use how much and what kinds of computer resources for account billing or for accumulating usage statistics.
- **Protection**
  - Ensuring that all access to system resources is controlled.
  - Security of the system from outsiders is also important.

35

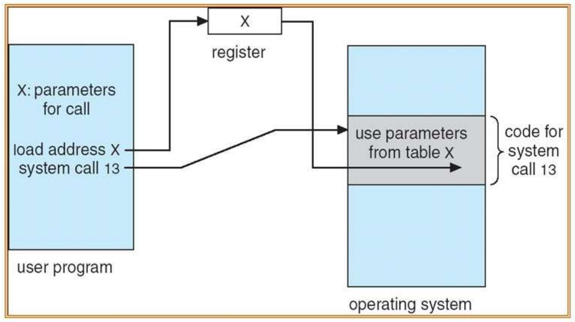


System Calls

- System calls provide the interface between a running program and the operating system.
  - Generally available as assembly-language instructions.
  - Languages defined to replace assembly language for systems programming allow system calls to be made directly (e.g., C, C++)
- User processes cannot perform privileged operations thems elves
- Must request OS to do so on their behalf by issuing syste m calls

36

Passing of Parameters As A Table



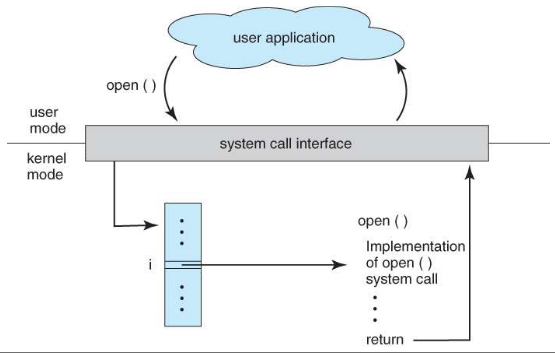
37

System Call Implementation

- Typically, a number associated with each system call
  - **System-call interface** maintains a table indexed according to these numbers
- The system call interface invokes the intended system call in OS kernel and returns status of the system call and any return values

38

API – System Call – OS Relationship



39

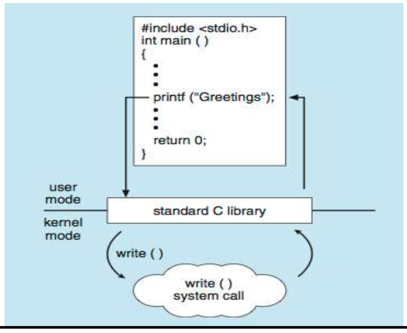
System Call Parameter Passing

- Often, more information is required than simply identity of desired system call
  - Exact type and amount of information vary according to OS and call
- Three general methods used to pass parameters to the OS
  - Simplest: pass the parameters in registers
    - In some cases, may be more parameters than registers
  - Parameters stored in a block, or table, in memory, and address of block passed as a parameter in a register
    - This approach taken by Linux and Solaris
  - Parameters placed, or **pushed**, onto the **stack** by the program and **popped** off the stack by the operating system
  - Block and stack methods do not limit the number or length of parameters being passed

40

Standard C Library Example

- C program invoking printf() library call, which calls write() system call



41

Types of System Calls

- **Process control**
  - create process, terminate process
  - end, abort
  - load, execute
  - get process attributes, set process attributes
  - wait for time
  - wait event, signal event
  - allocate and free memory

42

Types of System Calls

- **File management**
  - create file, delete file
  - open, close file
  - read, write, reposition
  - get and set file attributes
- **Device management**
  - request device, release device
  - read, write, reposition
  - get device attributes, set device attributes
  - logically attach or detach devices

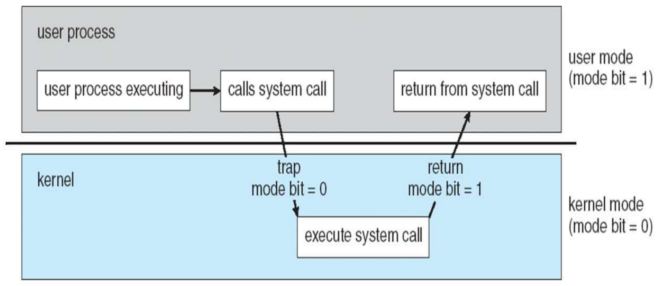
43

Types of System Calls (Cont.)

- **Protection**
  - Control access to resources
  - Get and set permissions
  - Allow and deny user access
- **Information maintenance**
  - get time or date, set time or date
  - get system data, set system data
  - get and set process, file, or device attributes

44

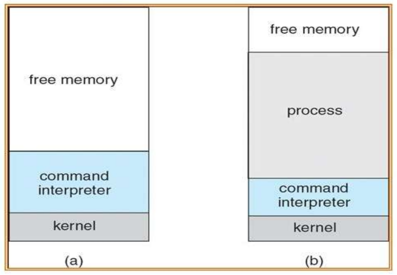
Transition from User to Kernel Mode



45

Process and job control MS-DOS Execution

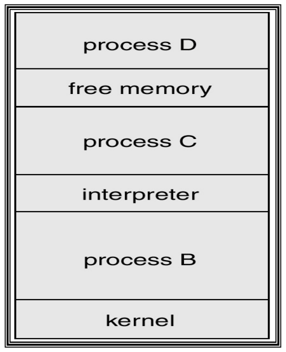
Simple, single-user process control



(a) At system startup (b) running a program

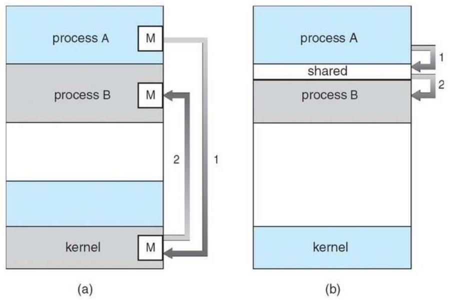
46

Process and job control UNIX Running Multiple Programs



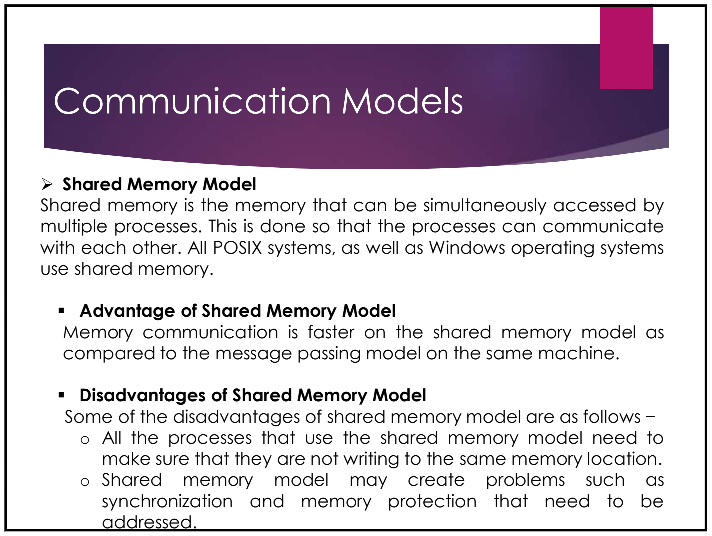
47

Process and job control Communication Models



(a) Message passing, (b) Shared memory

48



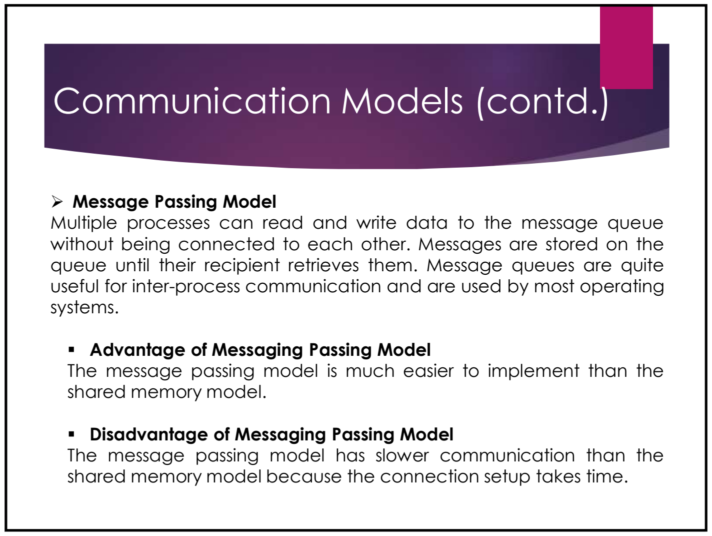
Communication Models

➢ **Shared Memory Model**
Shared memory is the memory that can be simultaneously accessed by multiple processes. This is done so that the processes can communicate with each other. All POSIX systems, as well as Windows operating systems use shared memory.

- **Advantage of Shared Memory Model**
  Memory communication is faster on the shared memory model as compared to the message passing model on the same machine.

- **Disadvantages of Shared Memory Model**
  Some of the disadvantages of shared memory model are as follows –
  - All the processes that use the shared memory model need to make sure that they are not writing to the same memory location.
  - Shared memory model may create problems such as synchronization and memory protection that need to be addressed.

49



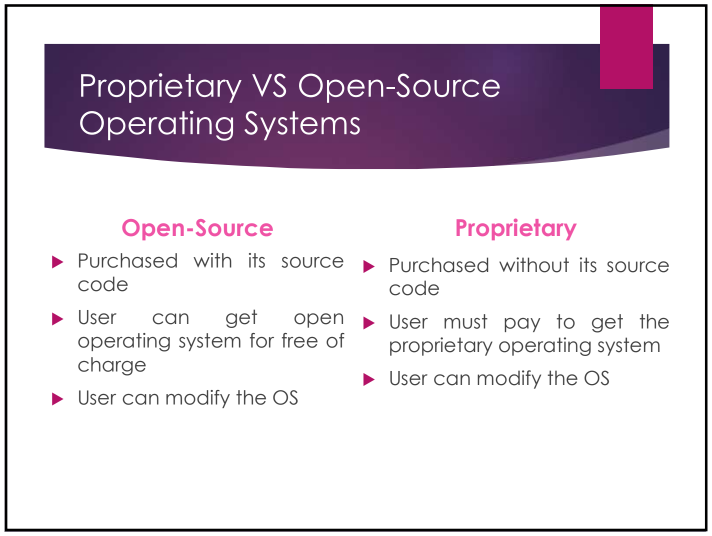
Communication Models (contd.)

➢ **Message Passing Model**
Multiple processes can read and write data to the message queue without being connected to each other. Messages are stored on the queue until their recipient retrieves them. Message queues are quite useful for inter-process communication and are used by most operating systems.

- **Advantage of Messaging Passing Model**
  The message passing model is much easier to implement than the shared memory model.

- **Disadvantage of Messaging Passing Model**
  The message passing model has slower communication than the shared memory model because the connection setup takes time.

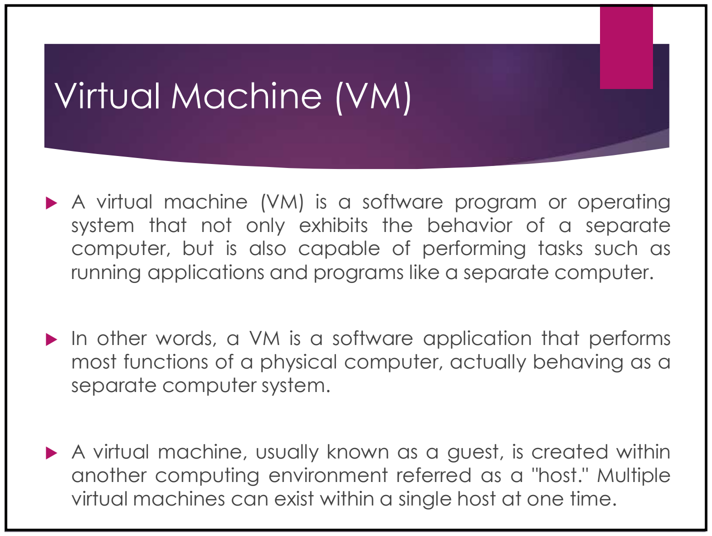
50



Proprietary VS Open-Source Operating Systems

Open-Source	Proprietary
▶ Purchased with its source code	▶ Purchased without its source code
▶ User can get open operating system for free of charge	▶ User must pay to get the proprietary operating system
▶ User can modify the OS	▶ User can modify the OS

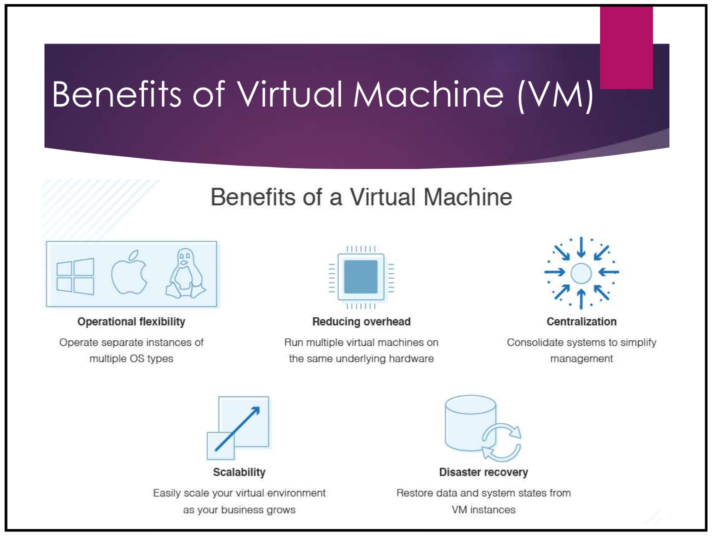
51



Virtual Machine (VM)

- A virtual machine (VM) is a software program or operating system that not only exhibits the behavior of a separate computer, but is also capable of performing tasks such as running applications and programs like a separate computer.
- In other words, a VM is a software application that performs most functions of a physical computer, actually behaving as a separate computer system.
- A virtual machine, usually known as a guest, is created within another computing environment referred as a "host." Multiple virtual machines can exist within a single host at one time.

52



Benefits of Virtual Machine (VM)

Benefits of a Virtual Machine

Operational flexibility
Operate separate instances of multiple OS types

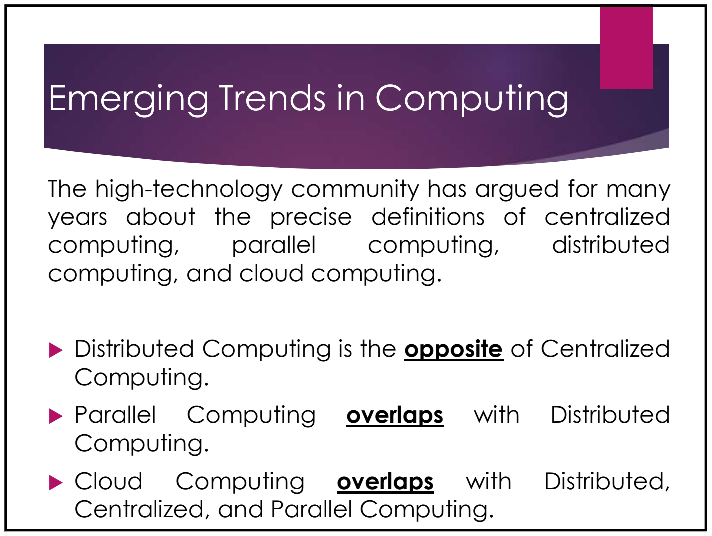
Reducing overhead
Run multiple virtual machines on the same underlying hardware

Centralization
Consolidate systems to simplify management

Scalability
Easily scale your virtual environment as your business grows

Disaster recovery
Restore data and system states from VM instances

53

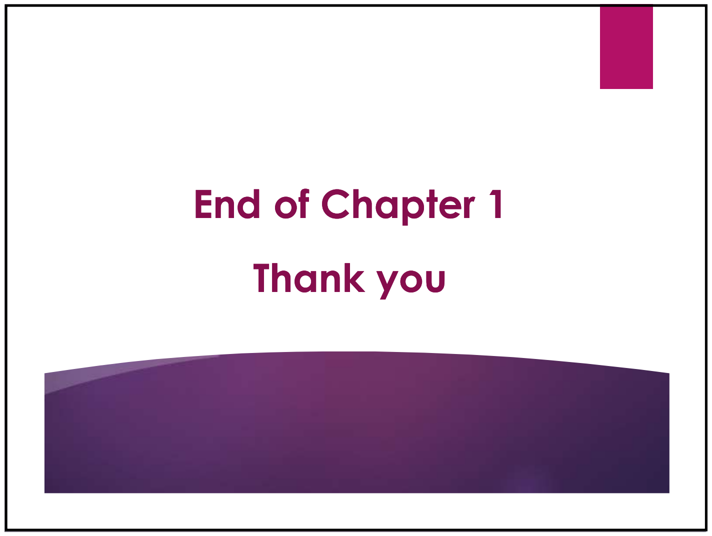


Emerging Trends in Computing

The high-technology community has argued for many years about the precise definitions of centralized computing, parallel computing, distributed computing, and cloud computing.

- Distributed Computing is the **opposite** of Centralized Computing.
- Parallel Computing **overlaps** with Distributed Computing.
- Cloud Computing **overlaps** with Distributed, Centralized, and Parallel Computing.

54



End of Chapter 1

Thank you

55