

0301304 FUNDAMENTAL OF OPERATIONG SYSTEM

UNIT	MODULES	WEIGHTAGE
1	INTRODUCTION TO OPERATING SYSTEM	20 %
2	PROCESS MANAGEMENT	20 %
3	PROCESS COMMUNICATION AND SYNCHRONIZATION	20 %
4	MEMORY MANAGEMENT	20 %
5	FILE MANAGEMENT , DISK MANAGEMENT , SECURITY AND PROTECTION	20 %

UNIT -2 Process Management

- Fundamentals of Process Management
- Relationship between Processes
- Life Cycle of Process
- Process Control Block
- Schedulers
 - Long Term Scheduler
 - Short Term Scheduler
 - Medium Term Scheduler

UNIT -2 Process Management

- Scheduling Algorithms
 - First Come First Serve (FCFS)
 - Priority Scheduling
 - SJN
 - Round Robin
 - SRT

UNIT -2 Fundamental of Process Management

- Program
- Job
- Task
- Process

Program = Job = Task = Process

UNIT -2 Program, Job & Task

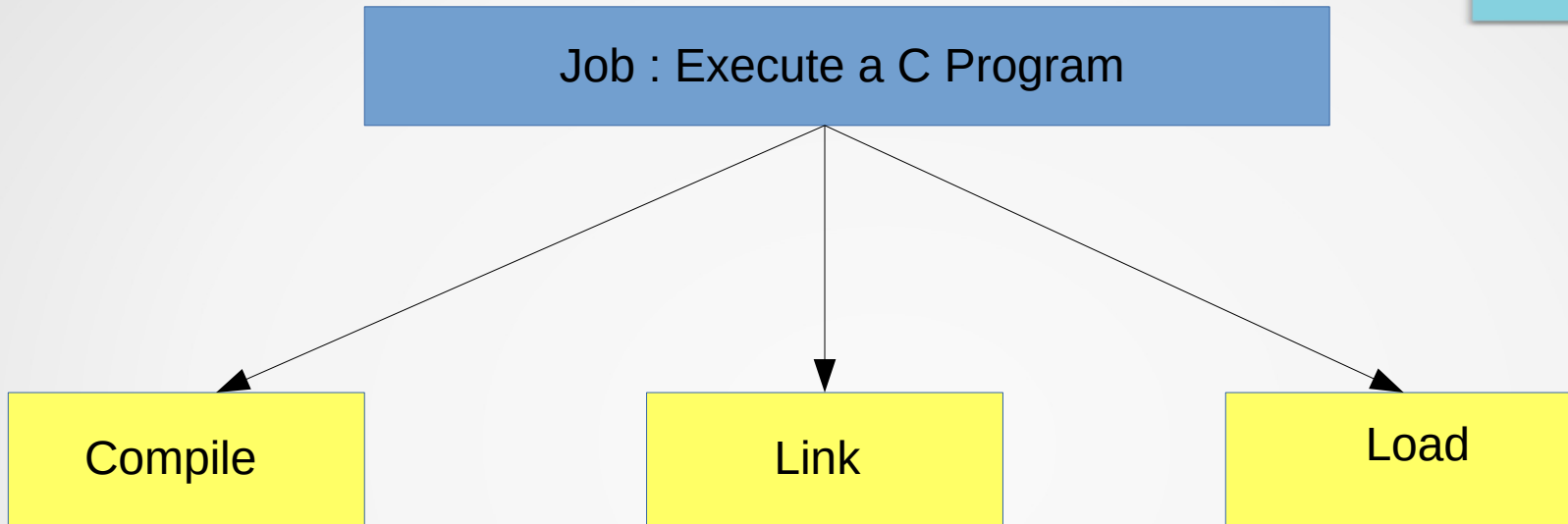
- To perform a computation on the computer system, we must have a unit of work or execution for the user computation.
- **Program**
 - A program can be **considered a set of instructions** in the form of modules.
 - For examp – C program

UNIT -2 Program, Job & Task

- **Job**

- In the Batch system, there was a requirement to load and unload the magnetic tapes for various activities such as
 - **Compiling**
 - **Linking**
 - **Loading**
- Job was used for performing the execution of a program through the execution of those activities.
- **Job is a sequence of single programs**

UNIT -2 Program, Job & Task



Job as a sequence of programs

UNIT -2 Program, Job & Task

- **Task**

- The term task was used **when there was a need to have concurrent execution on a single processor.**
- More than one program of a single user is called Task.
- In window environment opening multiple window.

UNIT -2 Program, Job & Task

- **Process**

- The term '**Process**' is different from the term '**Job**' or '**Program**'.
- A **Program** is a set of instructions the user has written and stored somewhere.
- It means a program is **passive entity** and continues to exist at a place.
- **When a program is ready for the execution, it becomes active and is known as a process.**

UNIT -2 Program, Job & Task

- **Process**

- When a program is ready for execution or in other words, when it becomes a process, **it means that now it can compete for resources.**
- The resources are :
 - CPU time
 - Memory
 - I/O devices, etc.

UNIT -2 Fundamental of Process Management

- Process is a basic term to understand the operating sytem.
- There are **number of user and system processes**, so their is need to manage them.
- Any given time, **any running process may be interrupted.**
- Due to **this interruption, the processes are not in the same state forever.** They change state according to an event in the system.

UNIT -2 Fundamental of Process Management

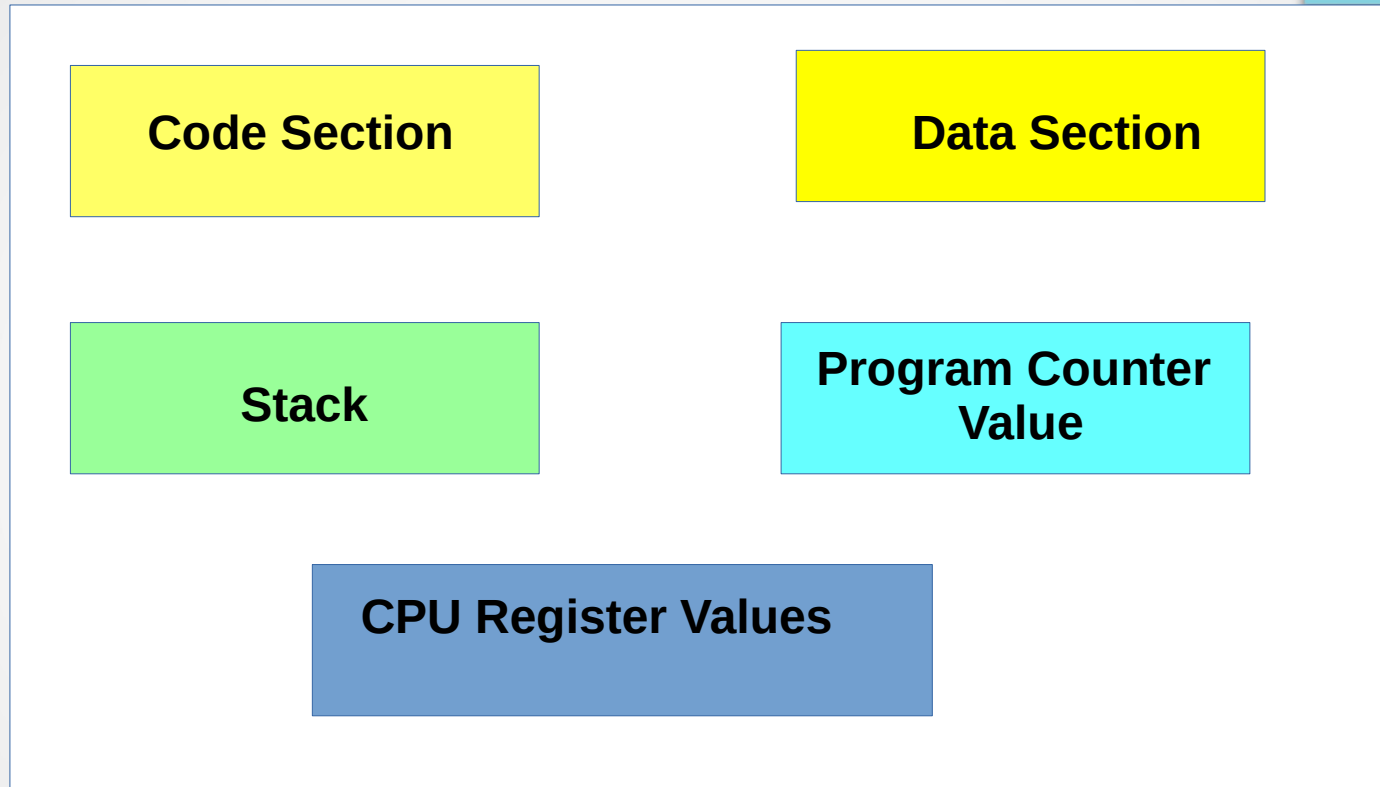
- If a process is interrupted, another process is scheduled to be dispatched to processor for execution.
- Addition to this, processes also need to communicate and synchronize with each other.
- It is critical to manage the processes in the system from the view point of their **state change, scheduling, dispatching, process switching, communication, synchronization and so on.**

UNIT -2 Program, Job & Task

- **Process**

- When a program need to execute and when it get the CPU time, it has :
 - ***Program Counter (PC)*** value – used for moving to the next instruction while executing along with
 - ***Program Code / Code Section***
 - ***Data Section and stack*** also allocated to process along with other resources.
 - At the time of exection process may store data at **CPU Registers.**

UNIT -2 Process Environment



Process Environment

UNIT -2 Program V/s Process

Program	Process
Passive / Static	Active / Dynamic
Can't compete for resources	Competes for resources
Has a code section	Has a code section, data section, stack, and program counter

UNIT -2 Relationship between Processes

- The program may consist of set of procedures or functions, but each of them executes in a sequence.
- This program when executed becomes a single process consisting of all procedures inside the program.
- In some case, some procedures or functions in a program can be executed simultaneously.
- That is, there is no sequence of order of execution between them.
- **In this case, the program when executed consists of many processes, this type of program is known as concurrent program.**

UNIT -2 Relationship Between Processes

- In general, there are some processes (created out of a program or concurrent program) whose execution overlap in time. **This processes are known as concurrent processes.**
- Concurrent processes may be indepedent of each other and are known as **independent processes.**
- It may happen that, they interact with each other, share data, send a message to each other or send signals to coordinate their activites. **These are known as interacting or cooperating processes.**

UNIT -2 Life Cycle of a Process

- A Process is an active entity and change its state with time.
- A Process from its creation to termination passes through various states.
- A process when created is in a ***new state as a program/job.***
- When new job is entered into the system, it is stored in the ***job pool / queue*** maintained in the hard disk.

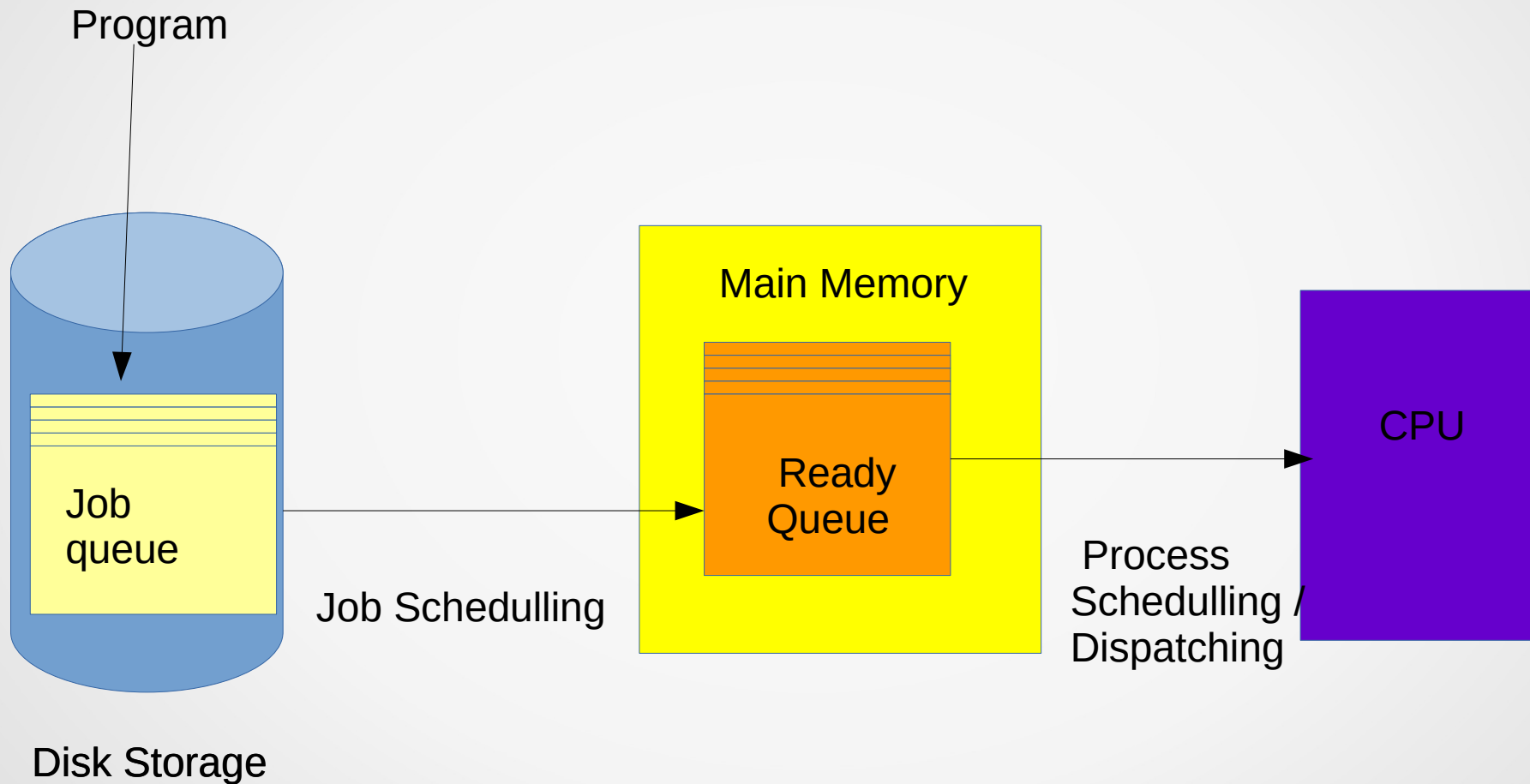
UNIT -2 Life Cycle of a Process

- In the job queue, a job waits for its turn to be moved to the main memory.
- When a job is selected to be brought into the main memory, it is ***called job scheduling***.
- When a job is loaded into the memory, it becomes a ***process*** and is stored in a waiting queue where all processes wait for their turn to be sent to the CPU for execution.
- This waiting queue is called ***ready queue***.
- A Process in ready queue becomes ready for execution as it can compete for the CPU and resources.

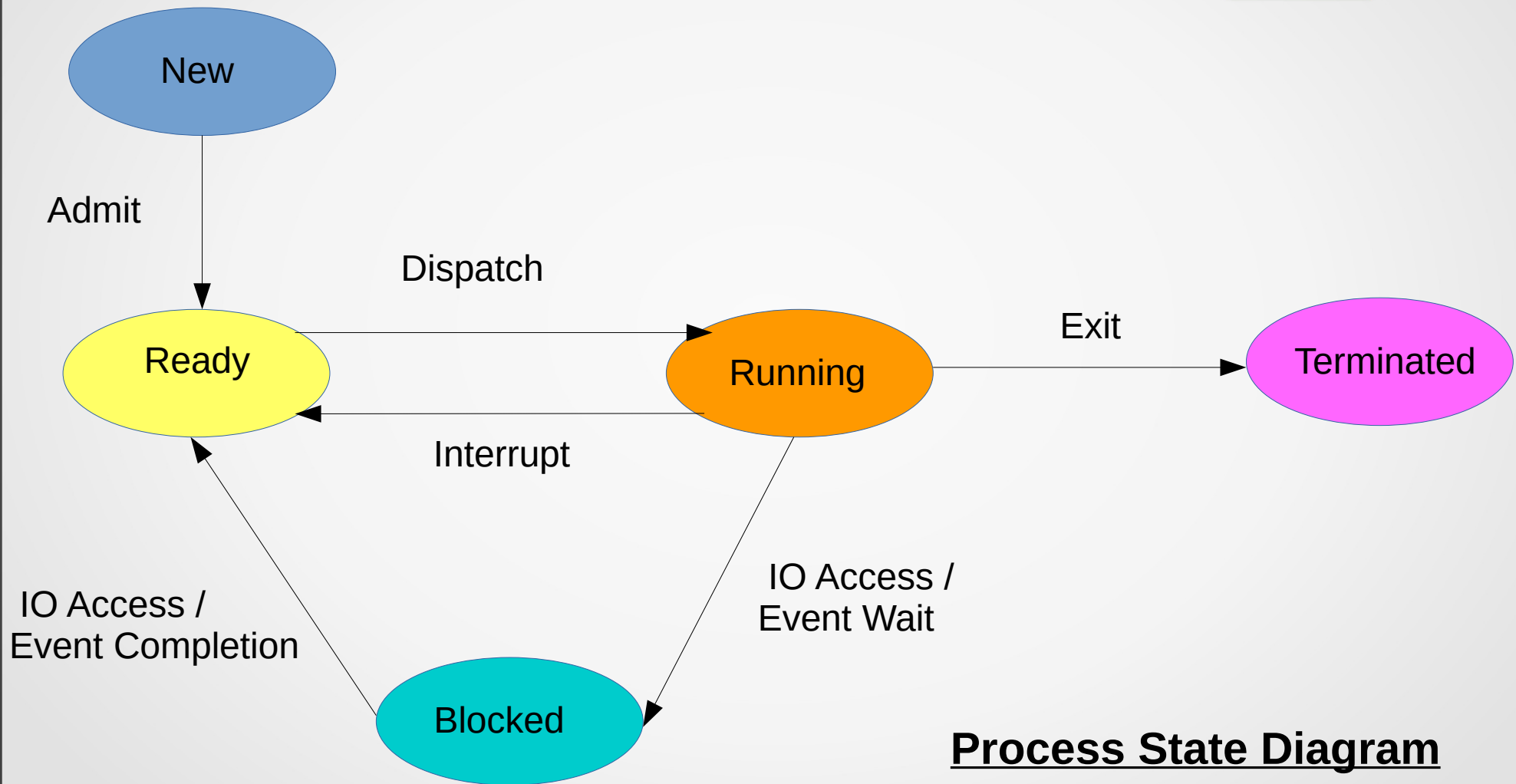
UNIT -2 Life Cycle of a Process

- The processes in ready queue are then selected for the next execution called ***process scheduling or CPU scheduling***.
- The selected process is sent for execution called ***process dispatching***.
- After getting the CPU time, the running process executes its full code and terminates.
- This scheduling and dispatching functions are performed by the ***scheduler*** and ***dispatcher***.

Job Scheduling & Process Scheduling



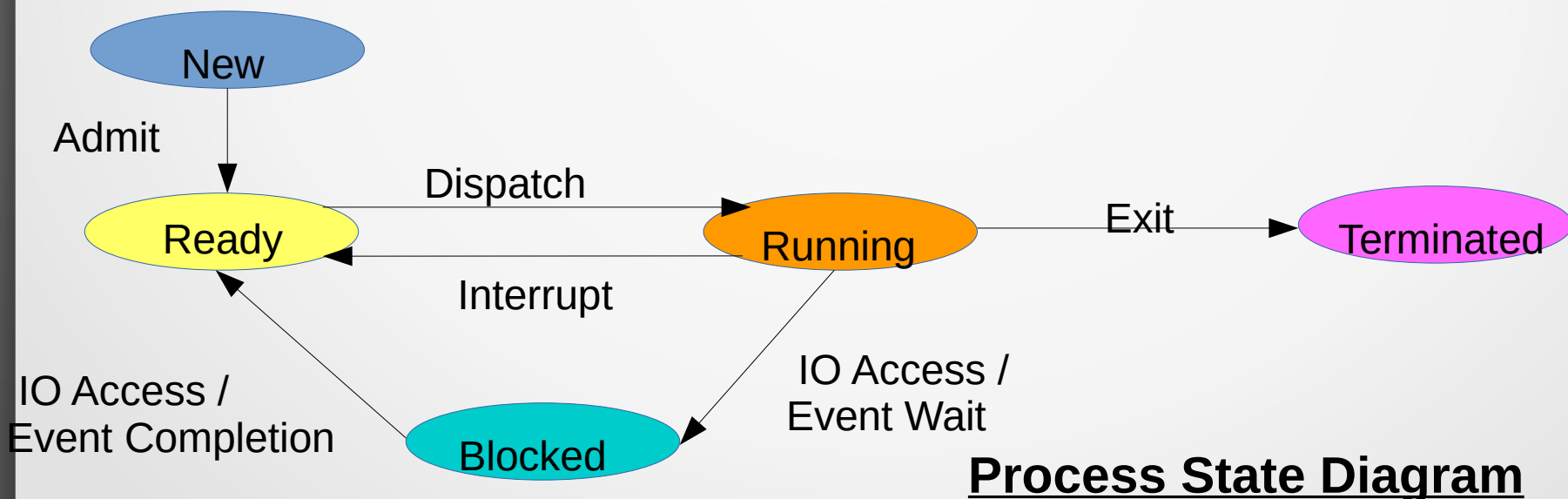
UNIT -2 Process States and State Transitions



UNIT -2 Process States and State Transitions

- **New State**

- Whenever a job/program enters the system, it is put into a **job queue**, the process is in its ***New state***.
- It means that the process is still **in the secondary storage** as a program.

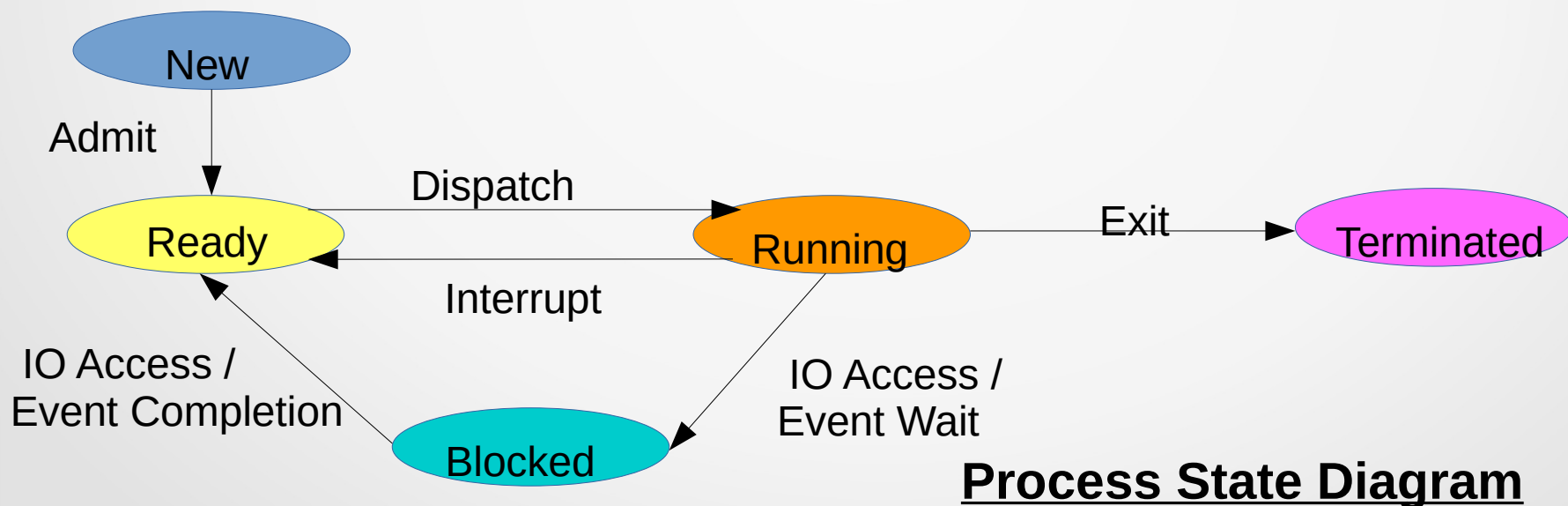


Process State Diagram

UNIT -2 Process States and State Transitions

- **Ready State**

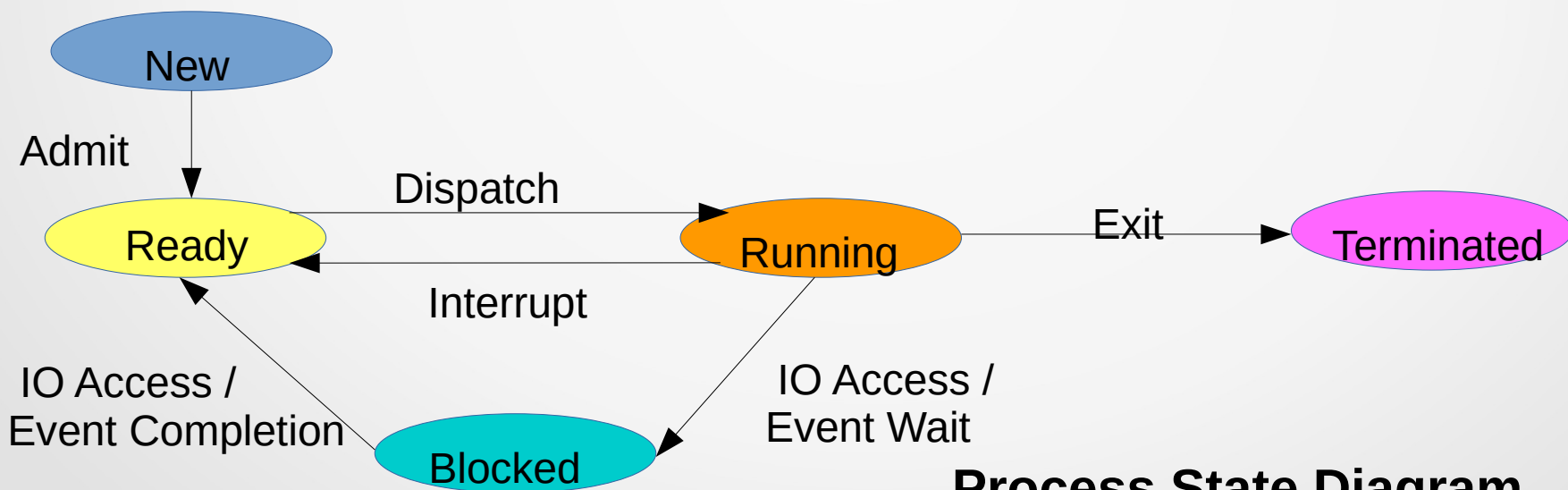
- When program in job queue is scheduled and brought to the main memory in **the ready queue**, the state of the process is **ready**.
- The process in ready state is called ready because now it is ready for execution but not executing.



UNIT -2 Process States and State Transitions

- **Running State**

- A process in the ready queue when **selected by the scheduling mechanism for execution and dispatched to the CPU becomes a running process.**
- The CPU executes the instruction in the code of process.
- A process while running does not mean that it will hold the CPU until it terminates.



Process State Diagram

UNIT -2 Process States and State Transitions

- **Blocked State**

- A process while executing may reach an instruction where it has to wait for **some I/O devices or some other event**.
- In this case, the processor will be taken away from the running process and may be given to another ready process.
- Therefore, **the current running process becomes a blocked process**.
- The blocked process waits for the event only in the main memory but in a separate queue known as **blocked queue**.

UNIT -2 Process States and State Transitions

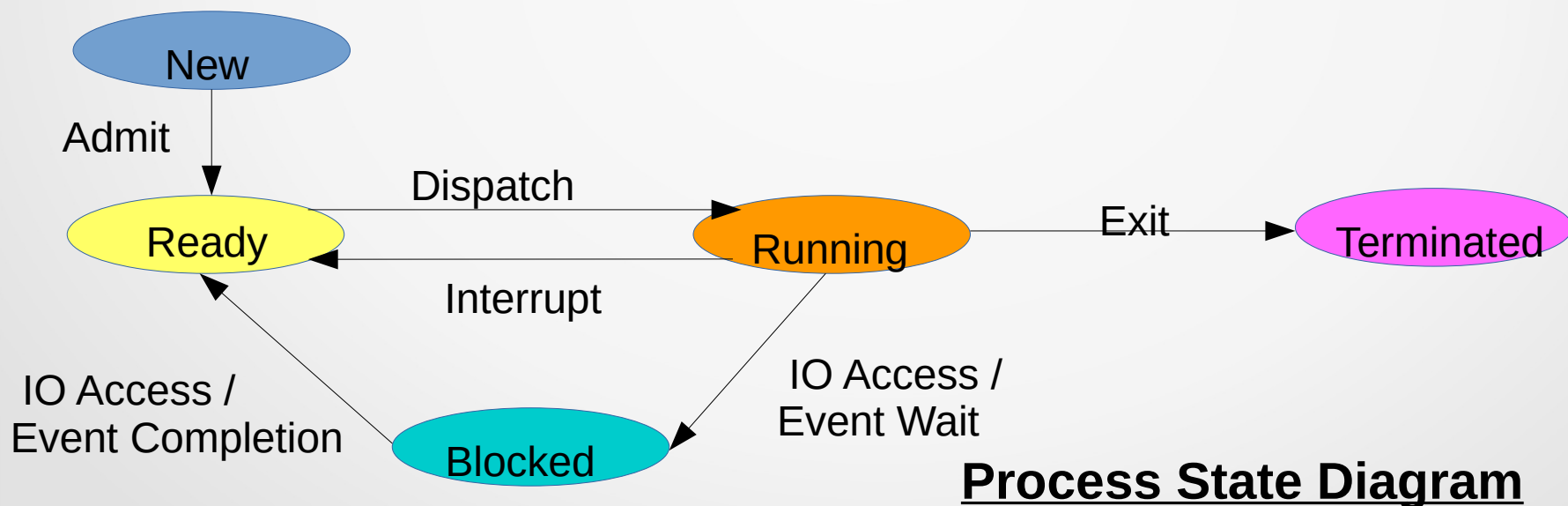
- **Blocked State**

- When the **I/O access or the other event is over, the process is now again ready to execute further.**
- However now it cannot be given the processor because some other process may be executing at that time.
- Therefore, **the blocked process after its wait will move again to the ready queue for its turn to execute.**

UNIT -2 Process States and State Transitions

- **Terminated State**

- A process executed completely till its end and terminates and becomes a ***terminated process***.



Process State Diagram

UNIT -2 Process States and State Transitions

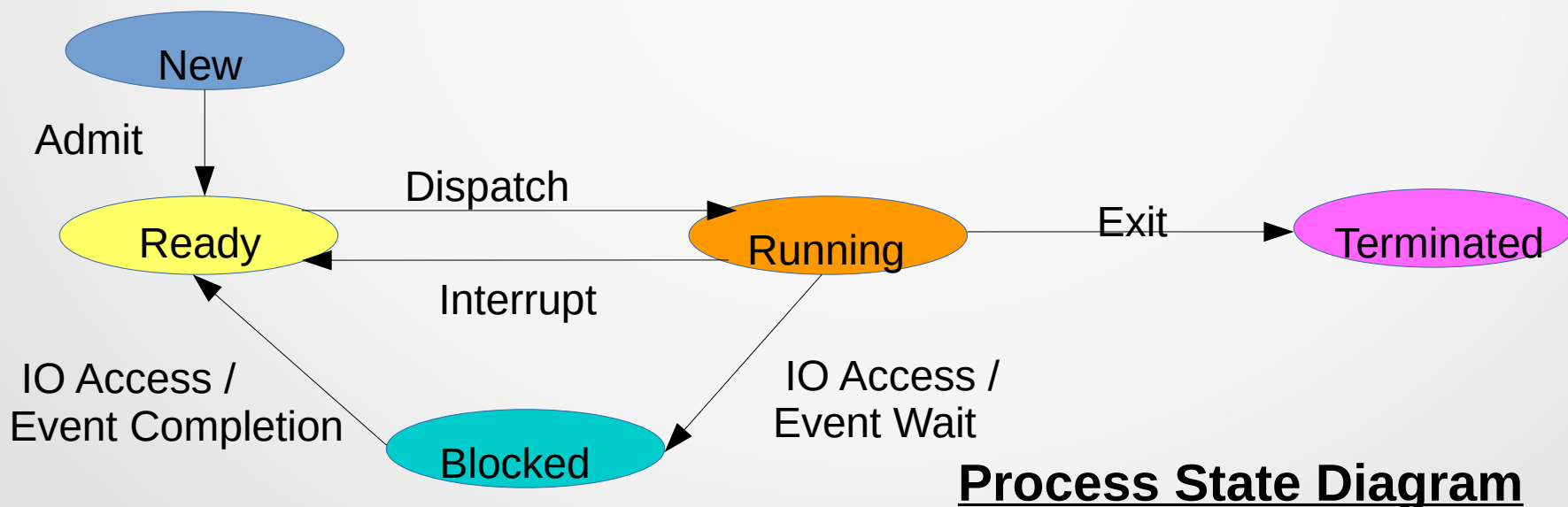
- The **process changes its state when there is an event causing a state transition**. The events can be of the following types:
 - A new process is created.
 - The process makes a resource request.
 - Resource is released.
 - The process requests an I/O device.
 - An I/O device is released after access
 - The allocated time slice for a process is over. In this case, system timer sends a timer interrupt.

UNIT -2 Process States and State Transitions

- The **process changes its state when there is an event causing a state transition**. The events can be of the following types:
 - A higher – priority process appears in the ready queue.
 - The process reaches its end of execution or is aborted.
 - Any hardware interrupt generated.
 - An error or exception condition is generated in the current running process.

UNIT -2 Process States and State Transitions

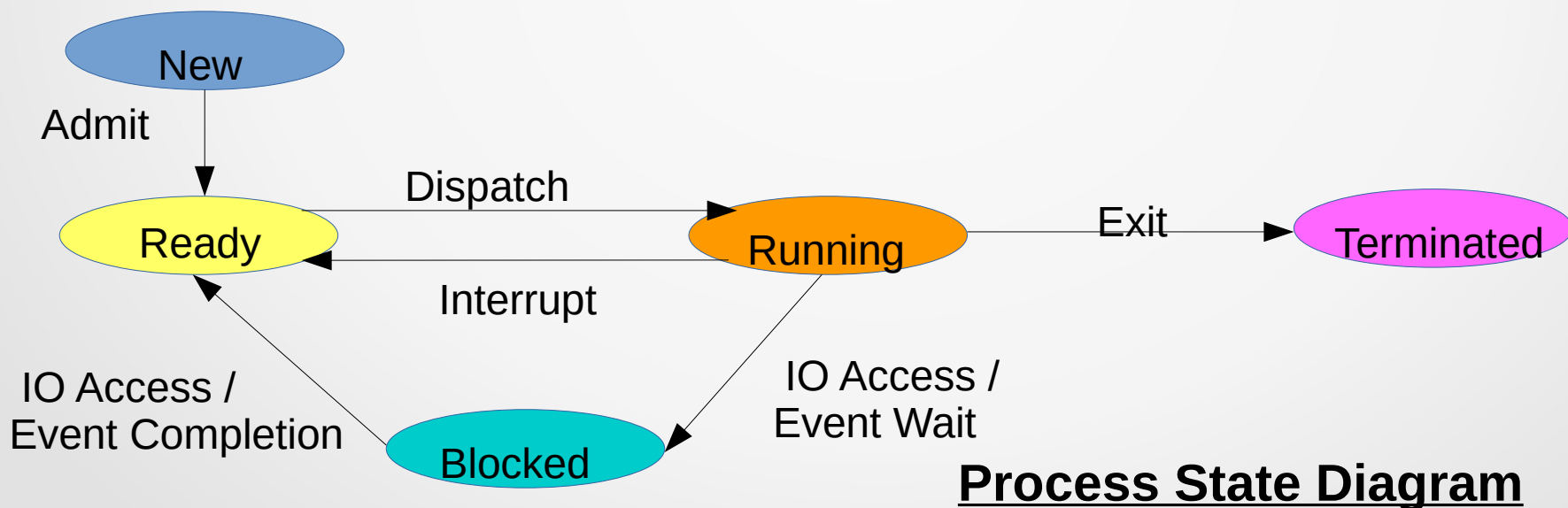
- **Event become the source of state transitions** of the processes and help in managing the processes.
 - **Admit** (New -> Ready)
 - **Dispatch** (Ready -> Running)
 - **Exit** (Running -> terminated)



Process State Diagram

UNIT -2 Process States and State Transitions

- **Event become the source of state transitions** of the processes and help in managing the processes.
 - **Interrupt** (Running - > Ready)
 - **I/O or Event Wait** (Running -> Blocked)
 - **I/O or Event Wait Completion** (Blocked -> Ready)



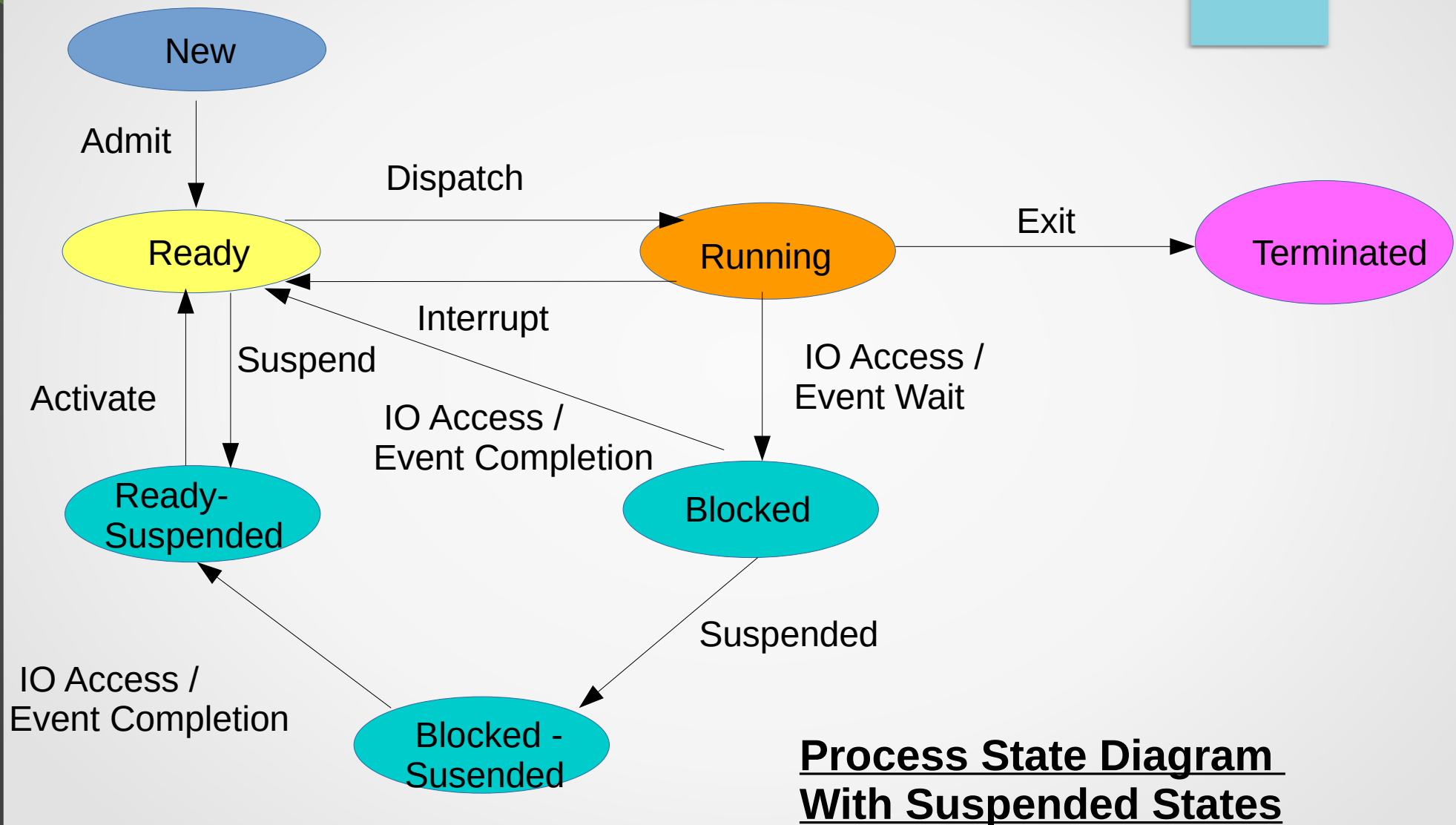
UNIT -2 Suspended Processes & Their State Transitions

- Any process **change their states** when there is an **event**.
- When a process waits for an I/O device, that is, when the process is blocked, another process from the ready queue is scheduled.
- However, there may be the case that **all the processes need I/O devices, that is all the processes at a particular instant of time are blocked** and are waiting for some event to happen and no process is under execution.
- In this case, **no useful work is being done by the processor**.

UNIT -2 Suspended Processes & Their State Transitions

- Therefore, it is **necessary to bring in some process that is ready for execution.**
- However, **there may be a situation that there is no space so that a new process may be swapped in.**
- Therefor, we need to create the memory space for this purpose.
- Since blocked processes cannot be executed unless their I/O devices are released, **some blocked process may be swapped out.**
- The swapped out process is known as **suspended process** and the **queue where it waits is called suspended queue** in the secondary storage such as **hard disk.**

UNIT -2 Process States and State Transitions



UNIT -2 Suspended Processes & Their State Transitions

- **Blocked Suspended State**

- The **blocked process** waiting in blocked queue in the memory is **suspended** and moved to **suspended queue** in disk.
- The state of the process is called **blocked suspended**.

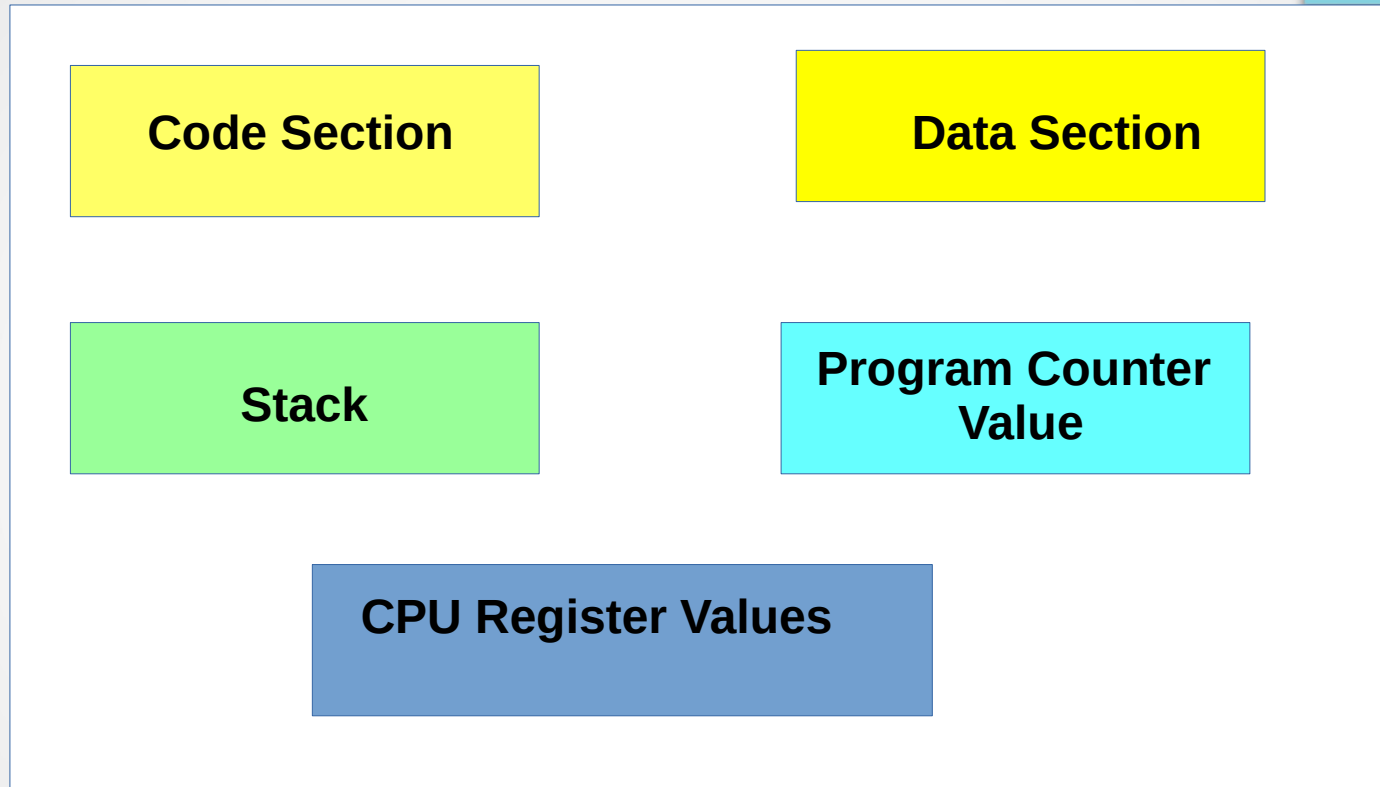
- **Ready Suspended State**

- When the event for which the **blocked suspended process was waiting has occurred**, its state changes.
- The state is **ready** because now it is **ready to be executed**.
- However yet it cannot be executed as it is still in the disk.
- Therefore, its state is **called ready Suspended**.

UNIT -2 Suspended Processes & Their State Transitions

- **Suspend** (Blocked -> Blocked Suspend)
- **I/O or Event Wait Completion** (Blocked Suspend -> Ready suspend)
- **Activate** (Ready Suspend -> Ready)
- **Suspended** (Ready -> Ready Suspend)

UNIT -2 Process Environment



Process Environment

UNIT -2 Process Control Block (PCB)

- Process environment consists of program code, data section and stack.
- But **these are not sufficient to control a process.**
- The **OS needs some attributes associated with the processes to implement and control them.**
- The attributes are stored in the data structure known as **Process Control Block (PCB).**

UNIT -2 Process Control Block (PCB)

- The **PCB is created for every process** whenever a new process is created in the system.
- It also **deleted as the process is terminated**.
- The **PCB contains all the information about the process needed for its control**.
- For e.g – for the identification of process, it must have PID.
 - Current state of Process and its PC value
 - Process priority
 - The information related to resources held by the process and accounting information such as CPU time used.

UNIT -2 Process Control Block (PCB)

PID
PC & CPU registers
Process State
Process Priority
Event Information
Memory – Related Inforamtion
Resouce – related Information
Scheduling – Related Information
Various Pointers

UNIT -2 Process Control Block (PCB)

- **PID**
 - It is a **unique identification** number of the process
- **PC**
 - Indicates the **address value at which the next instruction of the process will be executed** by the processor.
- **Registers**
 - CPU registers are used for the execution of a process. While the process is in execution, **data registers, address registers, control and status registers** are used for executing and controlling the process.
 - The **registers information must be saved when there is a state change of the process** so that it may resume its execution when its next turn comes.

UNIT -2 Process Control Block (PCB)

- **State**

- A process has a number of states in its life. For scheduling the processes, the current state of a process must be known.

- **Priority**

- The **priority number can be assigned to a process** to give preference to it over other.

- **Event Information**

- This is the **event for which a blocked process is waiting.** If the awaited event is over, the information regarding this event must be stored in this field so that the status of the blocked process is changed to ready.

UNIT -2 Process Control Block (PCB)

- **Memory – Related Information**
 - Memory management component of the **OS uses many registers and tables. The information regarding all this memory related information linked to a process** is also mentioned in the PCB.
- **Resource Related Information**
 - **The resources allocated to a process are listed here.** For e.g all files opened by this process are listed in this filed.

UNIT -2 Process Control Block (PCB)

- **Scheduling – Related Information**
 - A process will be executed according to a scheduling algorithm. **The scheduling related information of a process is also stored such as the time the process has waited, the amount of time the process executed the last time it was running.**
- **Pointer to Parent Process**
 - If a process is a child process, then the pointer to its parent process is stored here.

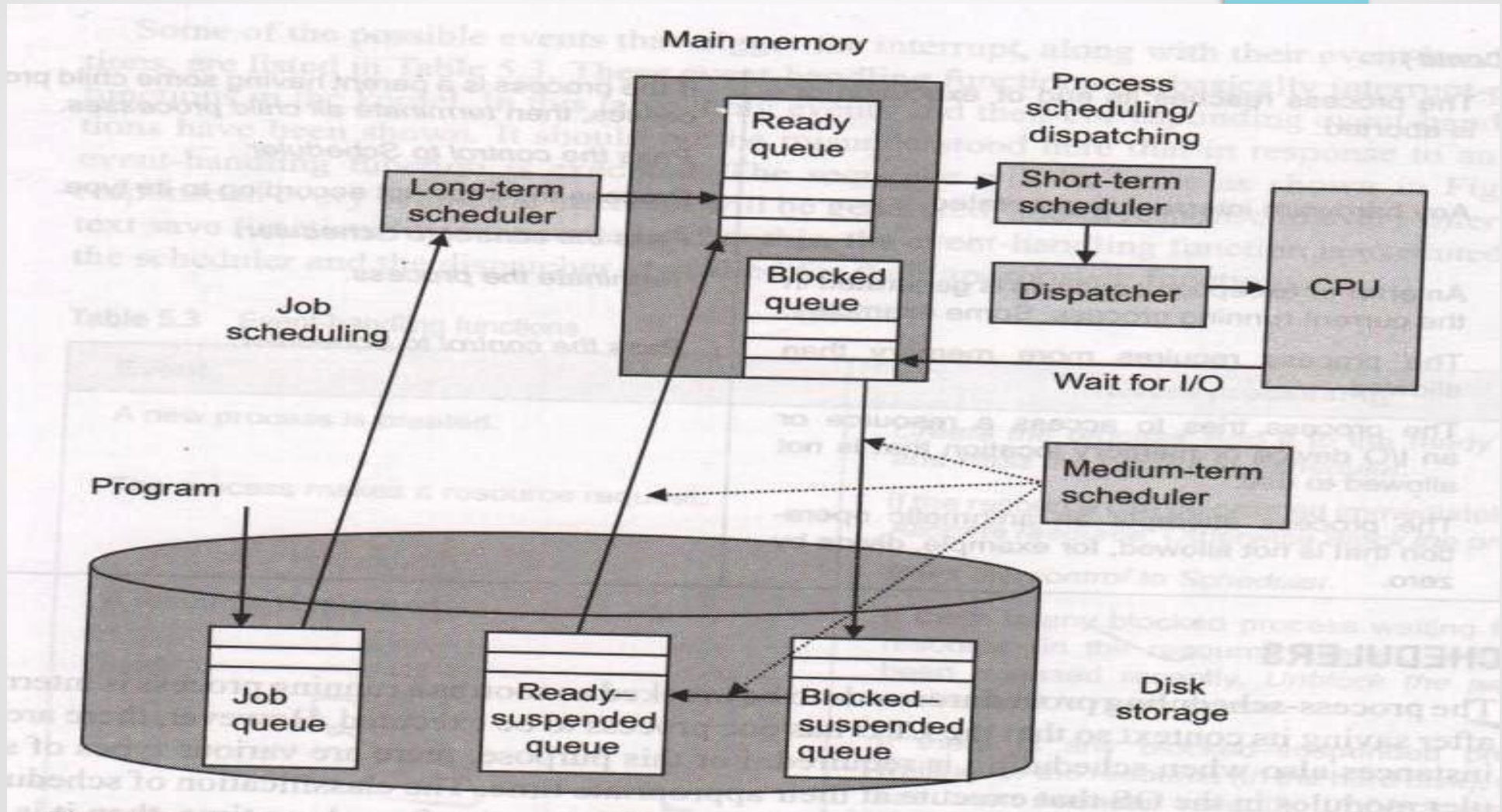
UNIT -2 Process Control Block (PCB)

- **Pointer to Child Process**
 - If a process has some child processes, then the pointer to its child processes is stored here.
- **Pointer to Address Space of the Process**
 - This is the pointer to the process data and instructions in the memory.

UNIT -2 Schedulers

- **The process scheduling procedure needs to be invoked as soon as a process changes its state.**
- The classification of schedulers is based on the frequency of their use in the system.
 - **Long – Term Schedulers**
 - **Short – Term Schedulers**
 - **Medium – Term Schedulers**

UNIT -2 Schedulers



UNIT -2 Long – Term Schedulers

- This scheduler is invoked when there is a need to perform job scheduling, that is, when a job from the job pool is selected to be sent to the ready queue.
- Since a job entering the system needs to be processed in the ready queue, this scheduler is invoked whenever there is a need to increase the degree of multiprogramming in the system.

UNIT -2 Short – Term Schedulers

- This scheduler is invoked when there is a need to perform process scheduling, **that is when a process from the ready queue is to be selected for dispatching to the CPU.**
- There are various instances in the system when this type of scheduling is needed.
- Whenever **there is an interrupt, the running process stops, and the short term scheduler is invoked every time to select another process for execution.**
- **That is why this scheduler is called a short term scheduler.**

UNIT -2 Medium – Term Schedulers

- This scheduler is **invoked when there is a need to swap out some blocked processes.**
- It can happen in the case when all processes are blocked for some I/O devices and there is no ready process to execute.
- Moreover, there is no space for any other process. In this case some blocked processes need to be swapped out from the main memory to the hard disk.
- There is another queue called blocked suspended queue for this purpose in the disk.
- The task for swapping the processes from the blocked queue to the blocked suspended queue is performed by another type of scheduler known as medium term scheduler.

UNIT -2 Scheduling Algorithms

- A scheduling algorithm **selects a process from the ready queue and the dispatcher sends this process to the CPU** for execution.
- Various Scheduling Algorithms are:
 - First Come First Served (FCFS)
 - Priority Scheduling
 - Shortest Process Next (SPN)
 - Shortest Remaining Time (SRT)
 - Round Robin

UNIT -2 First Come First Served

- There is always a simple approach to maintain a queue, that is, the item that comes first will be served first.
- The arriving process is added onto the tail of the queue and the process at the head of the queue is dispatched to the processor for execution.
- This scheduling policy is non pre-emptive because the process which has arrived first will be executed first for its completion.

UNIT -2 First Come First Served

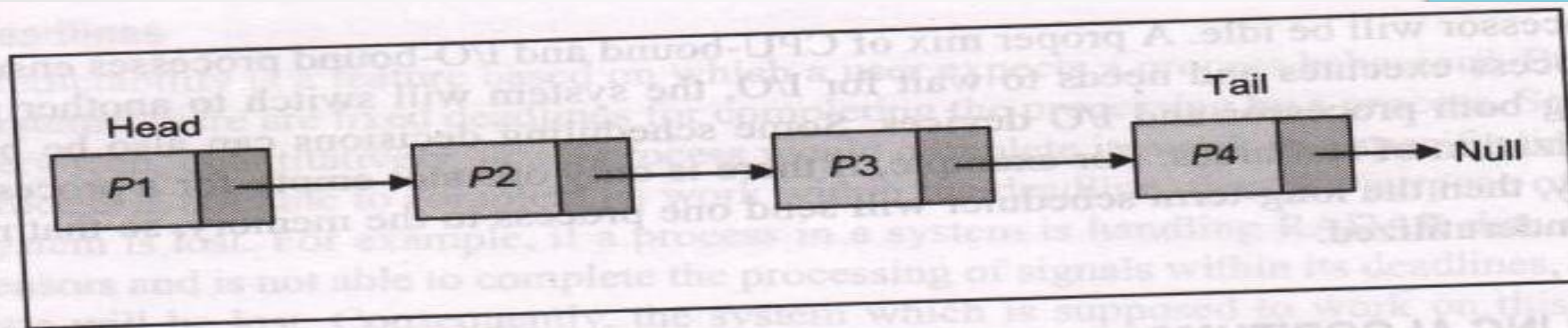


Fig. 6.13 Ready queue of processes

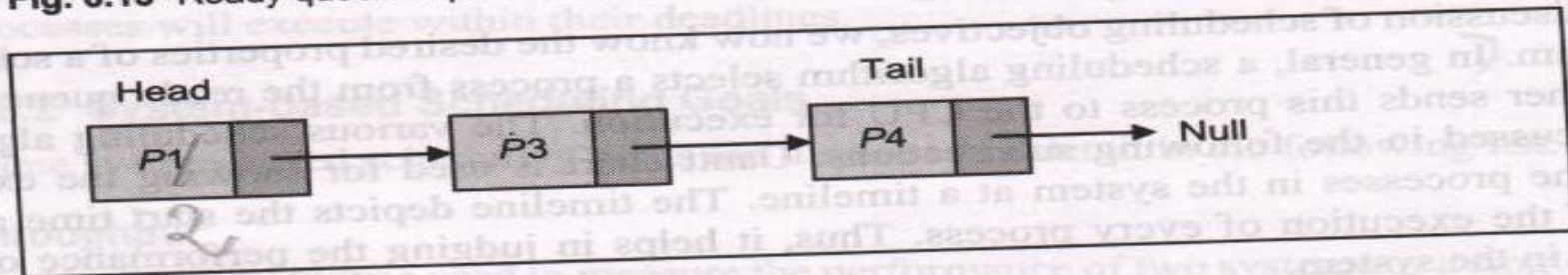
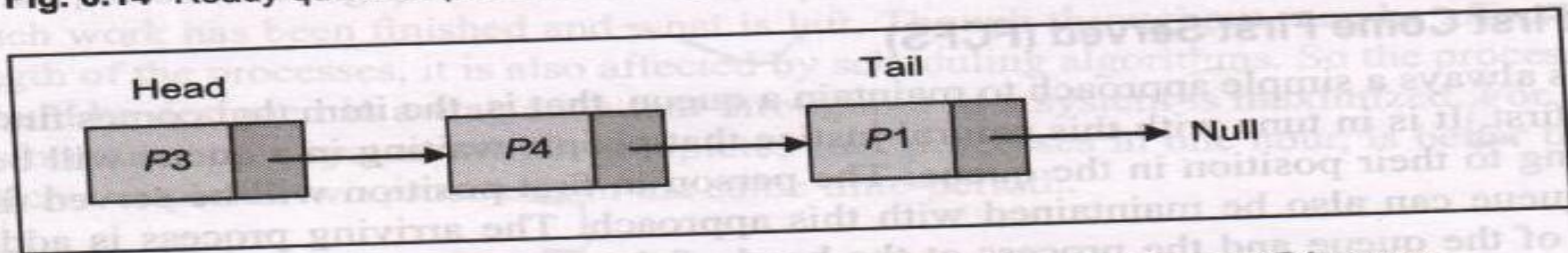


Fig. 6.14 Ready queue of processes: P1 gets execution



UNIT -2 First Come First Served - Example

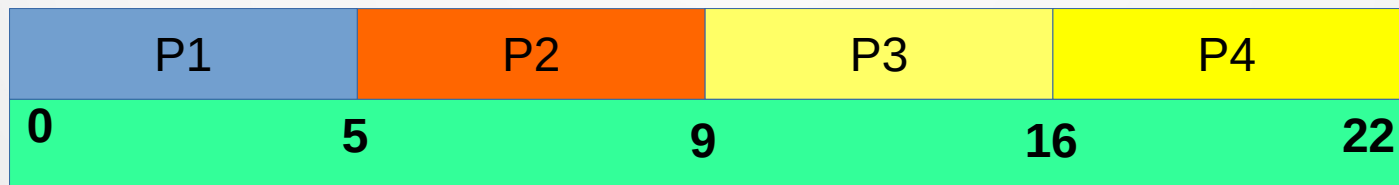
Process	Arrival Time	Execution Time
P1	0	5
P2	2	4
P3	3	7
P4	5	6

- Draw a Gant Chart.
- Show start time, end time using FCFS algorithm.
- Calculate Waiting Time, Turn Around time, Normalized Turnaround Time for each process.
- Average Waiting Time, Turn Around time, Normalized Turnaround Time for the system.

UNIT -2 First Come First Served - Example

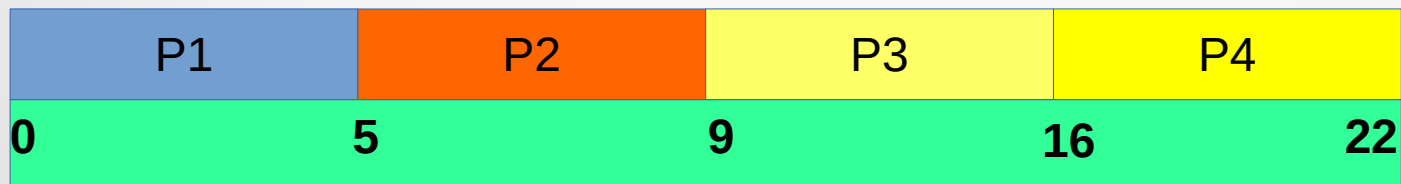
Process	Arrival Time	Execution Time
P1	0	5
P2	2	4
P3	3	7
P4	5	6

- Draw a Gant Chart.



UNIT -2 First Come First Served - Example

Process	Arrival Time	Execution Time (x)	Waiting time = completion time – arrival time – execution time	Turnaround time (tr) (Waiting time + execution time)	Normalized turnaround time (tr / x)
P1	0	5	0	5	1
P2	2	4	3	7	1.75
P3	3	7	6	13	1.85
P4	5	6	11	17	2.84
			Average waiting time = 5	Average turnaround time = 10.5	Average normalized turnaround time = 1.86



UNIT -2 Priority Scheduling

- In some system like real – time systems, each process is well defined with its functionality.
- It has a priority in the system, according to the importance of its functionality.
- If a process with higher priority has arrived late or at its defined time then it will be executed first according to its priority.
- In real time system, the priorities are defined as a number associated with a process. This number scheme is also different in various OS.
- Some system follow lower numbers as higher priorities and other follow higher numbers as higher priorities.
- In general, lower numbers are considered as high priorities.

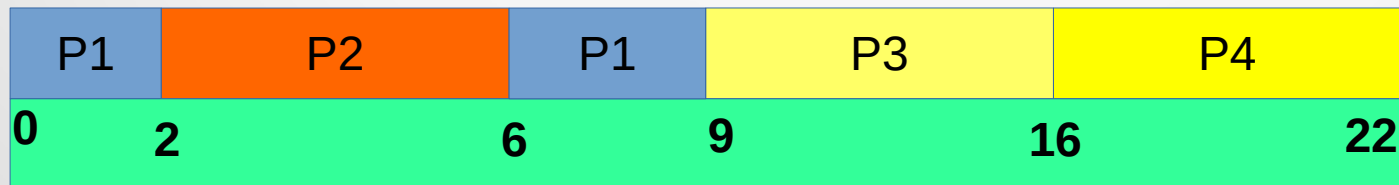
UNIT -2 Priority - Example

Process	Arrival Time	Execution Time	Priority
P1	0	5	2
P2	2	4	1
P3	3	7	3
P4	5	6	4

- Draw a Gant Chart.
- Show start time, end time using Priority algorithm.
- Calculate Waiting Time, Turn Around time, Normalized Turnaround Time for each process.
- Average Waiting Time, Turn Around time, Normalized Turnaround Time for the system.

UNIT -2 Priority - Example

Process	Arrival Time	Execution Time (x)	Priority Number	Turnaround time (tr)	Normalized turnaround time (tr / x)	Waiting time = completion time – arrival time – execution time
P1	0	5	2	9	1.8	4
P2	2	4	1	4	1	0
P3	3	7	3	13	1.85	6
P4	5	6	4	17	2.84	11
				Average turnaround time = 10.75	Average normalized turn around tiem = 1.87	Average waiting time = 5.25



Gant Chart.

UNIT -2 Priority Scheduling- Practice Example

Process	Arrival Time	Execution Time	Priority
P1	0	3	3
P2	2	7	4
P3	3	5	1
P4	5	9	2

UNIT -2 Shortest Job Next (SJN)

- It also called shortest Process Next (SPN)
- Another way to prioritize the processes is to run the processes with shorter execution times.
- In this algorithm, at an instant of time, the processes are compared based on their execution times.
- The process with the shortest execution time is executed first.
- This is a non-preemptive scheduling algorithm.

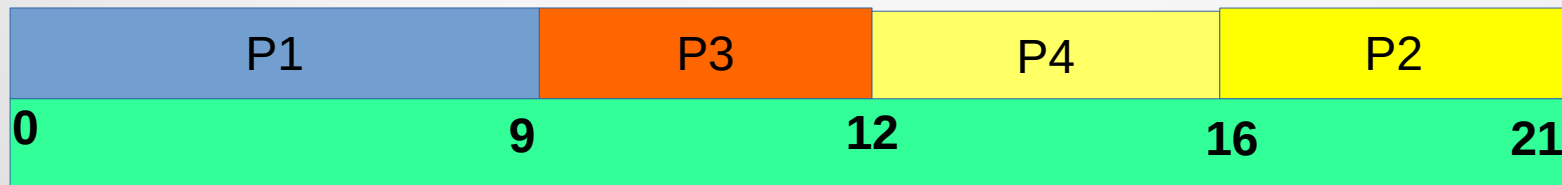
UNIT -2 SJN - Example

Process	Arrival Time	Execution Time
P1	0	9
P2	1	5
P3	2	3
P4	3	4

- Draw a Gant Chart.
- Show start time, end time using Priority algorithm.
- Calculate turnaround time, normalized turnaround time, waiting time for each process.
- Average turnaround time, average normalized turnaround time, average waiting time for the system.

UNIT -2 SJN - Example

Process	Arrival Time	Execution Time (x)	Turnaround time (tr)	Normalized turnaround time (tr / x)	Waiting Time
P1	0	9	9	1	0
P2	1	5	20	4	15
P3	2	3	10	3.34	7
P4	3	4	13	3.25	9
			Average turnaround time = 13	Average normalized turnaround time = 2.89	Average waiting time = 7.75



- Draw a Gant Chart.

UNIT -2 Shortest Remaining Time Next (SRN)

- This algorithm also considers the execution time of processes as in SPN.
- But it is a pre-emptive version of SPN.
- It means, we can pre-empt a process based on the execution time.
- The process with the shortest execution time will always pre-empt other process.
- Since the processes may not be able to complete their execution as they may be pre-empted, the preference for preemption will be based on the remaining execution time of processes.

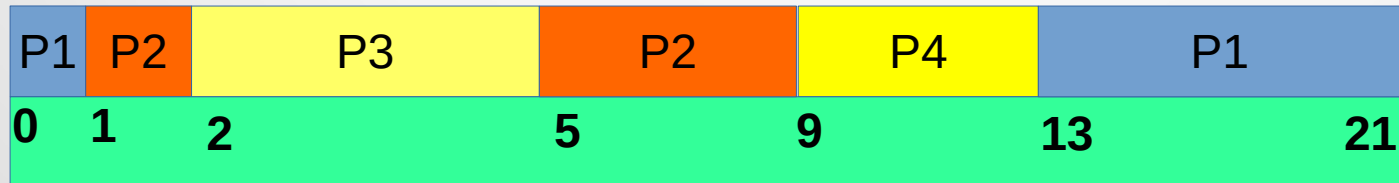
UNIT -2 SRN - Example

Process	Arrival Time	Execution Time
P1	0	9
P2	1	5
P3	2	3
P4	3	4

- Draw a Gant Chart.
- Show start time, end time using Priority algorithm.
- Calculate turnaround time, normalized turnaround time, waiting time for each process.
- Average turnaround time, average normalized turnaround time, average waiting time for the system.

UNIT -2 SRN - Example

Process	Arrival Time	Execution Time (x)	Turnaround time (tr)	Normalized turnaround time (tr / x)	Waiting Time
P1	0	9	21	2.34	12
P2	1	5	8	1.6	3
P3	2	3	3	1	0
P4	3	4	10	2.5	6
			Average turnaround time = 10.5	Average normalized turnaround time = 1.86	Average waiting time = 5.25



- Draw a Gant Chart.

UNIT -2 Round Robin Scheduling

- In multi user time-sharing systems or multi-tasking systems, the requirement of scheduling is different.
- The concept of multi user and multi tasking systems is to share the processor time among processes.
- In this each process gets same processor time, the response will be equally good for all the processes, and neither the short nor long process will suffer from starvation.
- The ready queue can be of the same pattern as that of FCFS, that is, FIFO queue.

UNIT -2 Round Robin Scheduling

- The only issue is that when one process is executing and its fixed allotted time finished, it must be temporarily stopped, and the processor must be given to the next process in the queue.
- In this way, every process gets equal time for execution, and no process can hold the processor for a long time.

UNIT -2 Round Robin Scheduling

- Algorithm:
 1. The ready queue is maintained as a FIFO queue.
 2. A fixed time period is allotted to every arriving process in the queue. This fixed time period is known as time slice or time quantum.
 3. The first arriving process is selected and dispatched to the processor. But if it is not able to complete its execution within its time slice, then an interrupt is generated with the help of the timer.

UNIT -2 Round Robin Scheduling

- Algorithm:

4. As soon as the timer interrupt arrives, the running process is stopped temporarily, and is placed back in the ready queue at the end of the queue. The context of the interrupted process is save, so that it can resume when its turn comes in the future.

5. The scheduler selects another process form the queue and dispatches it to the processor. It is executed until the allotted time slice expires.

6. In this way, scheduling is done in a round robin fashion.

UNIT -2 Round Robin - Example

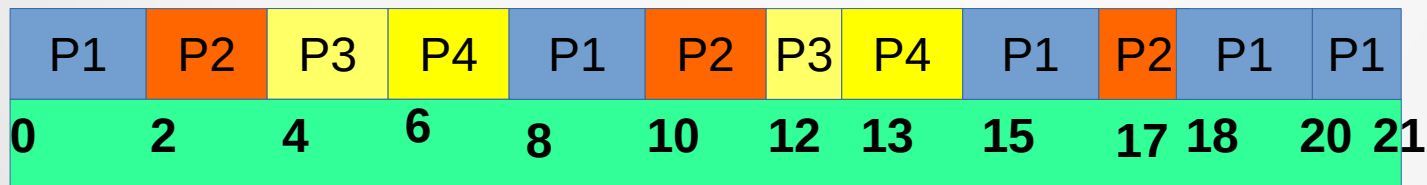
Process	Arrival Time	Execution Time
P1	0	9
P2	1	5
P3	2	3
P4	3	4

- Time quantum = 2
- Draw a Gant Chart.
- Show start time, end time using FCFS algorithm.
- Calculate turnaround time, normalized turnaround time, waiting time for each process.
- Average turnaround time, average normalized turnaround time, average waiting time for the system.

UNIT -2 Round Robin - Example

Process	Arrival Time	Execution Time
P1	0	9
P2	1	5
P3	2	3
P4	3	4

- Draw a Gant Chart.



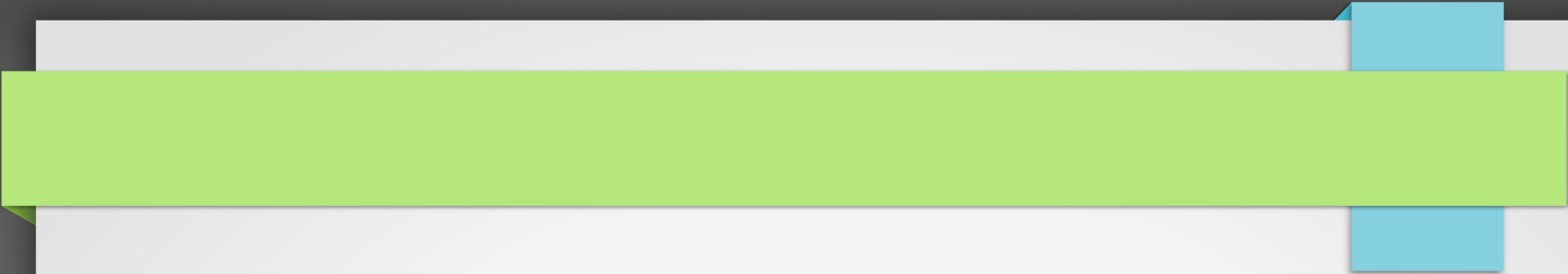
UNIT -2 Round Robin - Example

Process	Arrival Time	Execution Time (x)	Turnaround time (tr) (Waiting time + execution time)	Normalized turnaround time (tr / x)	Waiting time
P1	0	9	21	2.34	12
P2	1	5	17	3.4	12
P3	2	3	11	3.67	8
P4	3	4	12	3	8
			Average turnaround time= 15.25	Average normalized turnaround time = 3.10	Average waiting time = 10





UNIT 2 COMPLETED

- 
- **Assignment Submission – 20/07/2018**
 - **CEC exam for unit 2 on – 20/07/2018**