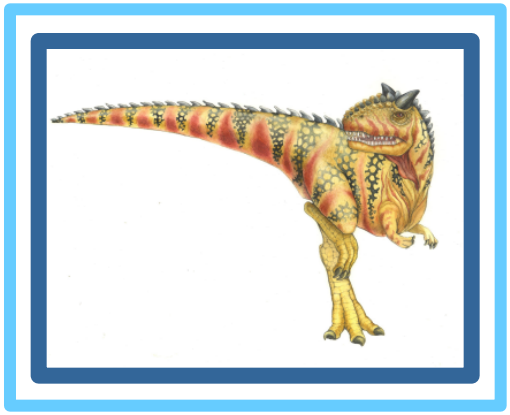
Chapter 2: Operating-System Structures

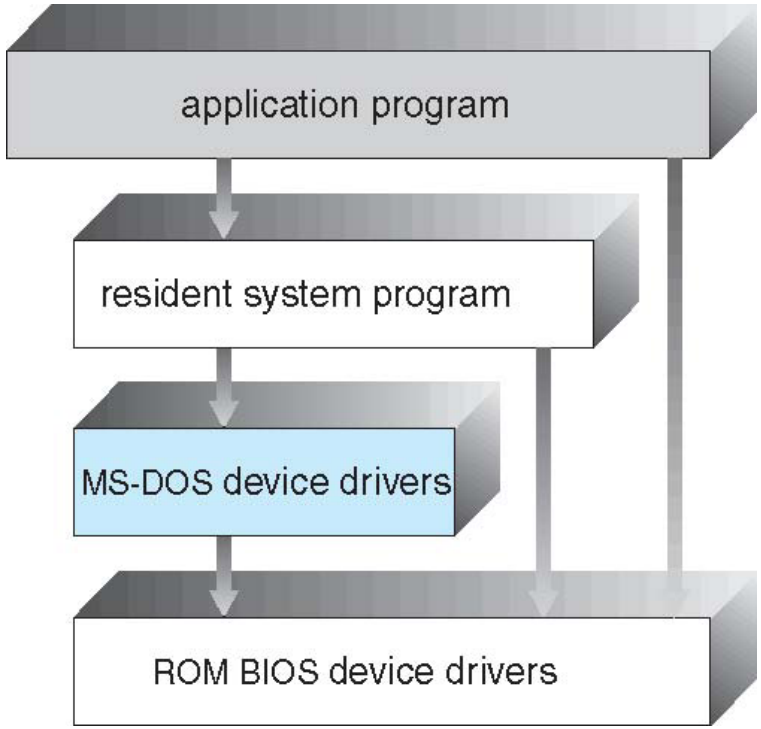


Operating System Structure

- General-purpose OS is very large program
- Types of structures are,
  - Simple — MS-DOS
  - Monolithic — UNIX
  - Layered – an abstraction
  - Microkernel — Mach
  - Modules

Simple Structure -- MS-DOS

- MS-DOS – written to provide the most functionality in the least space
  - Not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated

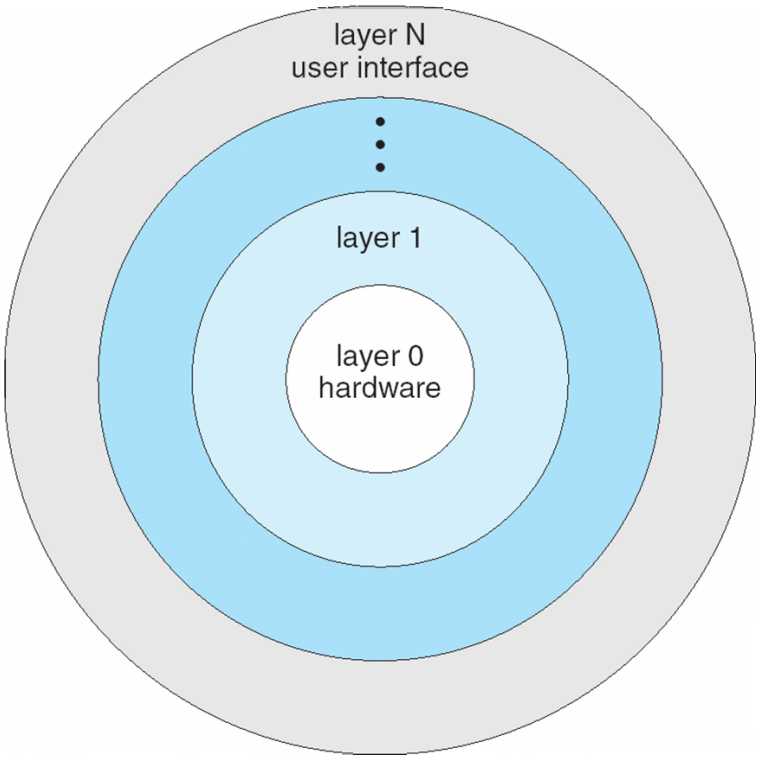


Monolithic Structure -- UNIX

- 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 Approach

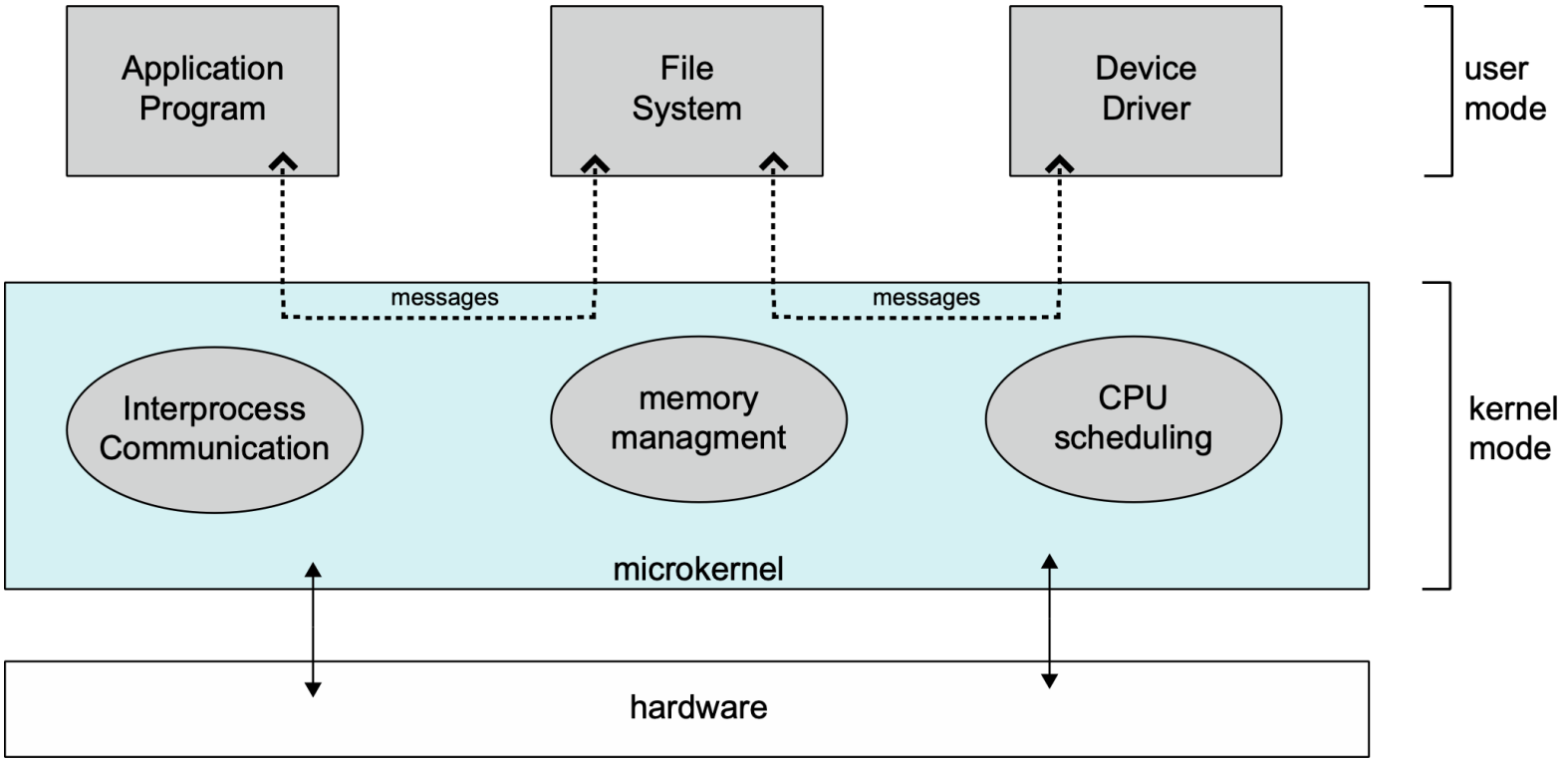
- The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers



Microkernel System Structure

- Moves as much from the kernel into user space
- **Mach** example of **microkernel**
  - Mac OS X kernel (**Darwin**) partly based on Mach
- Communication takes place between user modules using **message passing**
- Benefits:
  - Easier to extend a microkernel
  - Easier to port the operating system to new architectures
  - More reliable (less code is running in kernel mode)
  - More secure
- Detriments:
  - Performance overhead of user space to kernel space communication

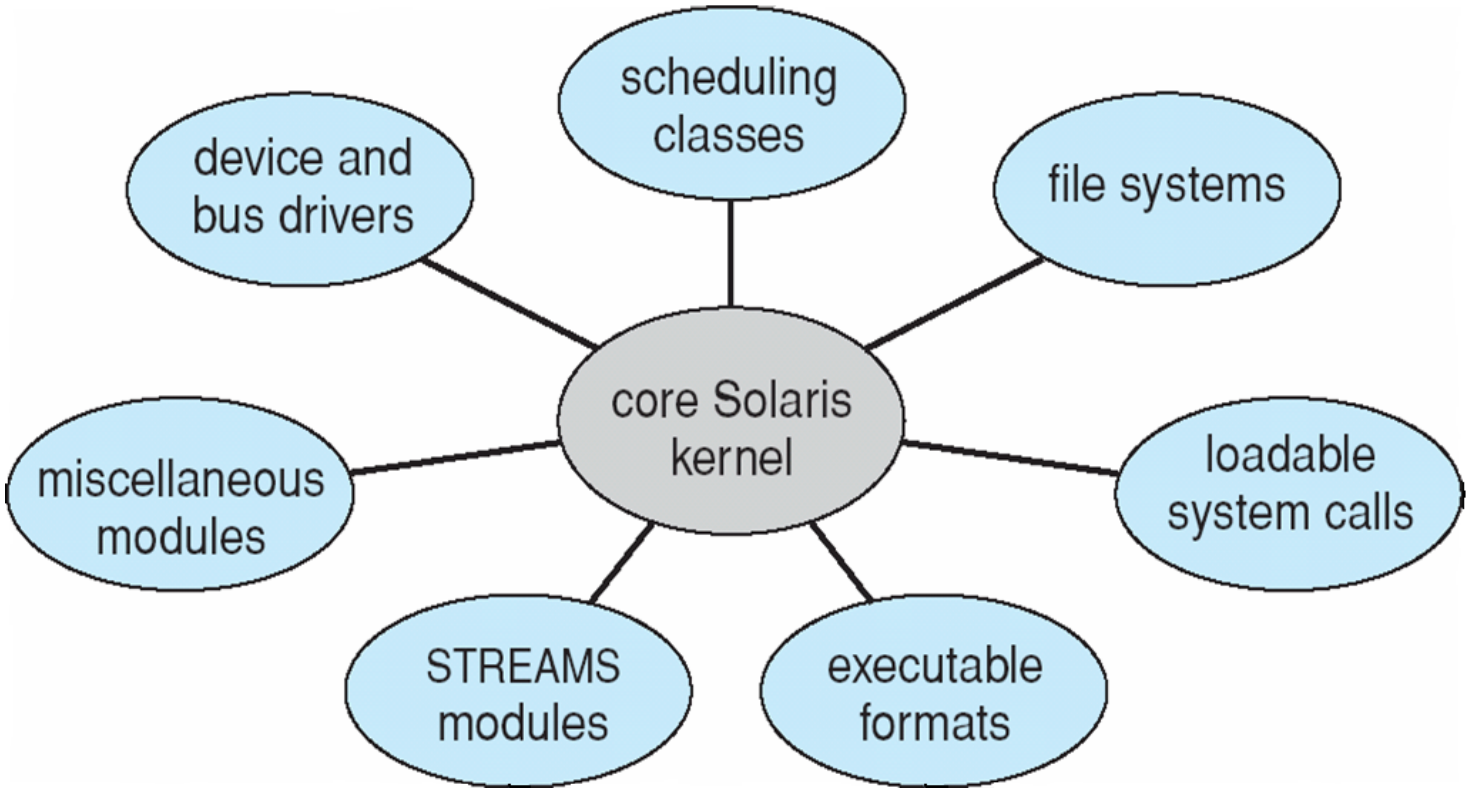
Microkernel System Structure



Modules

- Many modern operating systems implement **loadable kernel modules — LKMs**
  - Uses object-oriented approach
  - The core services are designed with the kernel
  - Additional services are added as LKMs
- Overall, similar to layers but with more flexible
  - Linux, Solaris, etc

Solaris Modular Approach



Hybrid Systems

- Most modern operating systems are actually not one pure model
  - Hybrid combines multiple approaches to address performance, security, usability needs
  - Linux and Solaris kernels in kernel address space, so monolithic, plus modular for dynamic loading of functionality
  - Windows mostly monolithic, plus microkernel

  - Eg: Apple Mac OS, iOS, Android

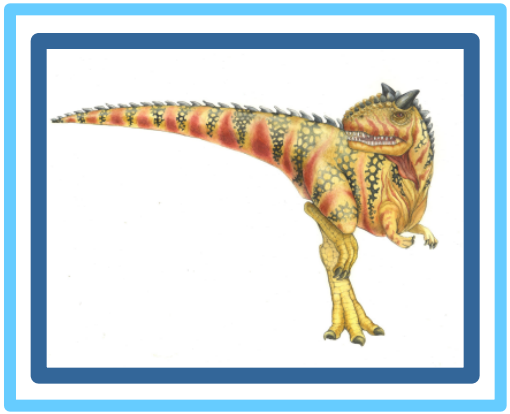
Operating-System Debugging

- **Debugging** is finding and fixing errors, or **bugs**
- OS generate **log files** containing error information
- Failure of an application can generate **core dump** file capturing memory of the process
- Operating system failure can generate **crash dump** file containing kernel memory

System Boot

- When power initialized on system, execution starts at a fixed memory location
  - Firmware ROM used to hold initial boot code
- Operating system must be made available to hardware so hardware can start it
  - Small piece of code – **bootstrap loader**, stored in **ROM** locates the kernel, loads it into memory, and starts it
  - Sometimes two-step process where **boot block** at fixed location loaded by ROM code, which loads bootstrap loader from disk

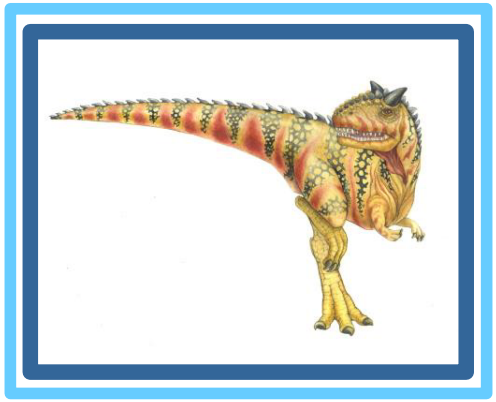
End of Chapter 2



Quiz

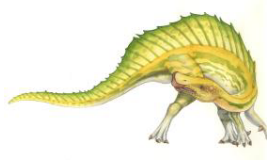
1. The OS design requirements belong to two groups —> …… and ……..
2. What is the main principle to follow while implementing an OS?
3. Different structures ?
4. What are the disadvantages of layered approach?
5. What is the communication type used in microkernel system structure?
6. The process of finding and fixing of errors is called as ……..?
7. The small piece of code that help to locate the kernel is known as ……..?

Chapter 3: Processes



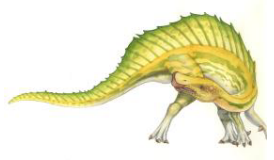
Chapter 3: Processes

- Process Concept
- Process Scheduling
- Operations on Processes
- Interprocess Communication
- Examples of IPC Systems
- Communication in Client-Server Systems



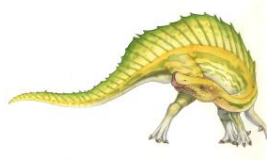
Objectives

- To introduce the notion of a process -- a program in execution, which forms the basis of all computation
- To describe the various features of processes, including scheduling, creation and termination, and communication
- To explore interprocess communication using shared memory and message passing
- To describe communication in client-server systems



Process

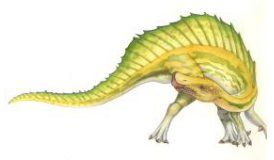
- Program is ***passive*** entity stored on disk (**executable file**), process is ***active***
  - Program becomes process when executable file loaded into memory
- Execution of program started via GUI mouse clicks, command line entry of its name, etc
- One program can be several processes
  - Consider multiple users executing the same program



Process Vs Threads

- An operating system executes a variety of programs
- **Process** – a program in execution
- **Thread** — Unit of execution within a process, A single process can have any number of threads.

- Task manager in Windows
- Activity monitor in Mac



Process Vs Threads

Activity Monitor (My Processes)

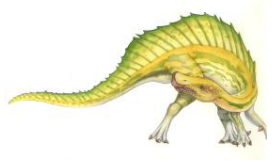
CPU | Memory | Energy | Disk | Network | Search

Process Name	% CPU	CPU Time	Threads	Idle Wake Ups	% GPU	GPU Time
photoanalysisd	23.7	23:13:02.31	4	15	0.0	0.00
Activity Monitor	11.8	13.44	5	1	0.0	0.00
screencapture	2.3	0.74	2	0	0.0	0.00
Atom	1.0	27:42.52	28	21	0.0	0.00
Pages	0.6	40:16.62	8	2	0.0	7.61
Microsoft Teams Helper (Renderer)	0.5	26:51.96	19	7	0.0	0.00
https://www.youtube.com	0.4	6:55.59	13	0	0.0	26.72
https://softwarekeep.com	0.3	21.25	10	0	0.0	0.42
Google Chrome Helper (Renderer)	0.3	20:15.91	16	1	0.0	0.00
Screen Shot	0.3	0.28	3	0	0.0	0.00
Safari Networking	0.2	24:34.73	11	0	0.0	0.00
com.apple.geod	0.2	1:50.92	6	1	0.0	0.00
https://www.vitalsource.com, ...	0.2	4:09.33	6	0	0.0	0.01
mdworker_shared	0.2	0.21	3	0	0.0	0.00
Microsoft Teams	0.1	12:03.21	40	1	0.0	0.00
cfprefsd	0.1	2:46.91	3	0	0.0	0.00
Keynote	0.1	8:30.54	3	1	0.0	2.30
https://drive.google.com	0.1	1:12.00	4	0	0.0	15.75
Mail	0.1	50:24.17	11	1	0.0	0.26
Google Chrome Helper (Renderer)	0.1	12.90	13	1	0.0	0.00
Grammarly for Safari (Safari)	0.1	7:09.53	7	1	0.0	0.00
Microsoft Teams Helper	0.0	4:27.51	8	0	0.0	0.00
Microsoft Teams Helper (Renderer)	0.0	11.14	12	0	0.0	0.00
https://bookshelf.vitalsource.com, ...	0.0	27:54.74	7	1	0.0	0.02
Google Chrome	0.0	1:01:03.42	26	1	0.0	0.00

System: 5.99%
User: 21.80%
Idle: 72.21%

CPU LOAD

Threads: 2,198
Processes: 518



Process State

- As a process executes, it changes **state**
  - **new**: The process is being created
  - **running**: Instructions are being executed
  - **waiting**: The process is waiting for some event to occur
  - **ready**: The process is waiting to be assigned to a processor
  - **terminated**: The process has finished execution

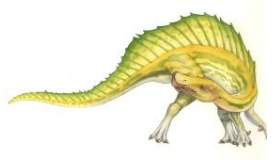
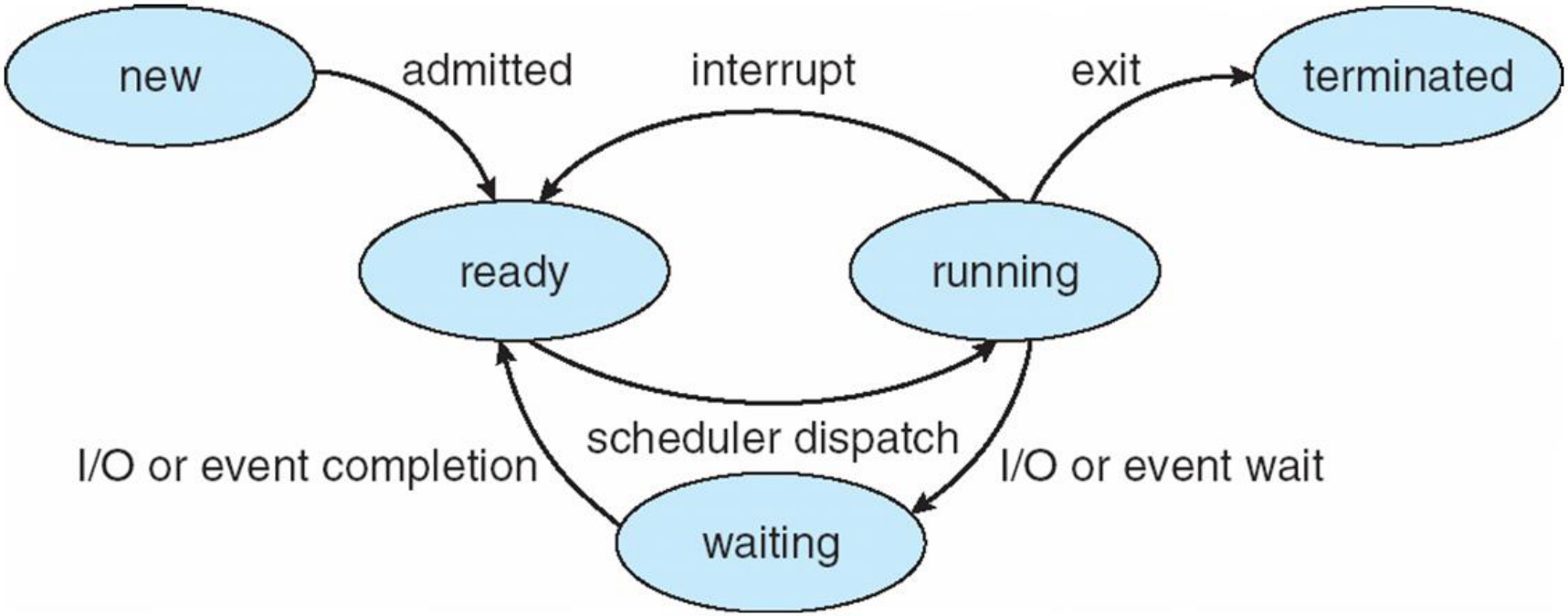
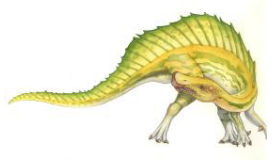


Diagram of Process State





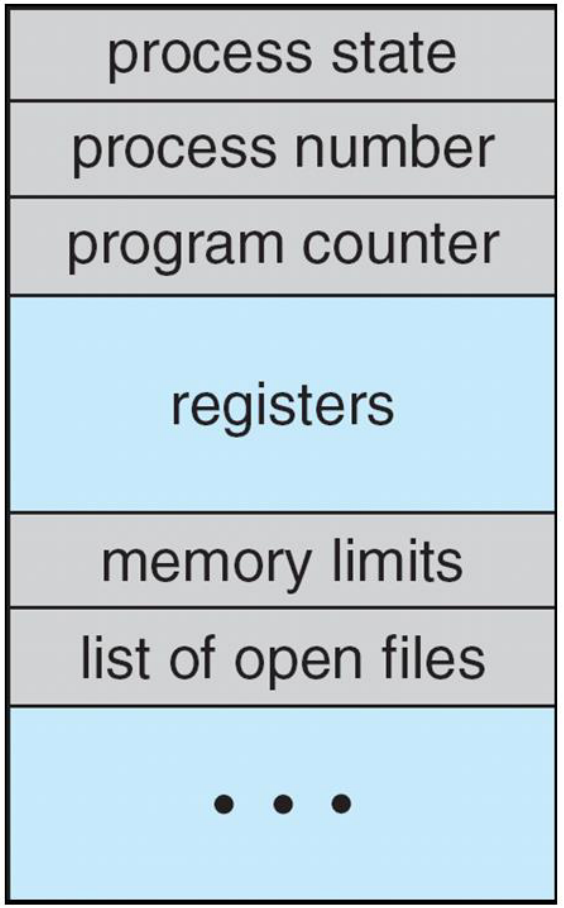
Process Control Block (PCB)

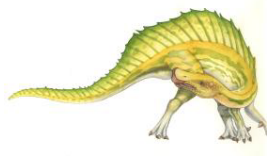
Each process can be represented in the OS by a PCB.

Information associated with each process

(also called **task control block**)

- Process state – running, waiting, etc
- Precess number — ID of the process
- Program counter – location of instruction to next execute, address of the next instruction in the code
- CPU registers – contents of all process-centric registers
- CPU scheduling information- priorities, scheduling queue pointers
- Memory-management information – memory allocated to the process
- Accounting information – CPU used, clock time elapsed since start, time limits
- I/O status information – I/O devices allocated to process, list of open files





Quiz

1. A single program can have multiple processes - T or F?
2. The unit of execution within a process is known as …….
3. In which state the process is waiting to be assigned to a processor?
4. The system does not do any useful task during ……
5. What is cascading termination?
6. Two methods for interprocess communication?