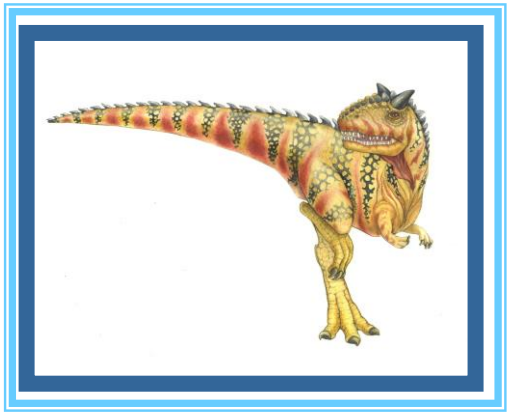
Introduction to Operating System
Day1: Sep 2021

Kiran Waghmare



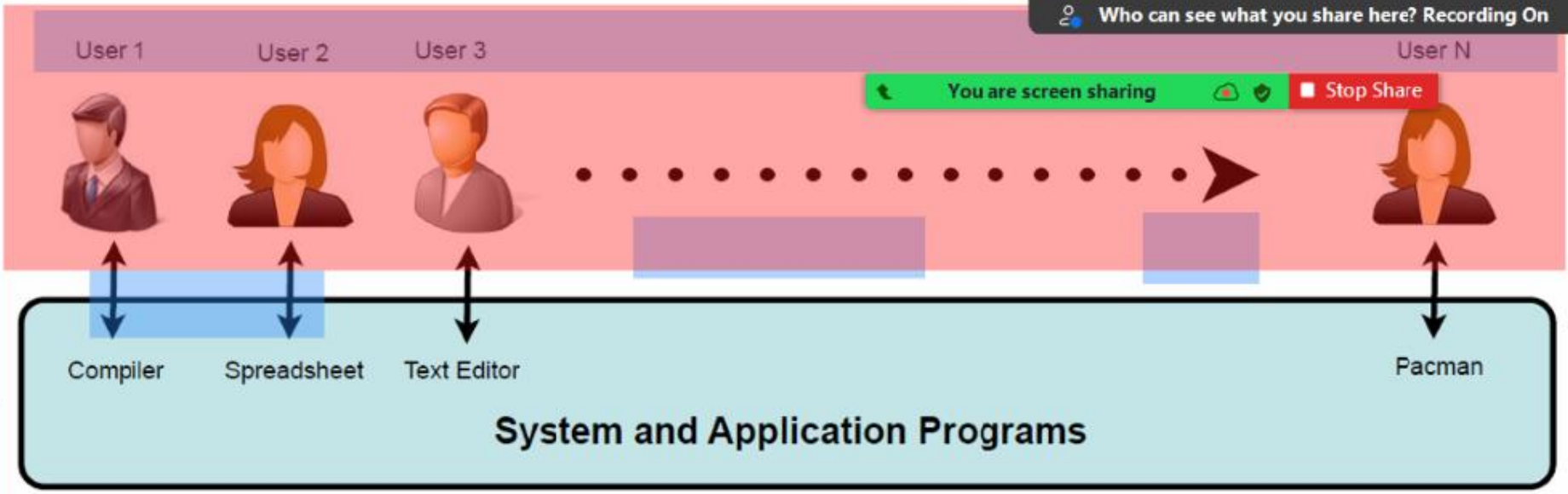
Learning and understanding

Top down

Bottom up

What is an Operating System?

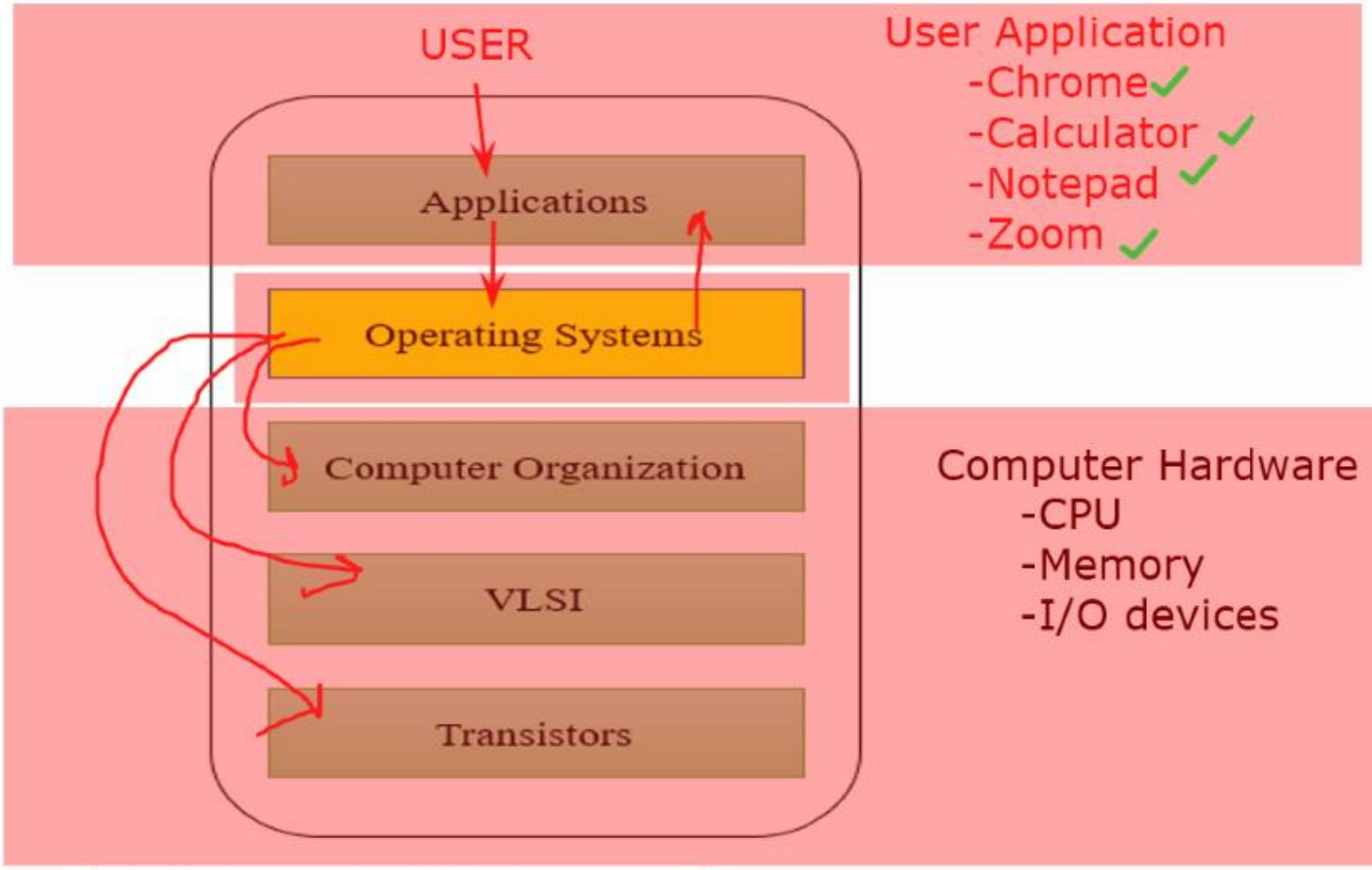
- A program that acts as an **intermediary between a user of a computer and the computer hardware**
- **Operating system goals:**
  - Execute user programs and **make solving user problems easier**
  - Make the **computer system convenient** to use
  - Use the computer hardware in an efficient manner



Operating System

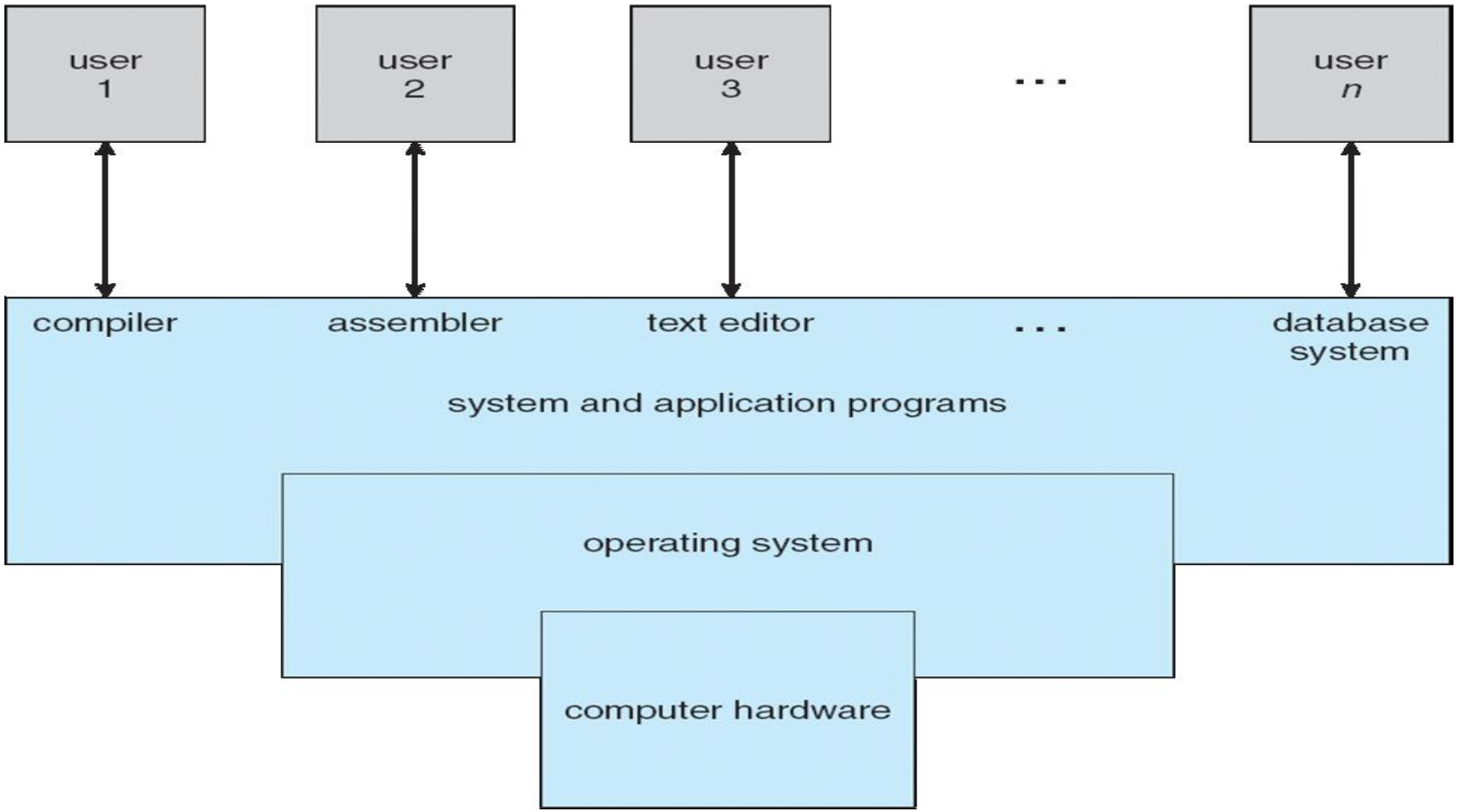
Controls the hardware and coordinates its use among the various application programs for the various users.

The Layers in Systems



1. Hardware Abstraction
2. Resource Mamagement

Four Components of a Computer System



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

Input devices

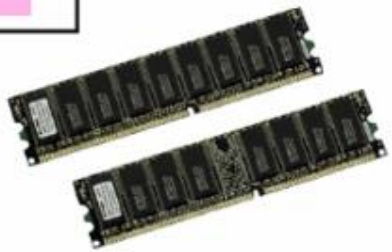
CPU

Control unit

Arithmetic logic unit

Output devices

Memory

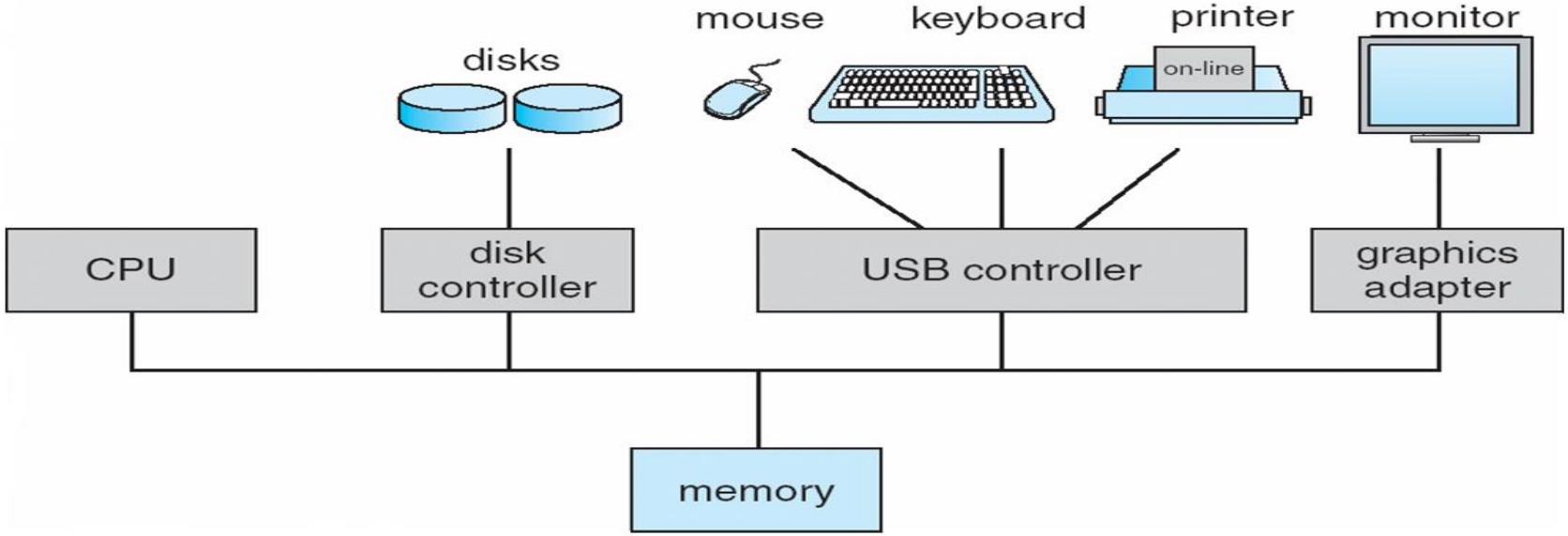


External storage

The von Neuman modell

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



Operating System Management Tasks

1. **Process management** which involves putting the tasks into order and pairing them into manageable size before they go to the CPU.
2. **Memory management** which coordinates data to and from RAM (random-access memory) and determines the necessity for virtual memory.
3. **Device management** provides an interface between connected devices.
4. **Storage management** which directs permanent data storage.
5. **An application** that allows standard communication between software and your computer.
6. **The user interface** allows you to communicate with your computer.

Functions of Operating System

1. It boots the computer
2. It performs basic computer tasks e.g. managing the various peripheral devices e.g. mouse, keyboard
3. It provides a user interface, e.g. command line, graphical user interface (GUI)
4. It handles system resources such as the computer's memory and sharing of the central processing unit(CPU) time by various applications or peripheral devices.
5. It provides file management which refers to the way that the operating system manipulates, stores, retrieves, and saves data.
6. Error Handling is done by the operating system. It takes preventive measures whenever required to avoid errors.

Types of Operating Systems

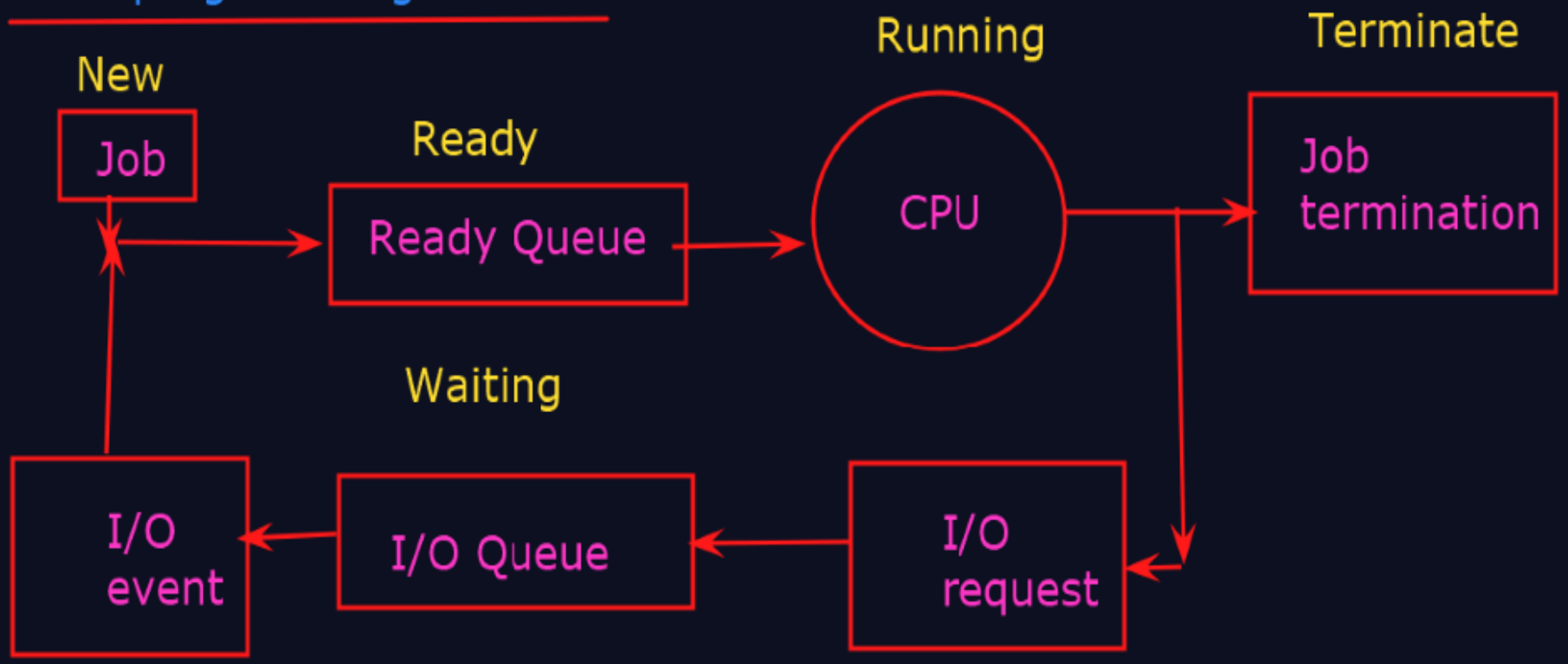
- Following are some of the most widely used types of Operating system.
  - Simple Batch System
  - Multiprogramming Batch System
  - Multiprocessor System
  - Desktop System
  - Distributed Operating System
  - Clustered System
  - Realtime Operating System
  - Handheld System

Day 11: Operating System
Date: 20-09-2022

Topics:

-Introduction to Operating System

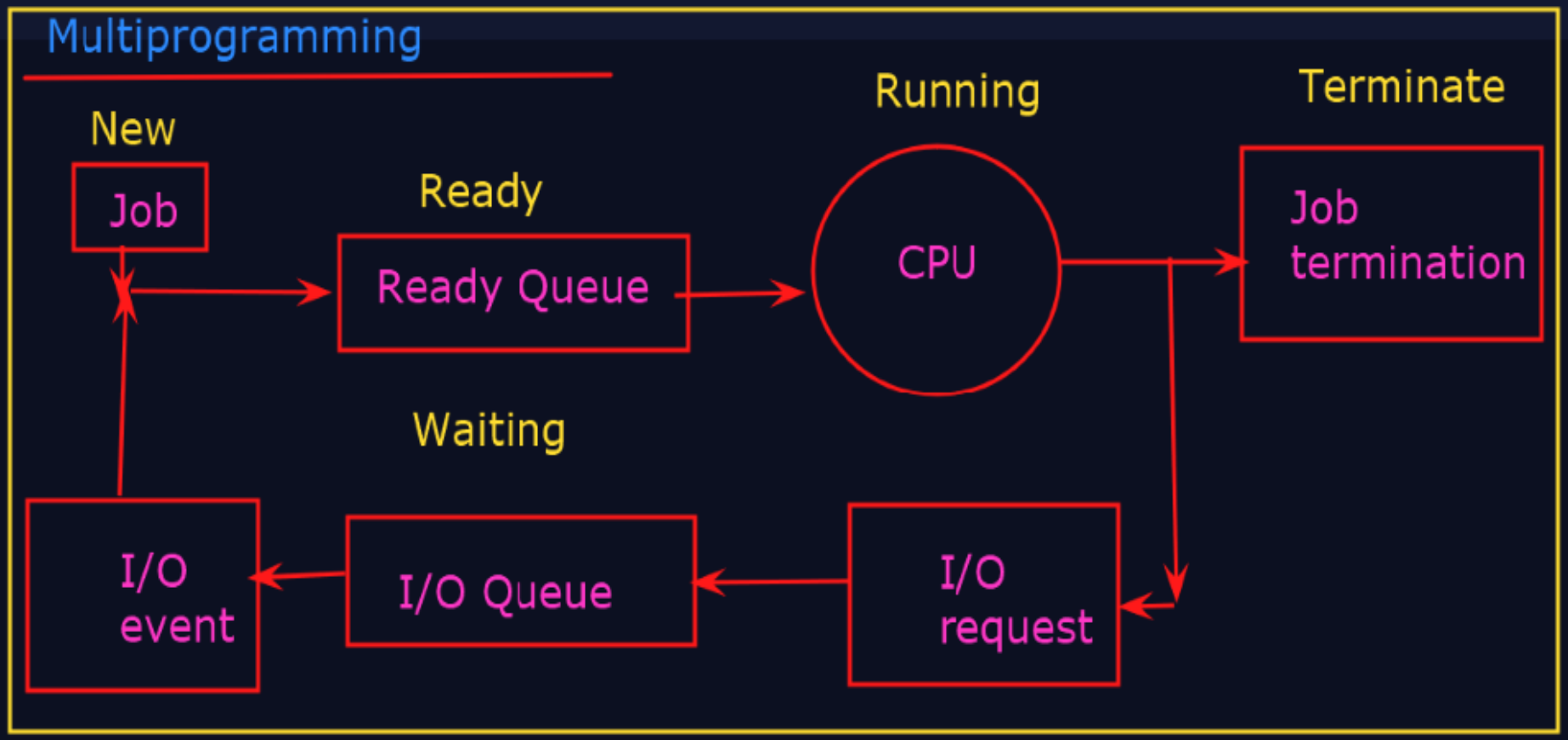
Multiprogramming



Day 11: Operating System
Date: 20-09-2022

Topics:

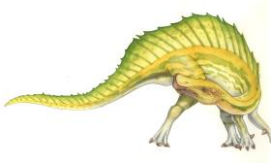
-Introduction to Operating System



Multiprogramming :interrupt CPU utilization
Multitasking :slice time sharing:5min
Multiprocessing

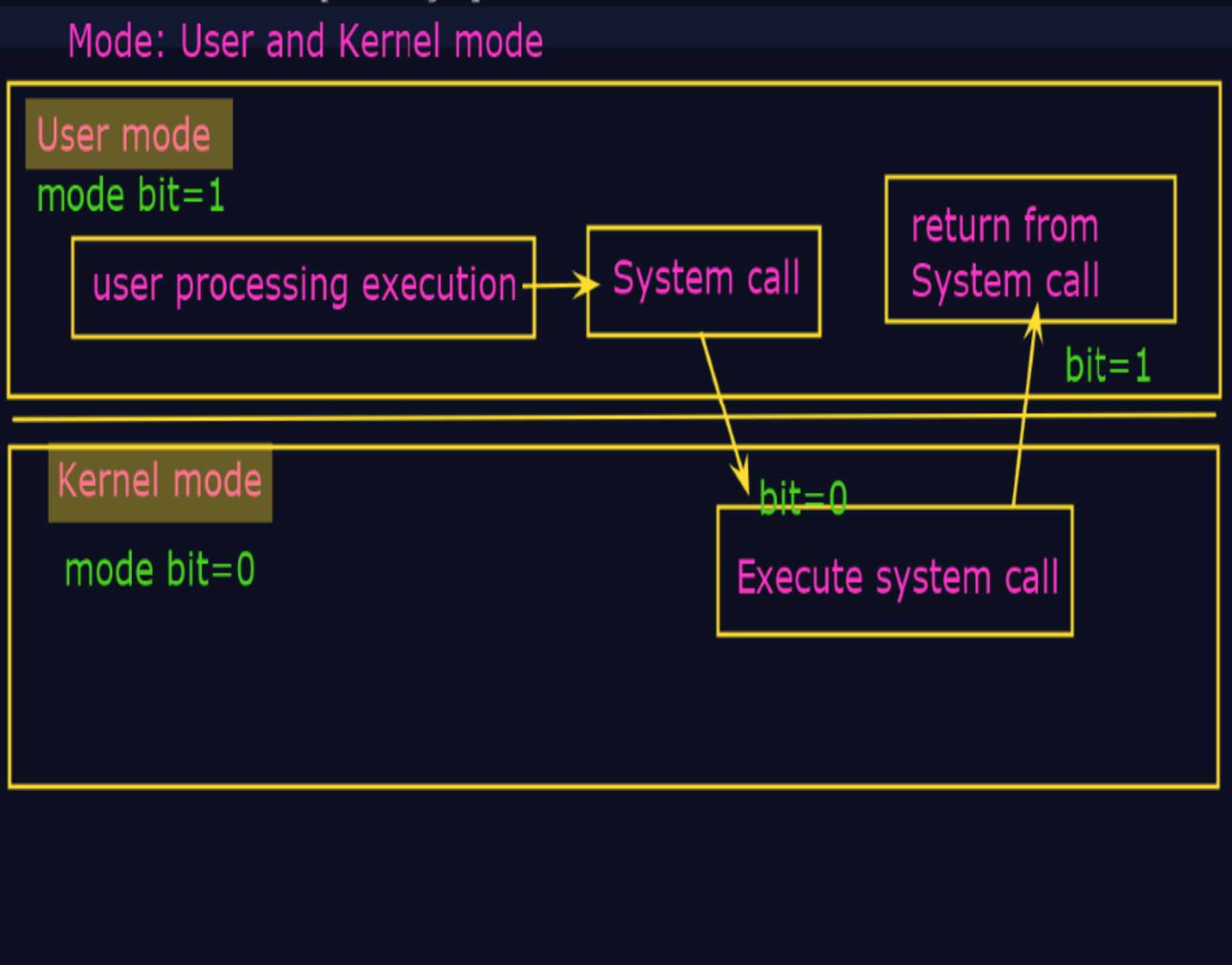
Computer-System Architecture

- Most systems use a single general-purpose processor (PDAs through mainframes)
  - Most systems have special-purpose processors as well
- Multiprocessors systems growing in use and importance
  - Also known as parallel systems, tightly-coupled systems
  - Advantages include
    1. Increased throughput
    2. Economy of scale
    3. Increased reliability – graceful degradation or fault tolerance
  - Two types
    1. Asymmetric Multiprocessing
    2. Symmetric Multiprocessing



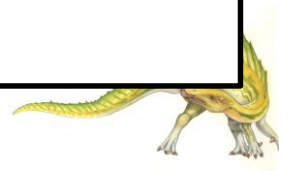
Operating-System Operations

- Interrupt driven by hardware
- Software error or request creates **exception** or **trap**
  - Division by zero, request for operating system service
- Other process problems include infinite loop, processes modifying each other or the operating system
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user


Mode: User and Kernel mode
User mode
mode bit=1
user processing execution
System call
return from
System call
bit=1
Kernel mode
mode bit=0
bit=0
Execute system call

Types of Distributed Operating Systems

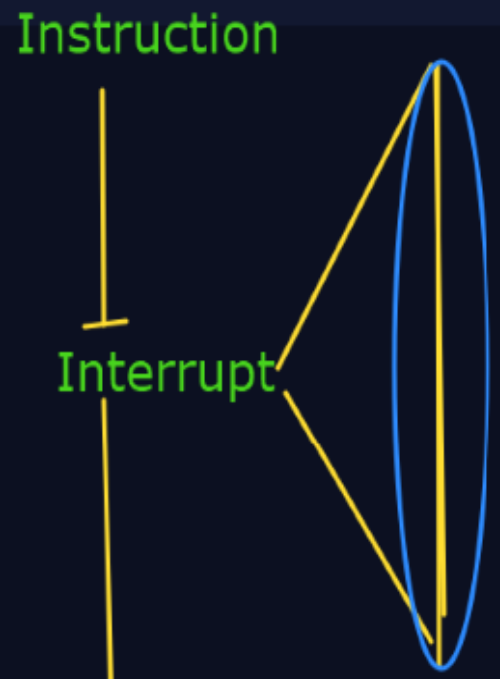
- Following are the two types of distributed operating systems used:
  - Client-Server Systems
  - Peer-to-Peer Systems
- **Client-Server Systems**
- Centralized systems today act as server systems to satisfy requests generated by client systems. The general structure of a client-server system is depicted in the figure below:
- Server Systems can be broadly categorized as: Compute Servers and File Servers.
- Compute Server systems, provide an interface to which clients can send requests to perform an action, in response to which they execute the action and send back results to the client.
- File Server systems, provide a file-system interface where clients can create, update, read, and delete files.

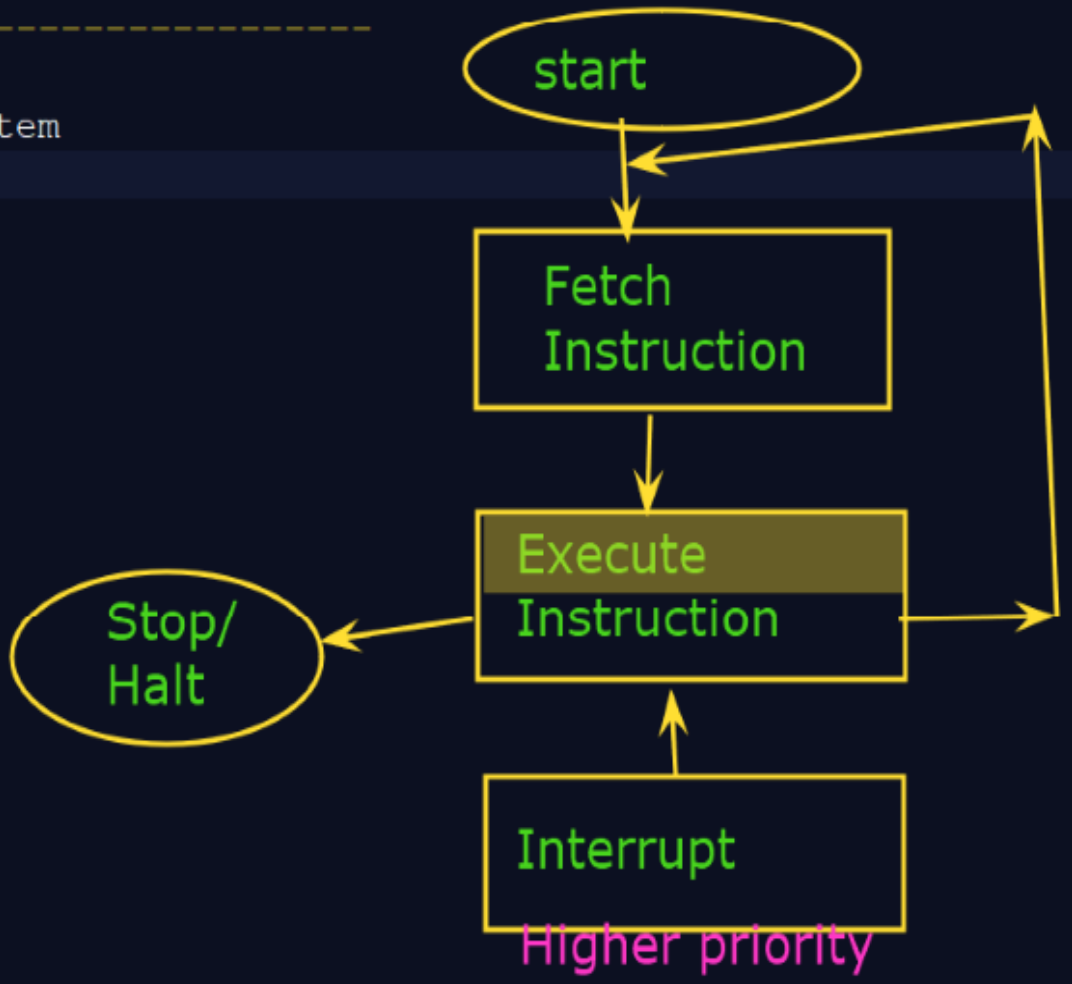


Day 11: Operating System
Date: 20-09-2022

Topics:

-Introduction to Operating System

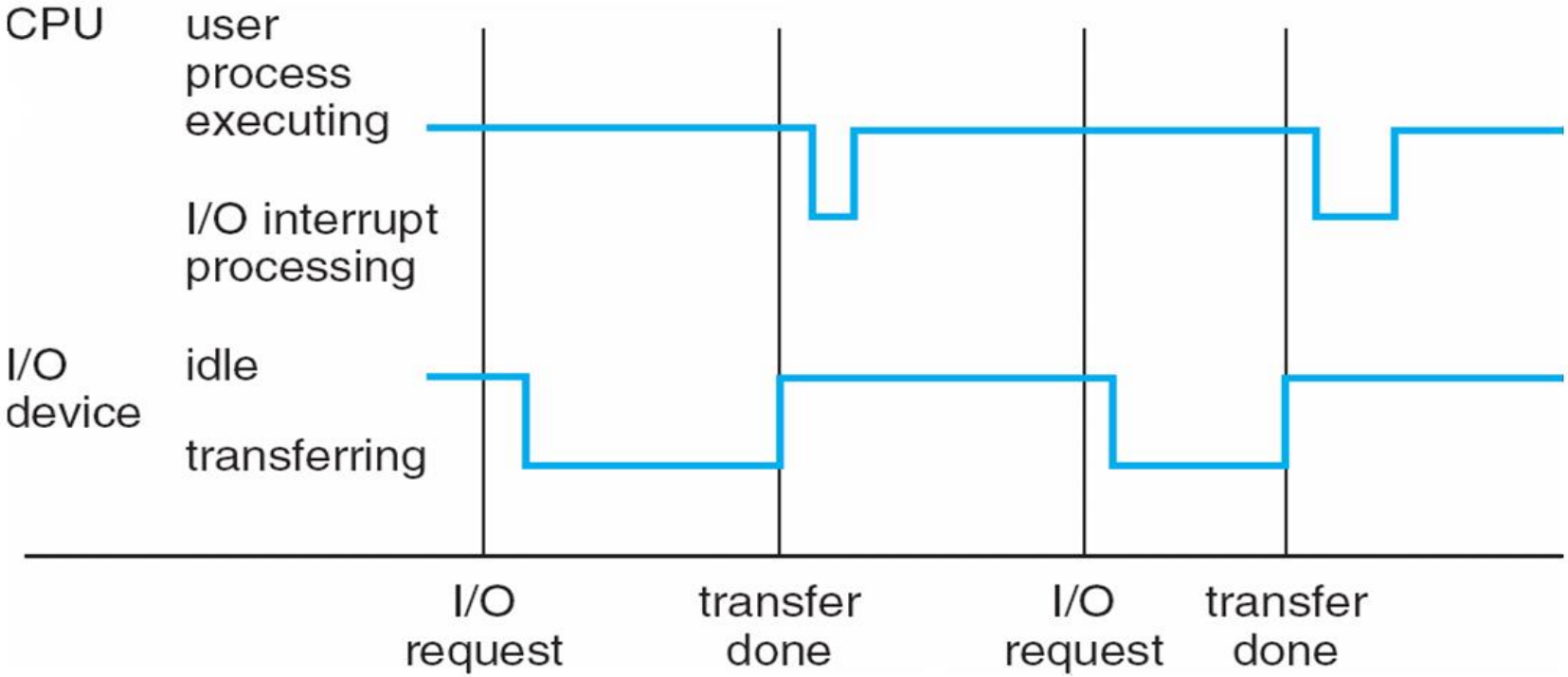




Interrupt:

-defines as event that alter the sequence of instructions executed by a processor.

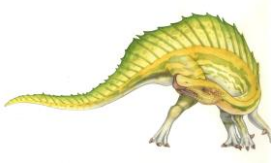
Interrupt Timeline



User Operating System Interface - CLI

Command Line Interface (CLI) or command interpreter allows direct command entry

- Sometimes implemented in kernel, sometimes by systems program
- Sometimes multiple flavors implemented – shells
- Primarily fetches a command from user and executes it
  - Sometimes commands built-in, sometimes just names of programs
    - If the latter, adding new features doesn't require shell modification



User Operating System Interface - GUI

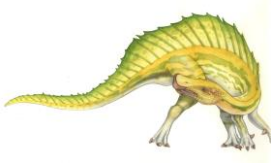
- User-friendly desktop metaphor interface
  - Usually mouse, keyboard, and monitor
  - Icons represent files, programs, actions, etc
  - Various mouse buttons over objects in the interface cause various actions (provide information, options, execute function, open directory (known as a folder)
  - Invented at Xerox PARC
- Many systems now include both CLI and GUI interfaces
  - Microsoft Windows is GUI with CLI “command” shell
  - Apple Mac OS X as “Aqua” GUI interface with UNIX kernel underneath and shells available
  - Solaris is CLI with optional GUI interfaces (Java Desktop, KDE)

Bourne Shell Command Interpreter

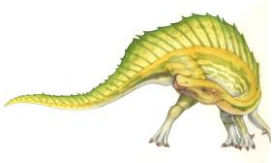
Terminal

File Edit View Terminal Tabs Help

```
fd0       0.0    0.0    0.0    0.0  0.0  0.0    0.0   0   0
sd0       0.0    0.2    0.0    0.2  0.0  0.0    0.4   0   0
sd1       0.0    0.0    0.0    0.0  0.0  0.0    0.0   0   0
                 extended device statistics
device    r/s    w/s   kr/s   kw/s wait actv  svc_t  %w  %b
fd0       0.0    0.0    0.0    0.0  0.0  0.0    0.0   0   0
sd0       0.6    0.0   38.4    0.0  0.0  0.0    8.2   0   0
sd1       0.0    0.0    0.0    0.0  0.0  0.0    0.0   0   0
(root@pbg-nv64-vm)-(11/pts)-(00:53 15-Jun-2007)-(global)
-(/var/tmp/system-contents/scripts)# swap -sh
total: 1.1G allocated + 190M reserved = 1.3G used, 1.6G available
(root@pbg-nv64-vm)-(12/pts)-(00:53 15-Jun-2007)-(global)
-(/var/tmp/system-contents/scripts)# uptime
 12:53am  up 9 min(s),  3 users,  load average: 33.29, 67.68, 36.81
(root@pbg-nv64-vm)-(13/pts)-(00:53 15-Jun-2007)-(global)
-(/var/tmp/system-contents/scripts)# w
  4:07pm  up 17 day(s), 15:24,  3 users,  load average: 0.09, 0.11, 8.66
User     tty           login@  idle   JCPU   PCPU  what
root     console      15Jun0718days      1         /usr/bin/ssh-agent -- /usr/bi
n/d
root     pts/3        15Jun07           18      4  w
root     pts/4        15Jun0718days                w
(root@pbg-nv64-vm)-(14/pts)-(16:07 02-Jul-2007)-(global)
-(/var/tmp/system-contents/scripts)#
```



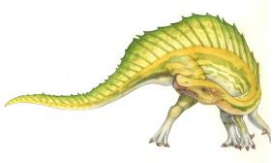
The Mac OS X GUI



System Calls

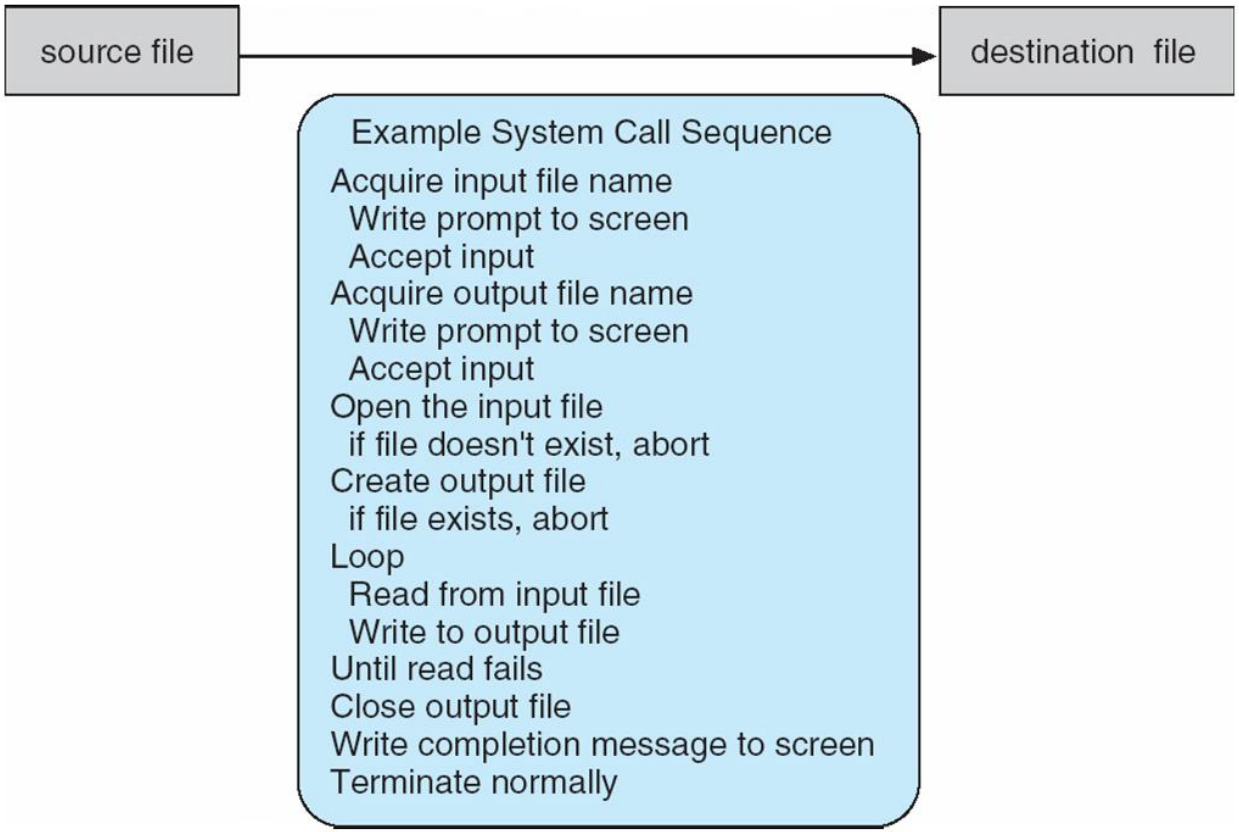
- Programming interface to the services provided by the OS
- Typically written in a high-level language (C or C++)
- Mostly accessed by programs via a high-level Application Program Interface (API) rather than direct system call use
- Three most common APIs are Win32 API for Windows, POSIX API for POSIX-based systems (including virtually all versions of UNIX, Linux, and Mac OS X), and Java API for the Java virtual machine (JVM)
- Why use APIs rather than system calls?

(Note that the system-call names used throughout this text are generic)



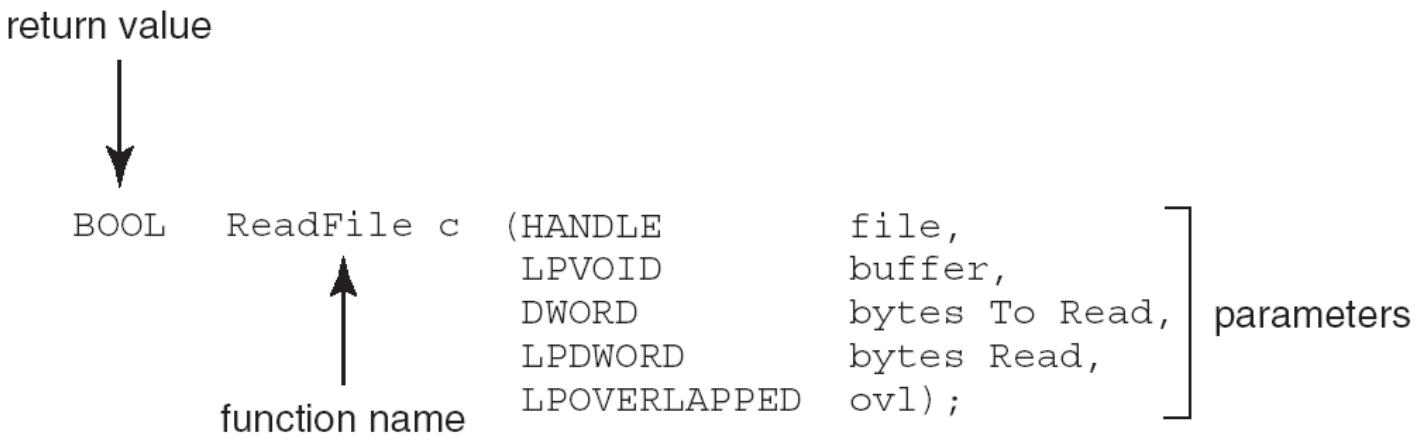
Example of System Calls

- System call sequence to copy the contents of one file to another file



Example of Standard API

- Consider the ReadFile() function in the
- Win32 API—a function for reading from a file

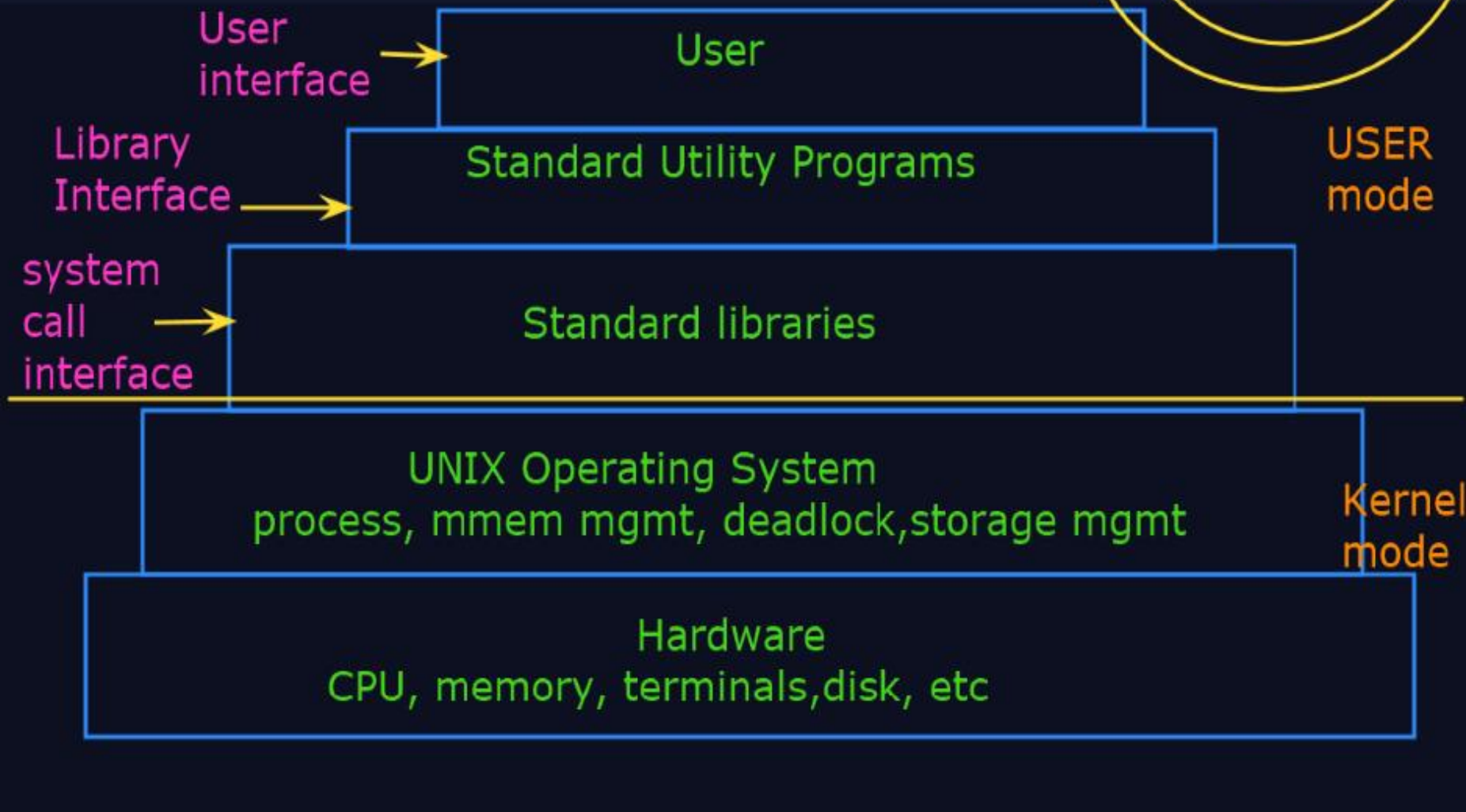


- A description of the parameters passed to ReadFile()
  - HANDLE file—the file to be read
  - LPVOID buffer—a buffer where the data will be read into and written from
  - DWORD bytesToRead—the number of bytes to be read into the buffer
  - LPDWORD bytesRead—the number of bytes read during the last read
  - LPOVERLAPPED ovl—indicates if overlapped I/O is being used

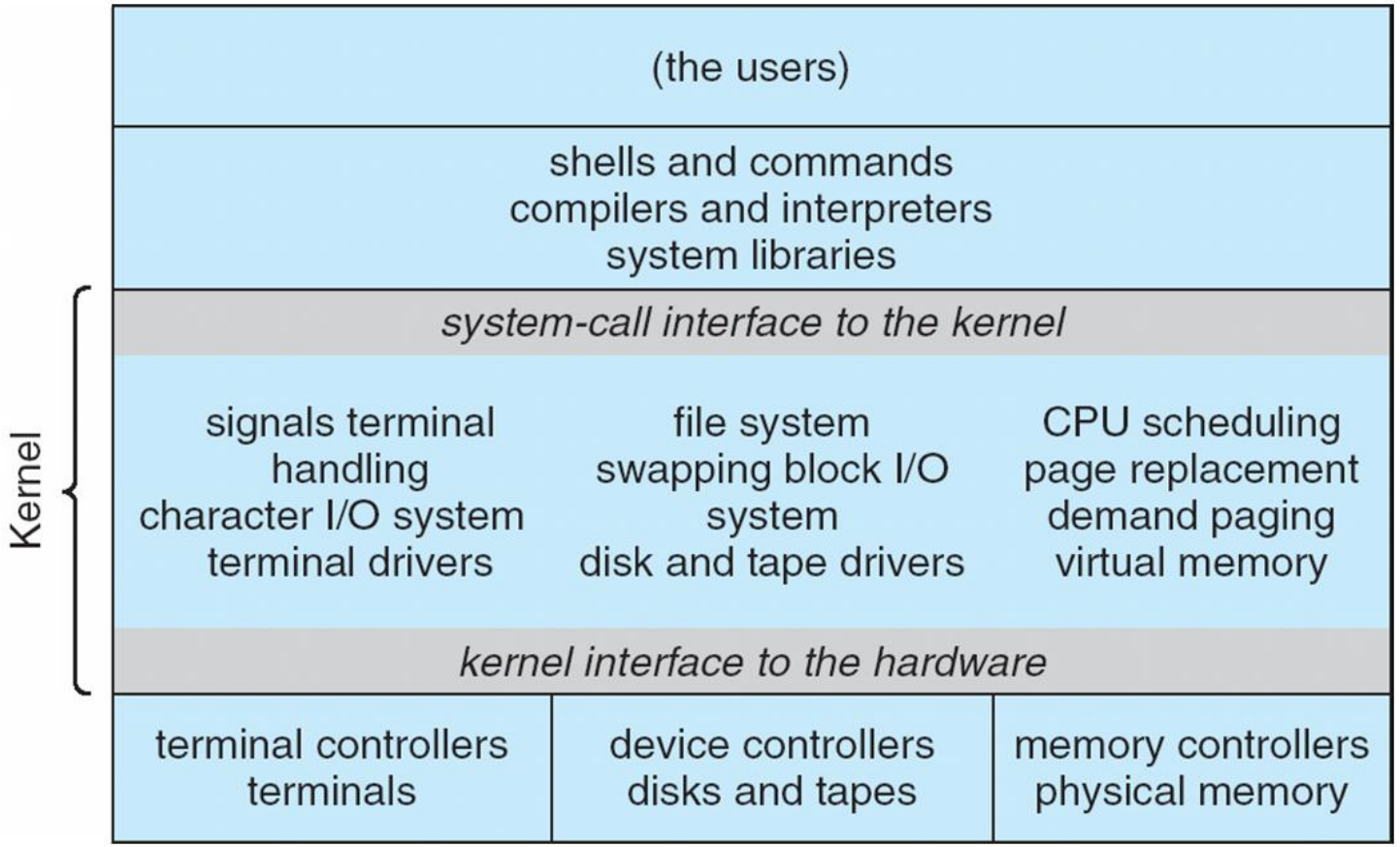
-Centos

Who can see what you share here? Recording On

Architecture:
- -Hardware
- -Kernel
- -System call interface
- -Application/Commands



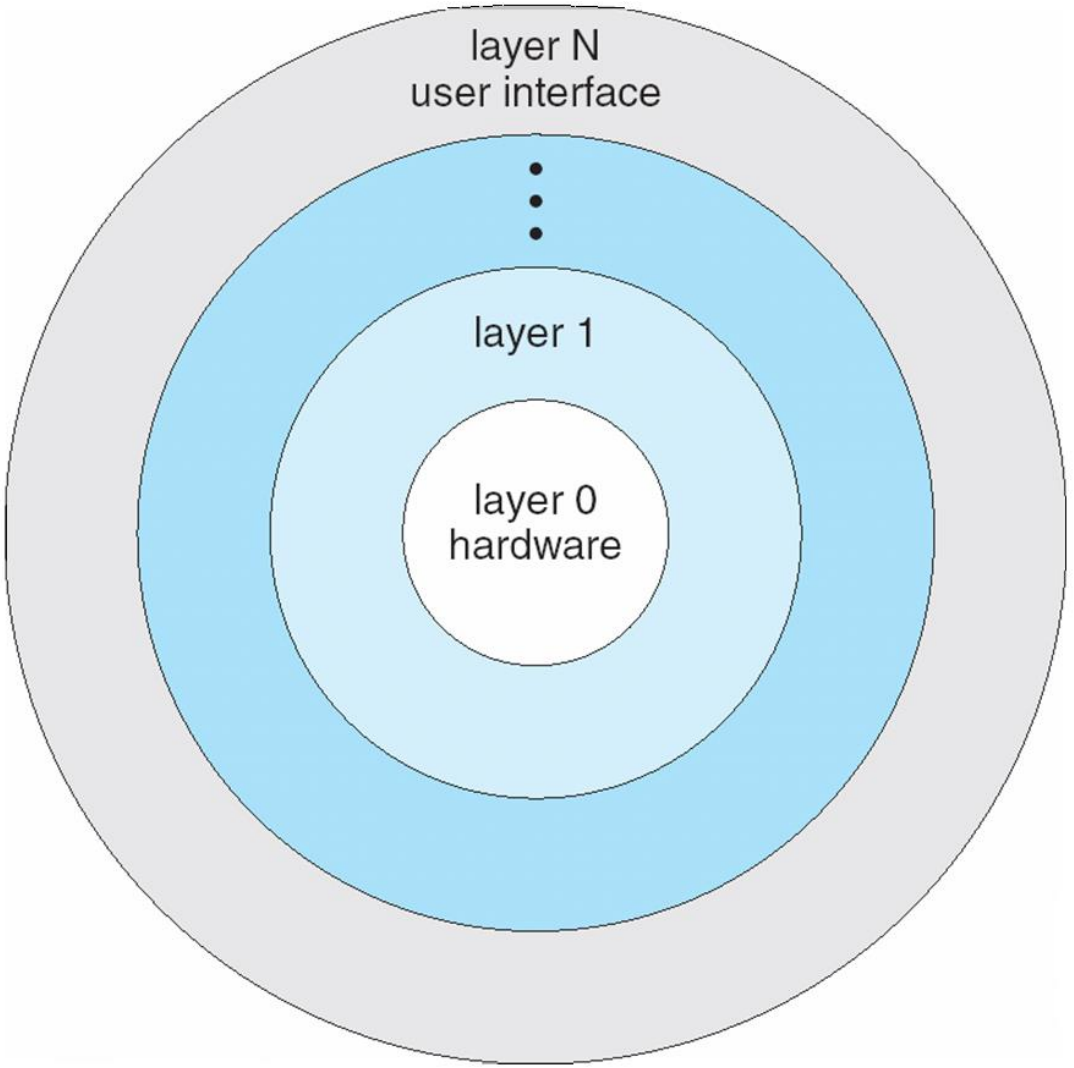
Traditional UNIX System Structure



UNIX

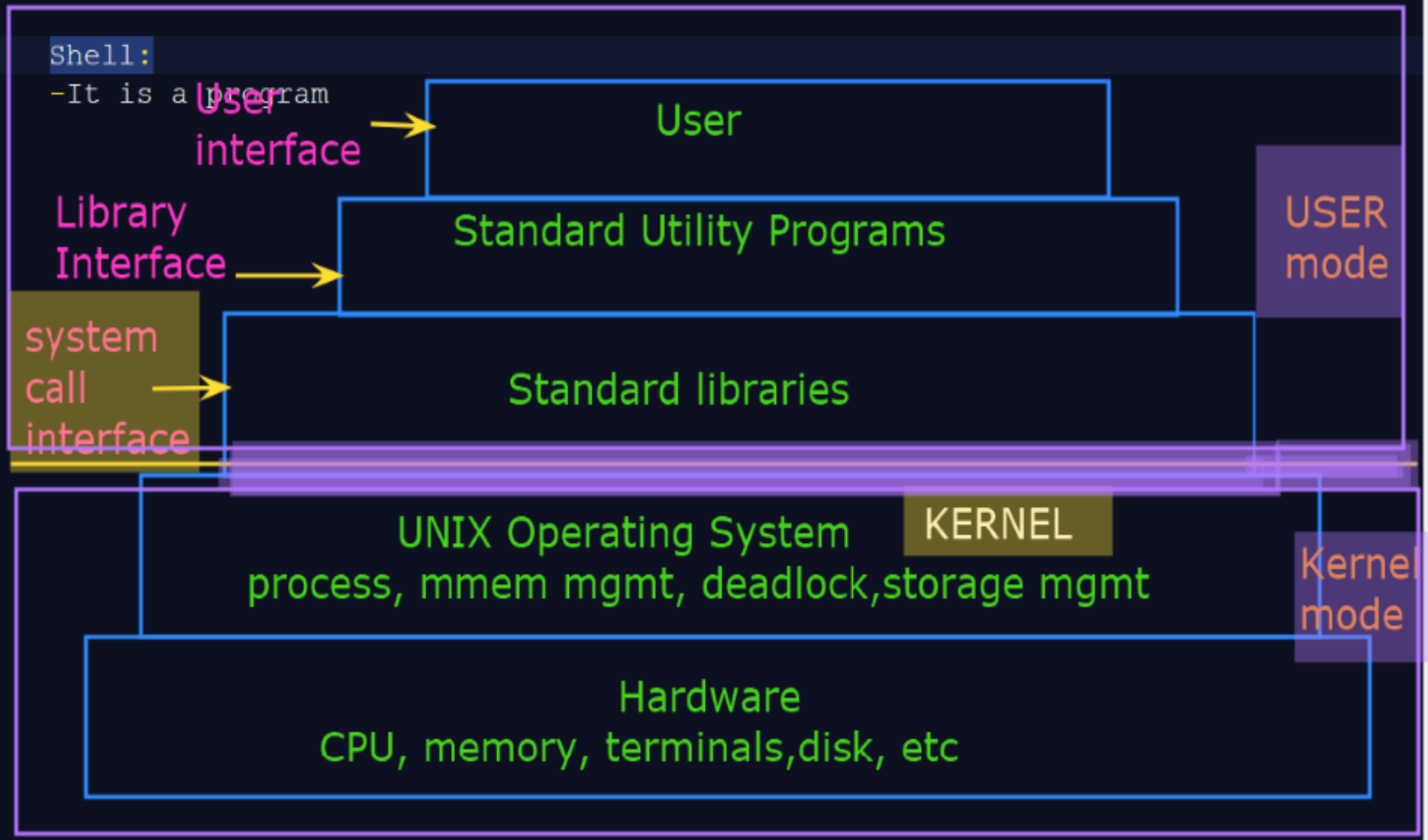
- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring. The UNIX OS consists of two separable parts
  - Systems programs
  - The kernel
    - Consists of everything below the system-call interface and above the physical hardware
    - Provides the file system, CPU scheduling, memory management, and other operating-system functions; a large number of functions for one level

Layered Operating System



Architecture:
- -Hardware
- -Kernel
- -System call interface
- -Application/Commands

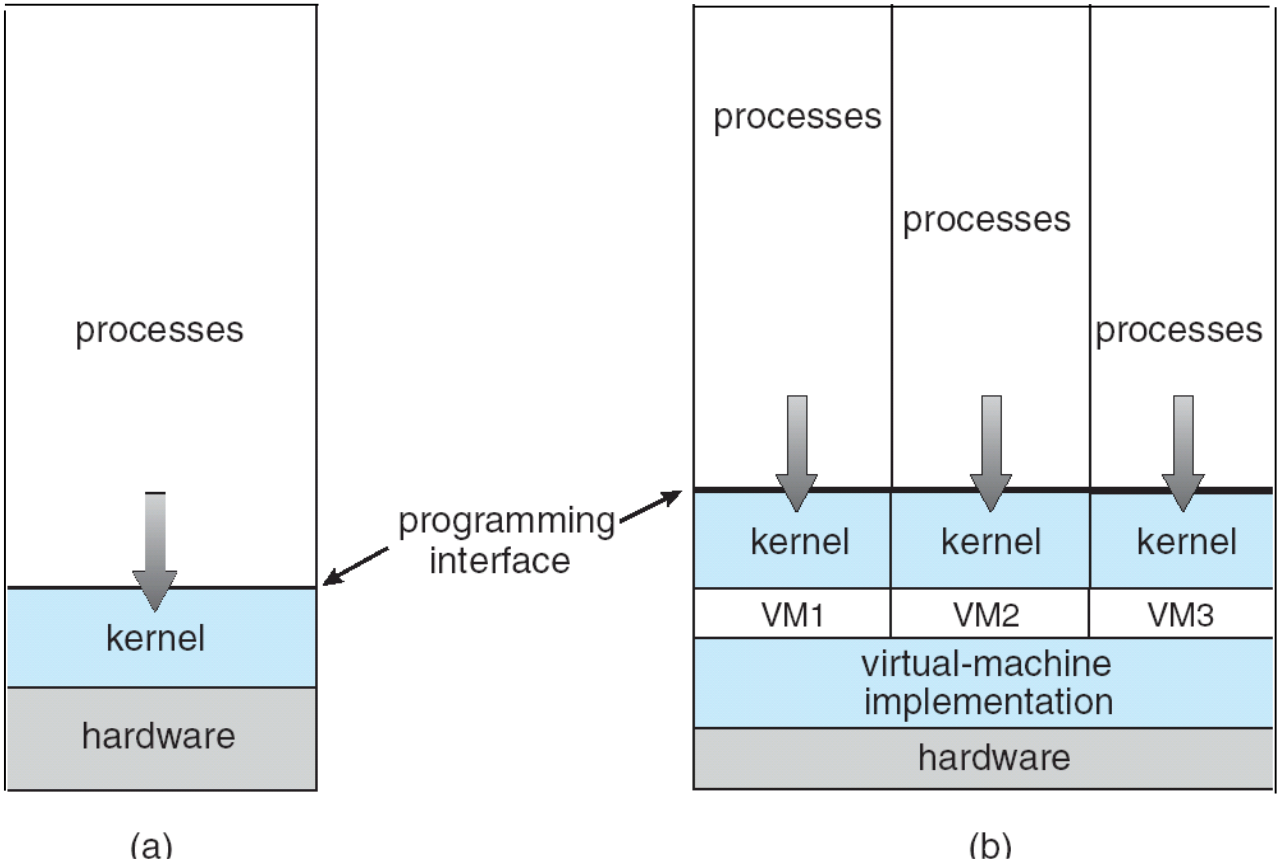
Shell:
-It is a program



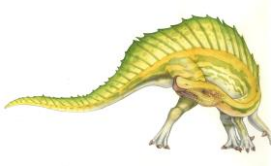
Virtual Machines

- A virtual machine takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware
- A virtual machine provides an interface *identical* to the underlying bare hardware
- The operating system host creates the illusion that a process has its own processor and (virtual memory)
- Each guest provided with a (virtual) copy of underlying computer

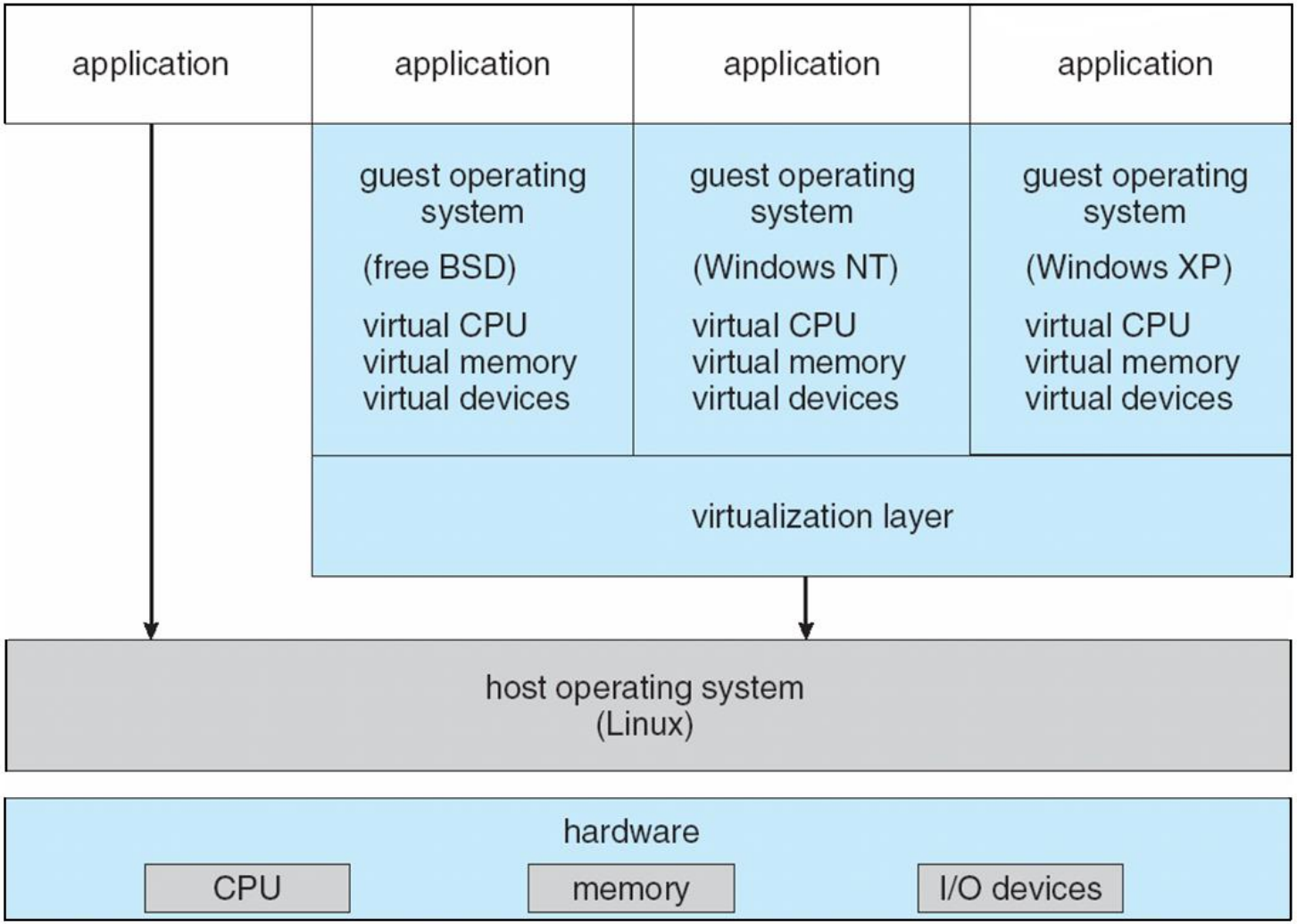
Virtual Machines (Cont)



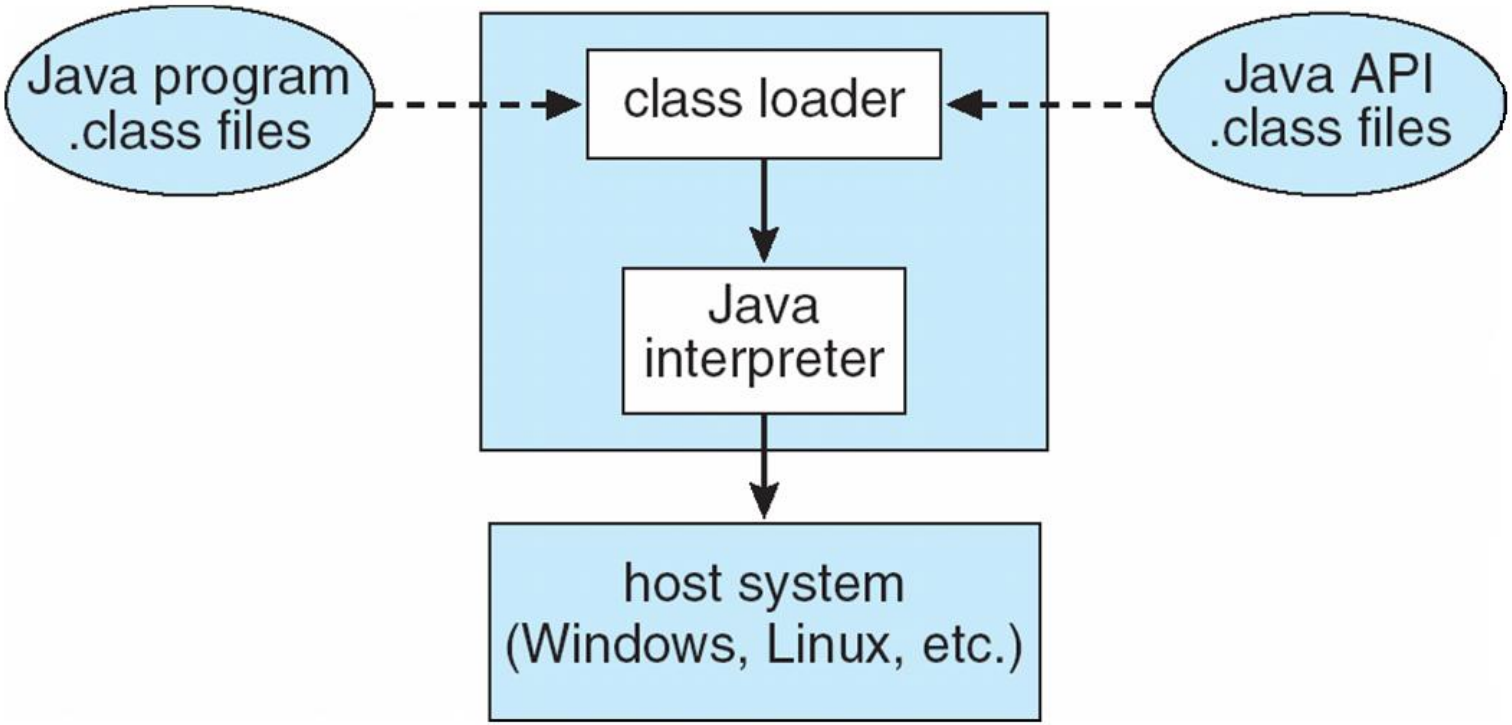
(a) Nonvirtual machine (b) virtual machine



VMware Architecture

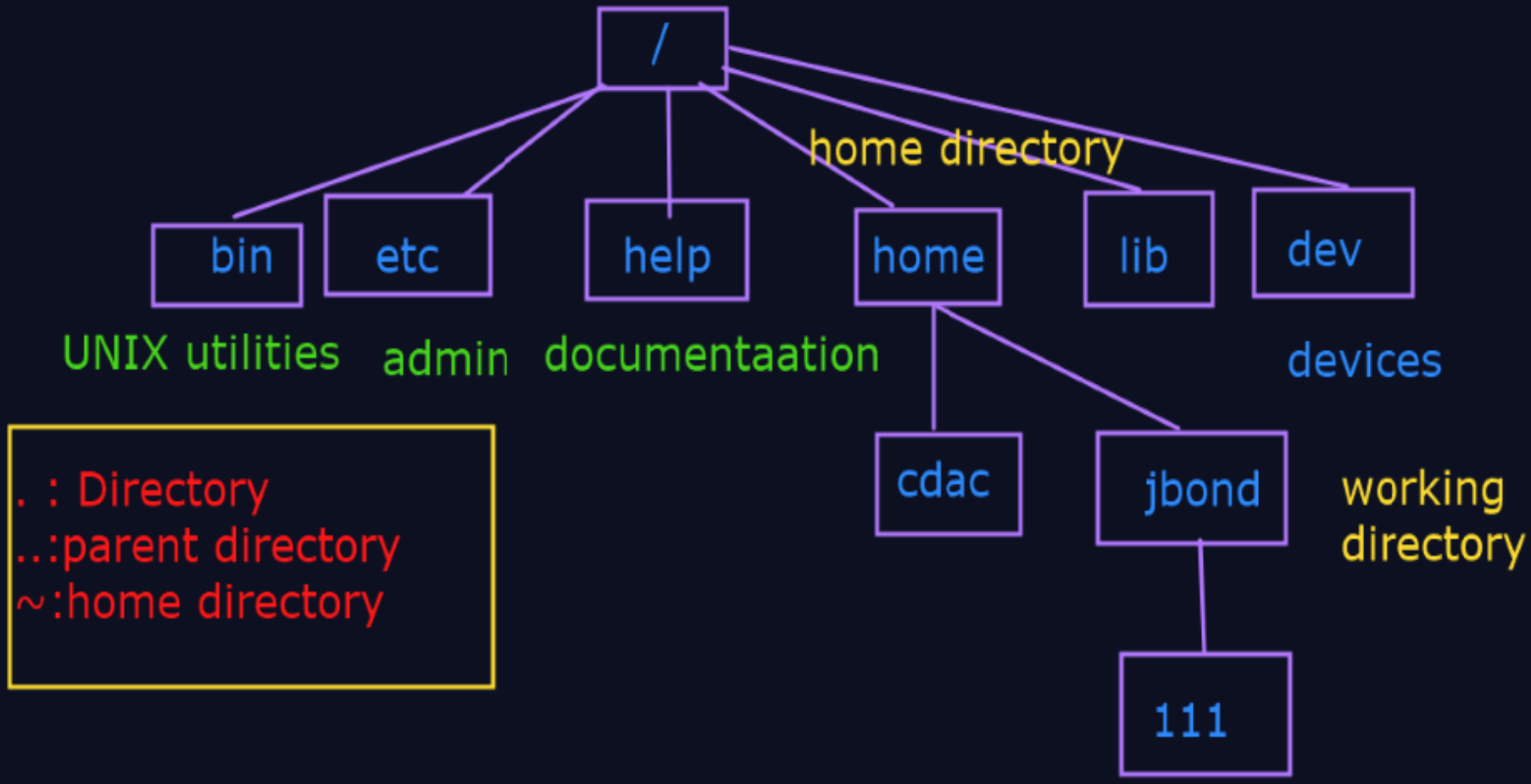


The Java Virtual Machine



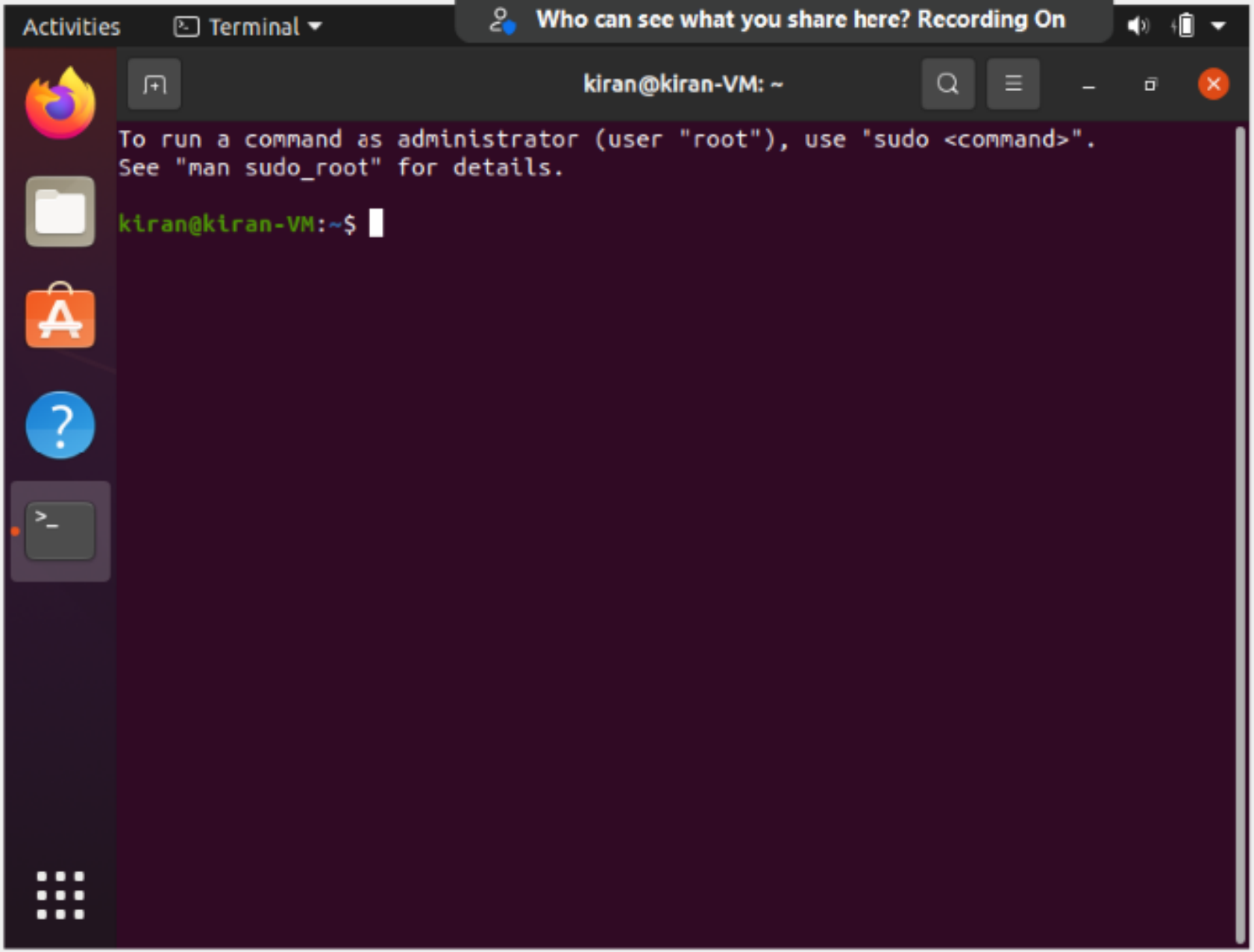
Hierarchical structure of File Organization:

-organizing files in ordered view.

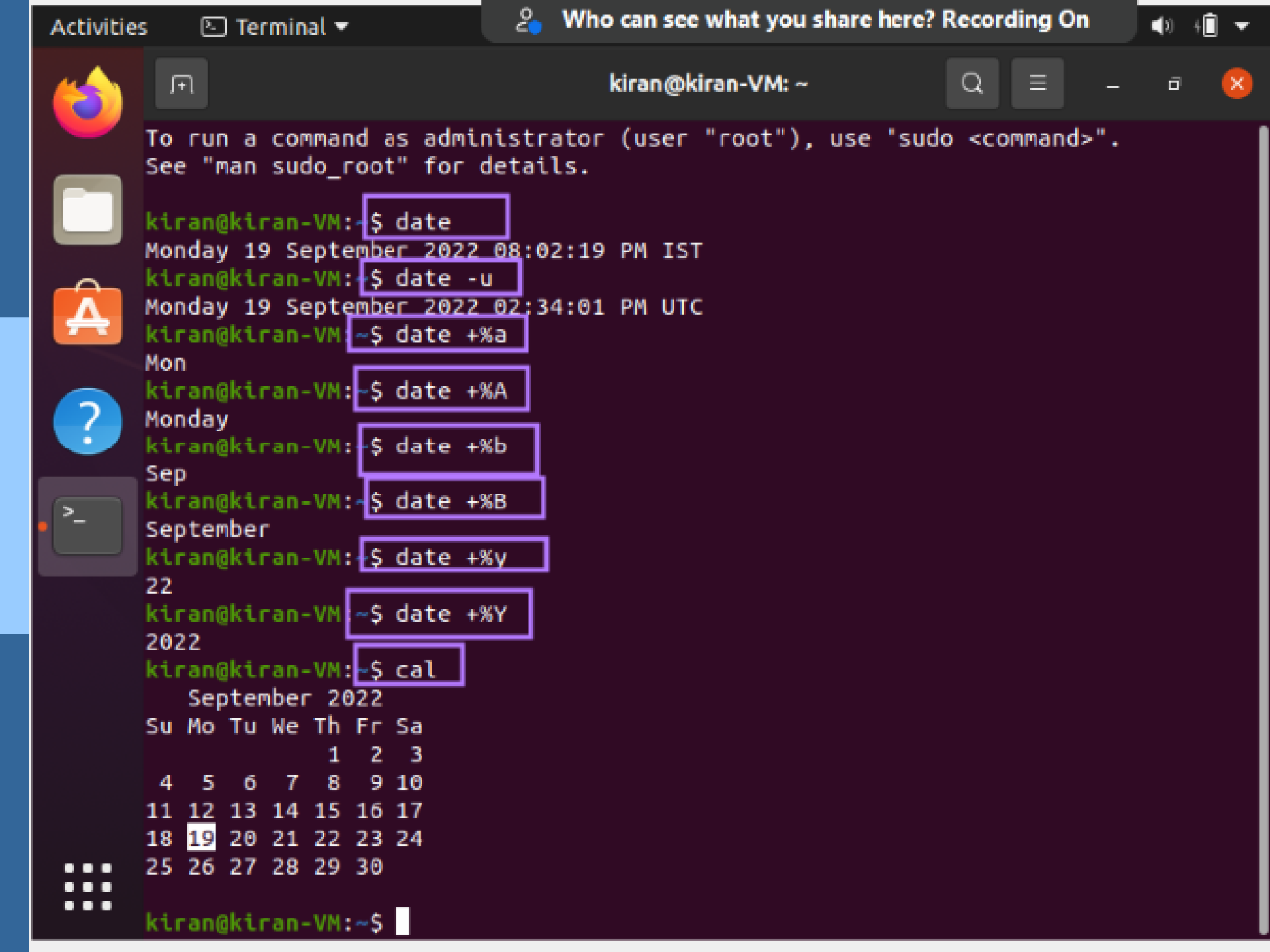


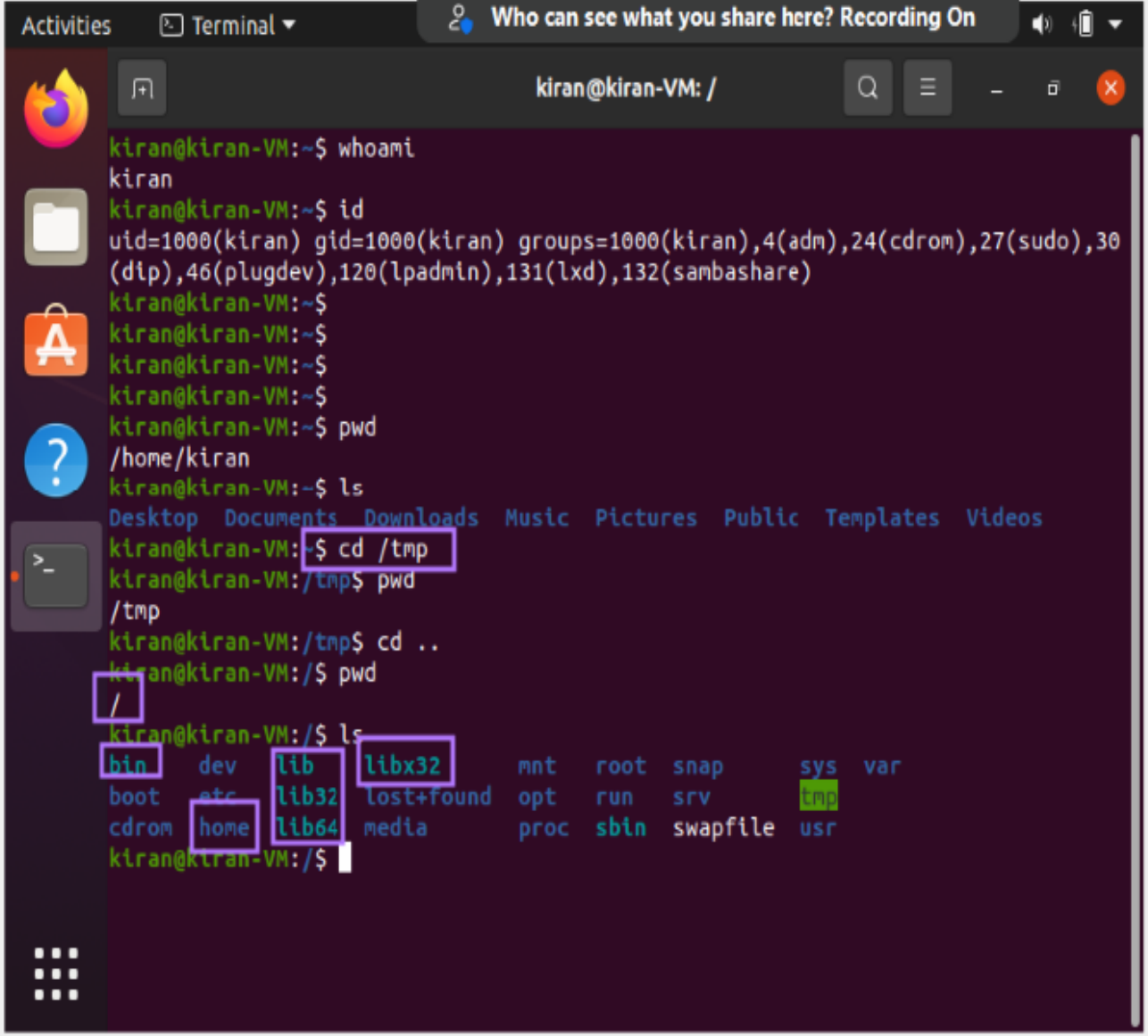
Directory files: branches of the tree

Regular files : leaves in the tree



username@hostname directory


Activities
Terminal
Who can see what you share here? Recording On
kiran@kiran-VM: ~
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
kiran@kiran-VM:~$ date
Monday 19 September 2022 08:02:19 PM IST
kiran@kiran-VM:~$ date -u
Monday 19 September 2022 02:34:01 PM UTC
kiran@kiran-VM:~$ date +%a
Mon
kiran@kiran-VM:~$ date +%A
Monday
kiran@kiran-VM:~$ date +%b
Sep
kiran@kiran-VM:~$ date +%B
September
kiran@kiran-VM:~$ date +%y
22
kiran@kiran-VM:~$ date +%Y
2022
kiran@kiran-VM:~$ cal
September 2022
Su Mo Tu We Th Fr Sa
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30
kiran@kiran-VM:~$



man cal
Exit:q

pwd: print working directory

cd:change directory