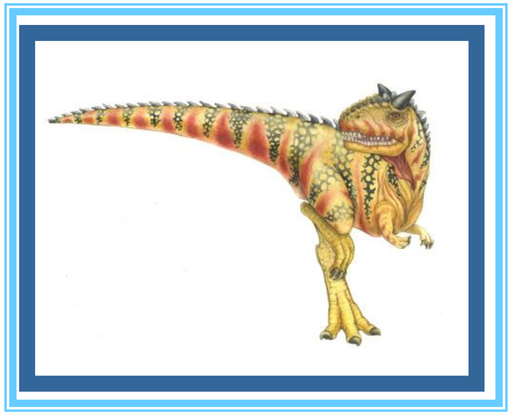
Processes
Day3: Sep 2021

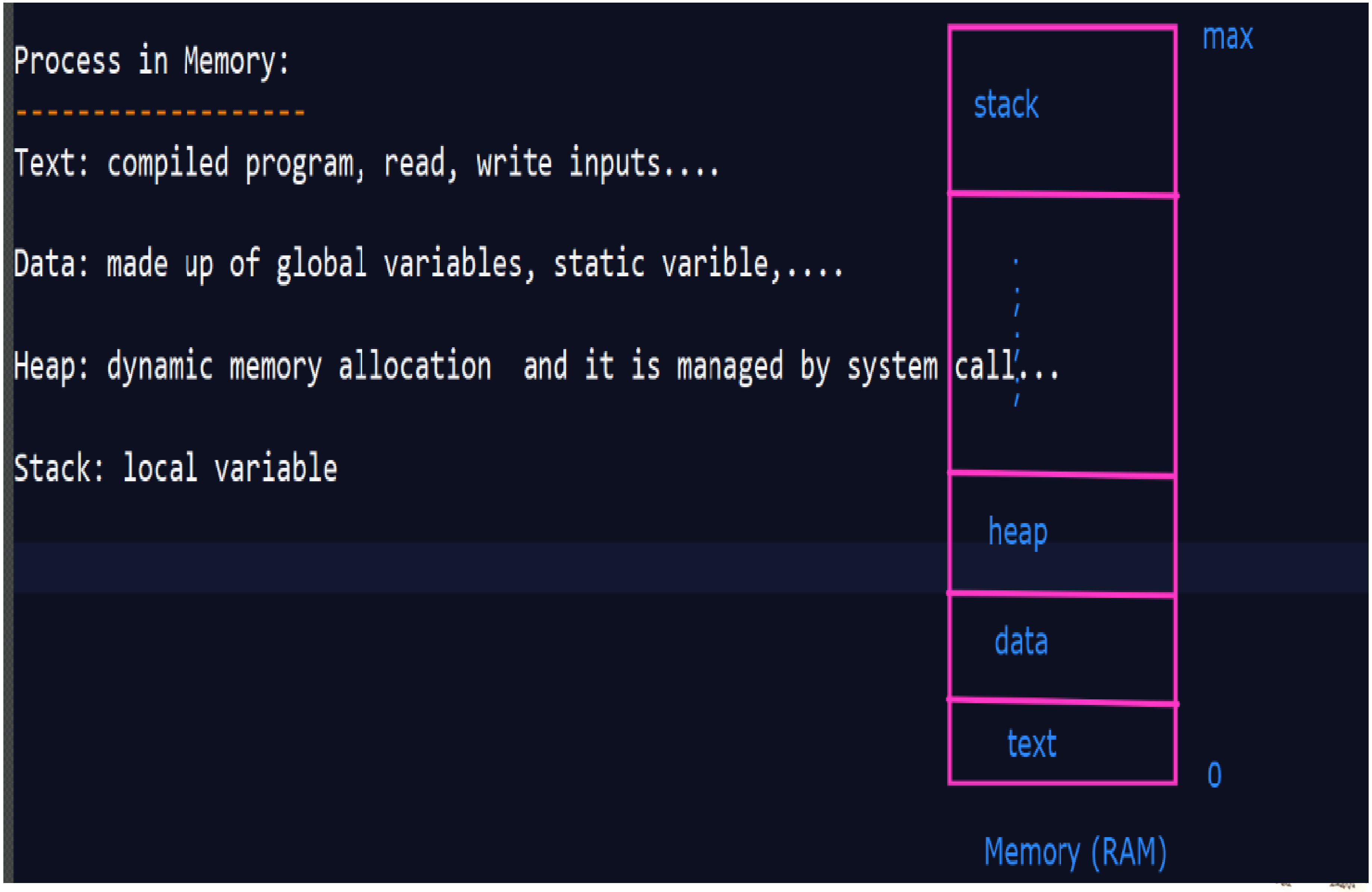
Kiran Waghmare



Agenda: Processes

- Preemptive and non preemptive
- Process mgmt
- Process life cycle
- Schedulers
- Scheduling algorithms
- Creation of fork, waitpid, exec system calls
- Orphan and zombie

Process in Memory



```
Process in Memory:
-------------------
Text: compiled program, read, write inputs....

Data: made up of global variables, static varible,....

Heap: dynamic memory allocation  and it is managed by system call...

Stack: local variable
```

Process in Operating System

- A process is a **program in execution** which then forms the basis of all computation.
- The process is not as same as program code but a lot more than it.
- A process is an 'active' entity as opposed to the program which is considered to be a 'passive' entity.
- Attributes held by the process include hardware state, memory, CPU, etc.

- **Process memory** is divided into four sections for efficient working :
- The **Text section** is made up of the compiled program code, read in from non-volatile storage when the program is launched.
- The **Data section** is made up of the global and static variables, allocated and initialized prior to executing the main.
- The **Heap** is used for the dynamic memory allocation and is managed via calls to new, delete, malloc, free, etc.
- The **Stack** is used for local variables. Space on the stack is reserved for local variables when they are declared.

Process Concept

- An operating system executes a variety of programs:
  - Batch system – jobs
  - Time-shared systems – user programs or tasks
- Textbook uses the terms *job* and *process* almost interchangeably
- **Process – a program in execution; process execution must progress in sequential fashion**
- A process includes:
  - program counter
  - stack
  - data section

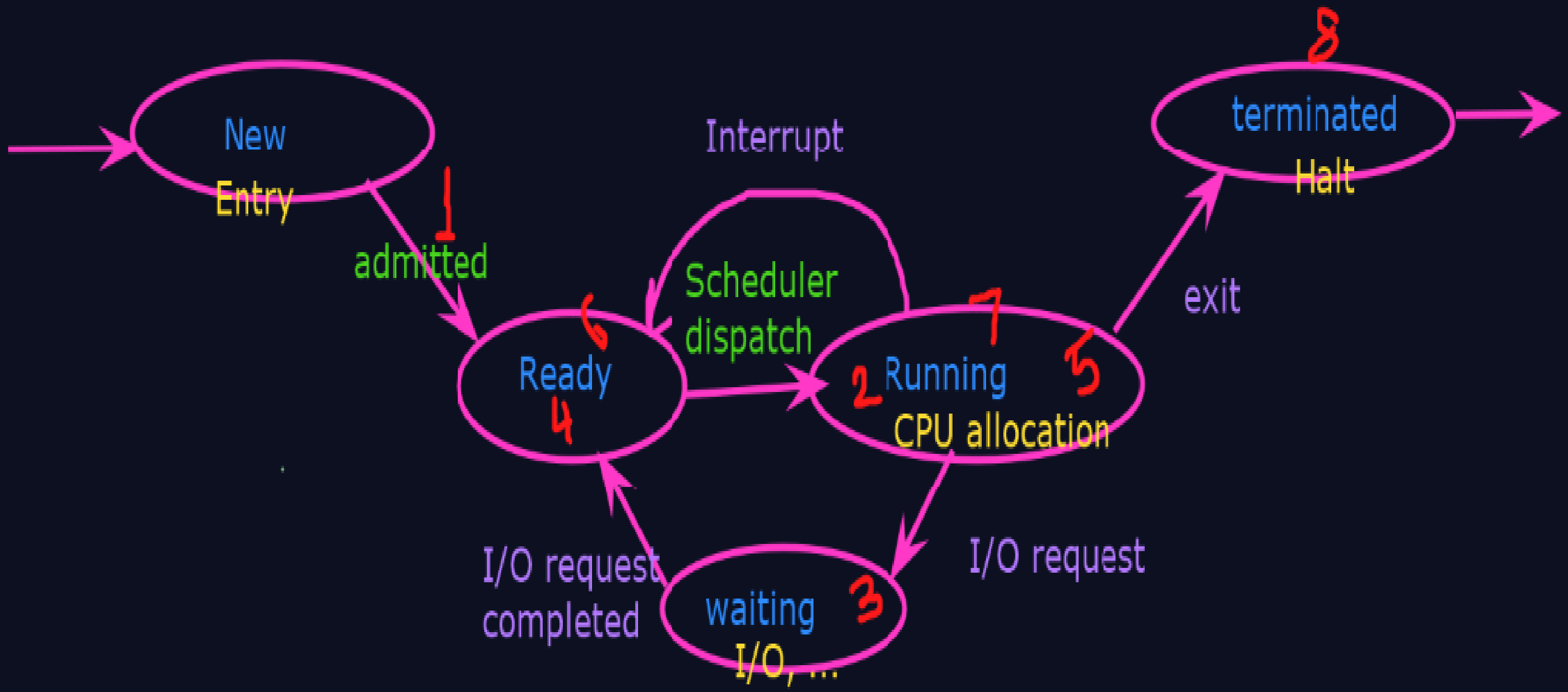
Process State

- 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

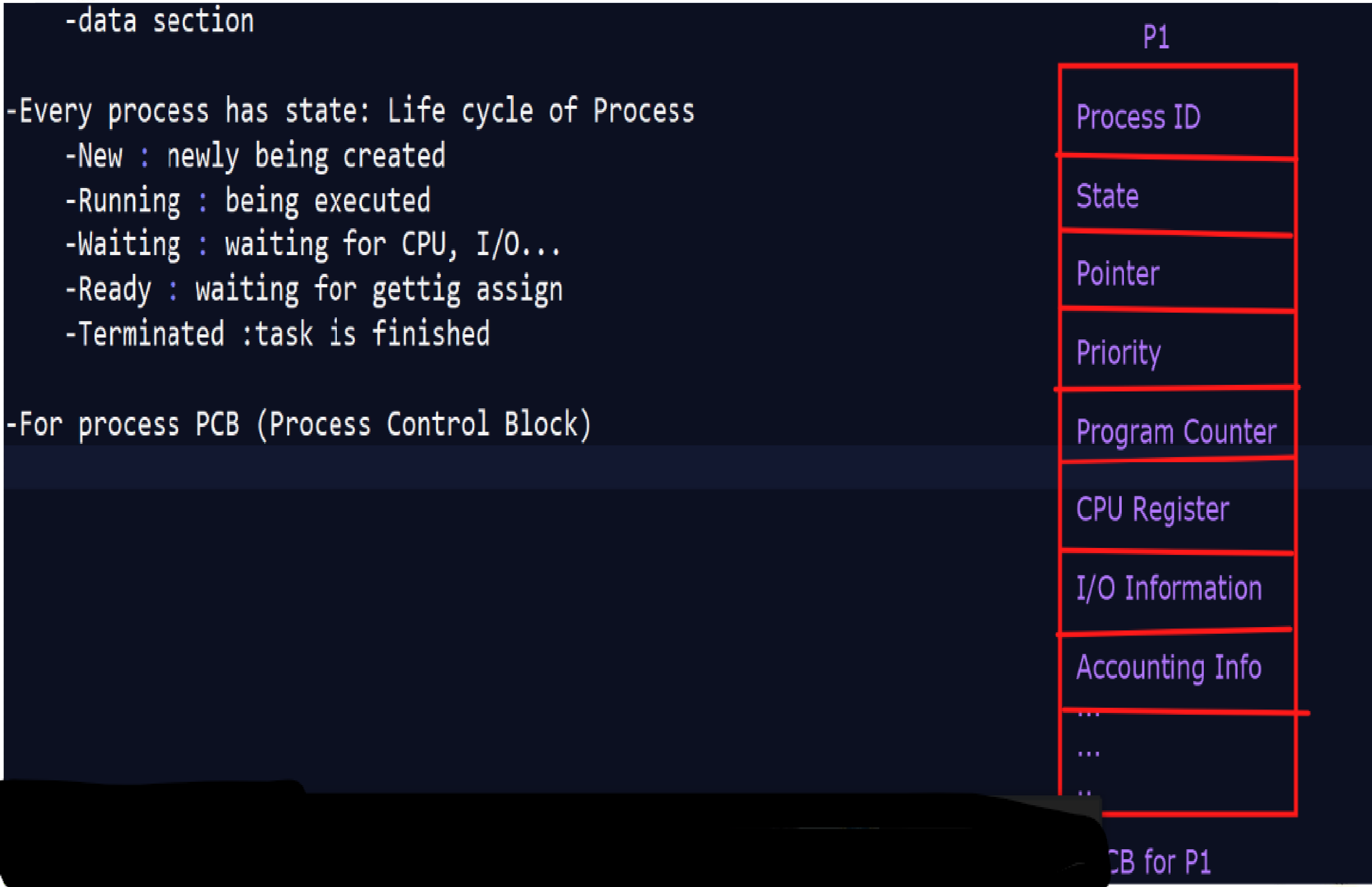
Diagram of Process State

-Every process has state: Life cycle of Process

- -New : newly being created
- -Running : being executed
- -Waiting : waiting for CPU, I/O...
- -Ready : waiting for gettig assign
- -Terminated :task is finished



Process Control Block (PCB)



```
    -data section

-Every process has state: Life cycle of Process
    -New : newly being created
    -Running : being executed
    -Waiting : waiting for CPU, I/O...
    -Ready : waiting for gettig assign
    -Terminated :task is finished

-For process PCB (Process Control Block)
```

P1

Process ID
State
Pointer
Priority
Program Counter
CPU Register
I/O Information
Accounting Info
... ...

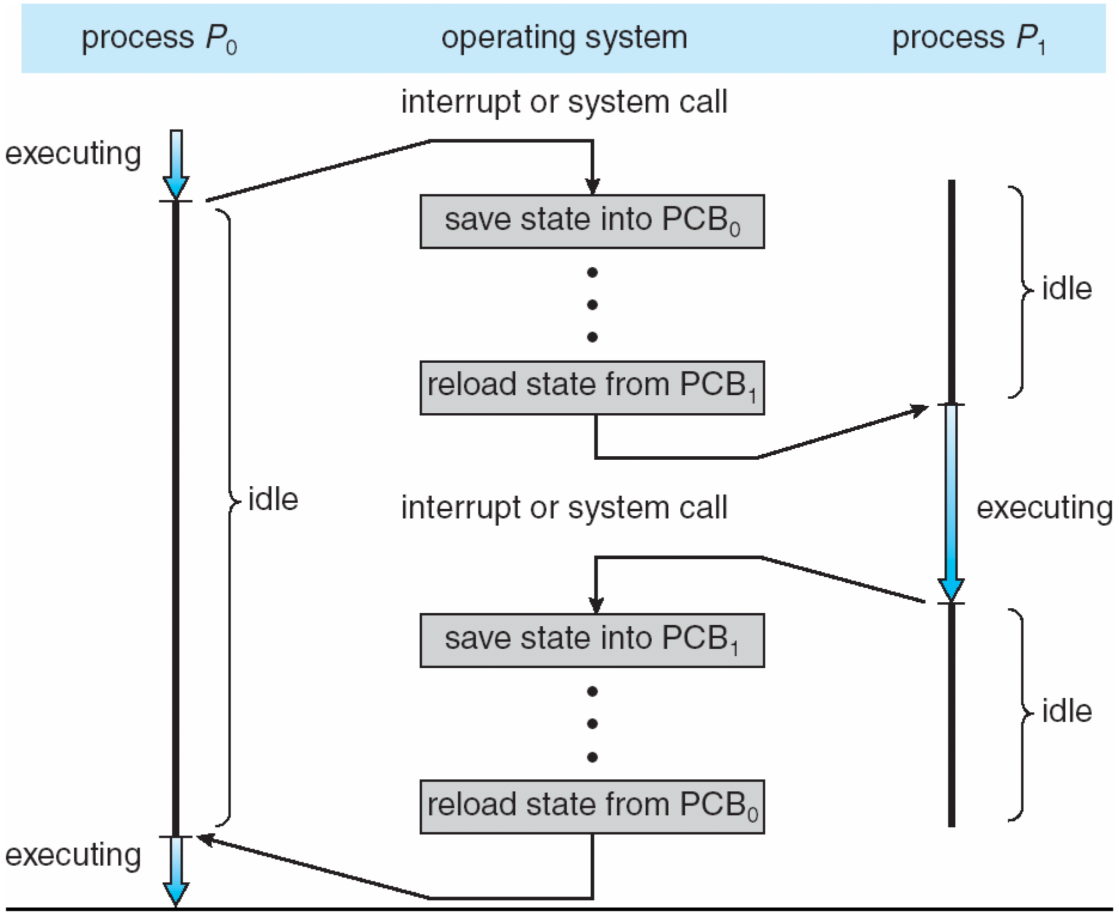
[redacted]

CB for P1

Process Control Block

- There is a Process Control Block for each process, enclosing all the information about the process. It is also known as the task control block. It is a data structure, which contains the following:
- **Process State**: It can be running, waiting, etc.
- **Process ID** and the **parent process ID**.
- CPU registers and Program Counter. **Program Counter** holds the address of the next instruction to be executed for that process.
- **CPU Scheduling** information: Such as priority information and pointers to scheduling queues.
- **Memory Management information**: For example, page tables or segment tables.
- **Accounting information**: The User and kernel CPU time consumed, account numbers, limits, etc.
- **I/O Status information:** Devices allocated, open file tables, etc.

CPU Switch From Process to Process



Process Scheduling

- When there are two or more runnable processes then it is decided by the Operating system which one to run first then it is referred to as Process Scheduling.
- A scheduler is used to make decisions by using some scheduling algorithm.
- Given below are the properties of a **Good Scheduling Algorithm**:
- Response time should be **minimum** for the users.
- The **number of jobs processed per hour should be maximum** i.e Good scheduling algorithm should give maximum throughput.
- The utilization of the CPU should be 100%.
- Each process should get a fair share of the CPU.

Process Scheduling Queues

- **Job queue** – set of all processes in the system
- **Ready queue** – set of all processes residing in main memory, ready and waiting to execute
- **Device queues** – set of processes waiting for an I/O device
- Processes migrate among the various queues

What are Scheduling Queues?

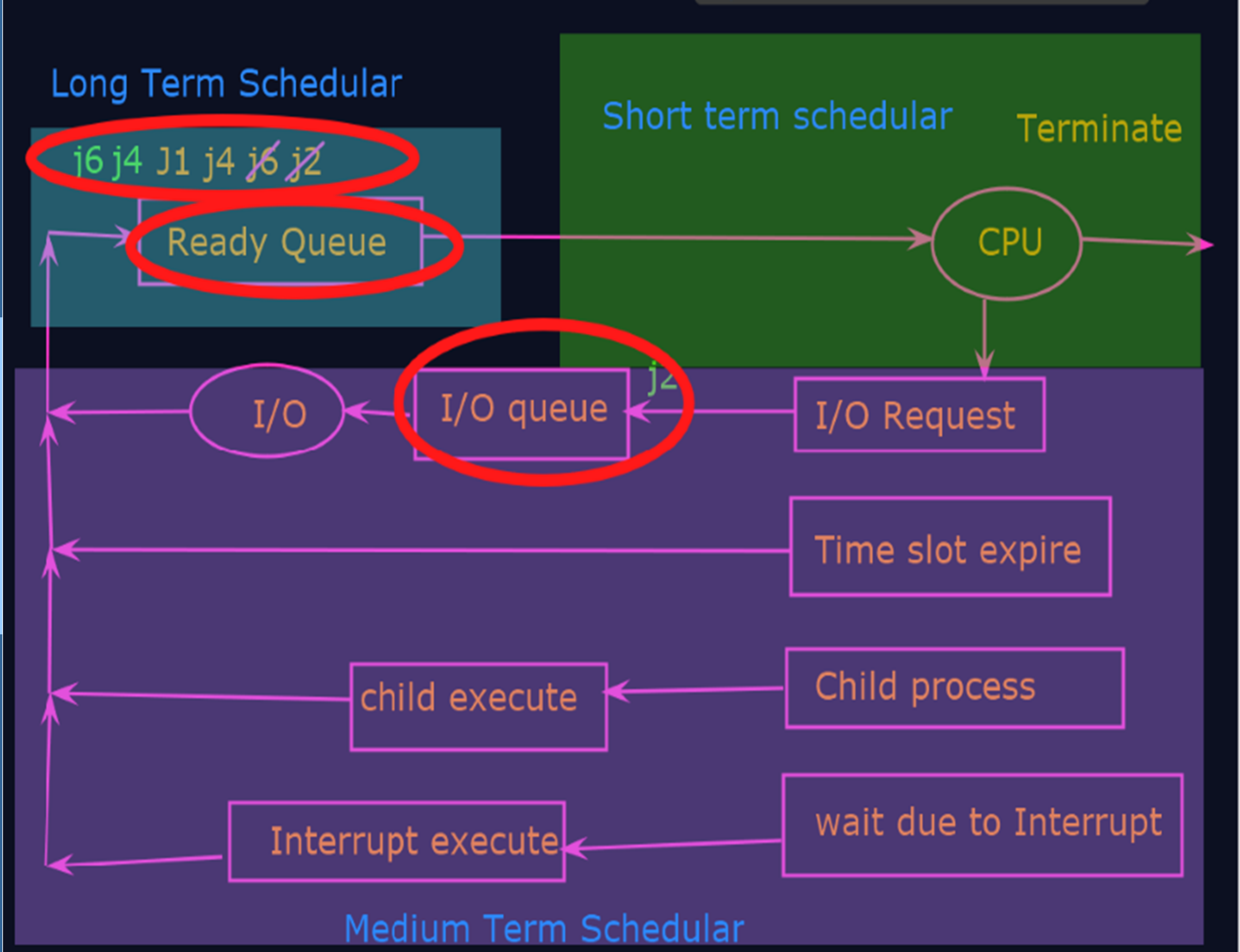
- All processes, upon entering into the system, are stored in the **Job Queue**.
- Processes in the Ready state are placed in the **Ready Queue**.
- Processes waiting for a device to become available are placed in **Device Queues.** There are unique device queues available for each I/O device.
- A new process is initially put in the Ready queue. It waits in the ready queue until it is selected for execution(or dispatched). Once the process is assigned to the CPU and is executing, one of the following several events can occur:

- The process could issue an I/O request, and then be placed in the I/O queue.
- The process could create a new subprocess and wait for its termination.
- The process could be removed forcibly from the CPU, as a result of an interrupt, and be put back in the ready queue.

Types of Schedulers

- There are three types of schedulers available:

- Long Term Scheduler
- Short Term Scheduler
- Medium Term Scheduler


Long Term Schedular
j6 j4 J1 j4 ~~j6~~ ~~j2~~
Ready Queue
Short term schedular
Terminate
CPU
j2
I/O
I/O queue
I/O Request
Time slot expire
child execute
Child process
Interrupt execute
wait due to Interrupt
Medium Term Schedular