Printed Pages: 02 **University Roll No……**

Mid-Term Examination, Even Semester 2021-22

B. Tech (CSE, All Specialization), II-Year, III Semester

Operating System (BCSC0004)

Time: 2 Hours **Maximum Marks: 30**

Section- A

Note: Attempt All Three Questions. 3 x 2 = 6 Marks

1. Explain Non-preemptive and Preemptive CPU Scheduling with example. Elaborate all conditions when CPU scheduling is required.

Solution:

CPU scheduling decisions may take place under the following four circumstances:

1. When a process switches from the **running** state to the **waiting** state (for I/O request or invocation of wait for the termination of one of the child processes).
2. When a process switches from the **running** state to the **ready** state (for example, when an interrupt occurs).
3. When a process switches from the **waiting** state to the **ready** state (for example, completion of I/O).
4. When a process **terminates**.

In circumstances 1 and 4, there is no choice in terms of scheduling. A new process (if one exists in the ready queue) must be selected for execution. There is a choice, however in circumstances 2 and 3.

When Scheduling takes place only under circumstances **1** and **4**, we say the scheduling scheme is **non-preemptive**; otherwise the scheduling scheme is **preemptive**.

2. Which CPU scheduling Algorithm causes Starvation? Write starvation and briefly explain its solution?

Starvation or indefinite blocking is phenomenon associated with the Priority scheduling algorithms, in which a process ready to run for CPU can wait indefinitely because of low priority. In heavily loaded computer system, a steady stream of higher-priority processes can prevent a low-priority process from ever getting the CPU.

Solution to Starvation: Aging

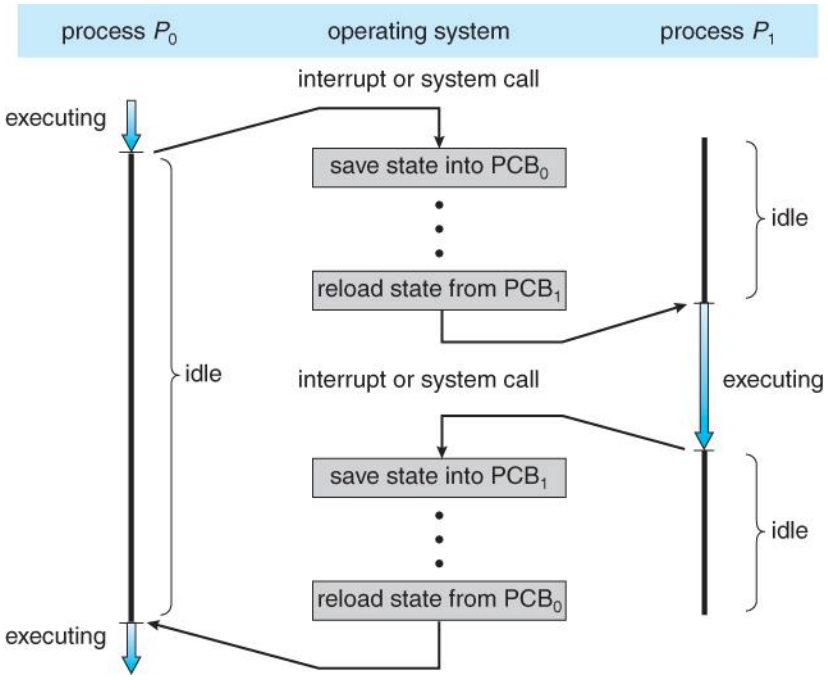
Aging is a technique of gradually increasing the priority of processes that wait in the system for a long time. For example, if priority range from 127(low) to 0(high), we could increase the priority of a waiting process by 1 Every 15 minutes. Eventually even a process with an initial priority of 127 would take no more than 32 hours for priority 127 process to age to a priority-0 process

3. What is Process Control Block? Explain how PCB functions during context switch?

PROCESS CONTROL BLOCK:

Each process is represented in the OS by a PROCESS CONTROL BLOCK (PCB)- also called Task Control Block.

The PCB serves as a repository for any information that may ary from process to process. Because PCBs need to be manipulated quickly by the OS, many computer systems contain a H/W register that always points to the PCB of the currently executing process.



Section- B

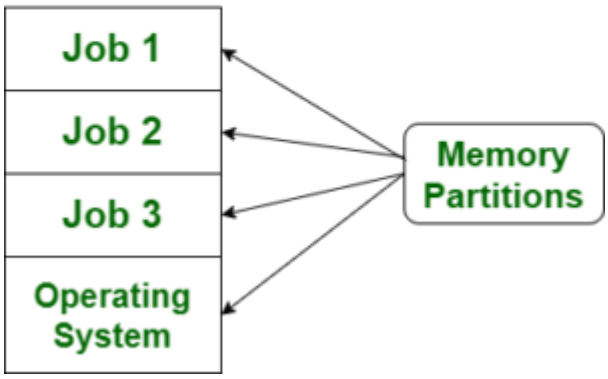
Note: Attempt All Three Questions 3 x 3 = 09 Marks

1. Distinguish Multiprogramming, Multitasking and Multiprocessing systems.

Multi-Programming:-

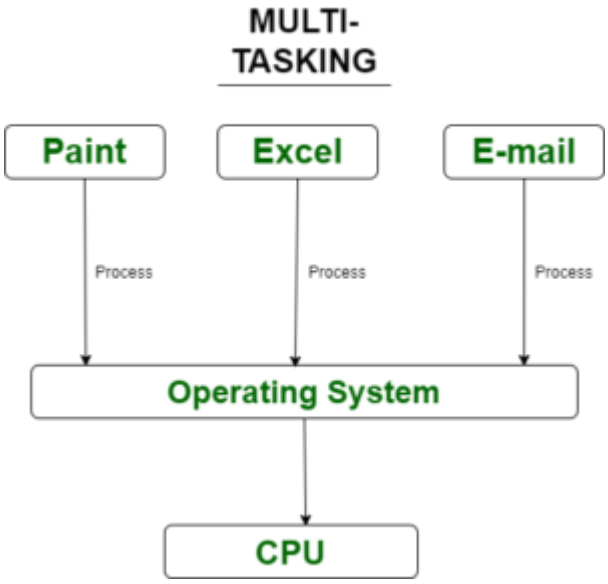
Multi-programming increases CPU utilization by organizing jobs (code and data) so that the CPU always has one to execute. The idea is to keep multiple jobs in main memory. If one job gets occupied with IO, CPU can be assigned to other job.

Multiprogramming



Multi-tasking:-

Multi-tasking is a logical extension of multiprogramming. Multitasking is the ability of an OS to execute more than one **task** simultaneously on *a CPU machine.* These multiple tasks share common resources (like CPU and memory). In multi-tasking systems, the CPU executes multiple jobs by switching among them typically using a small time quantum, and the switches occur so quickly that the users feel like interact with each executing task at the same time.



Multiprocessing –

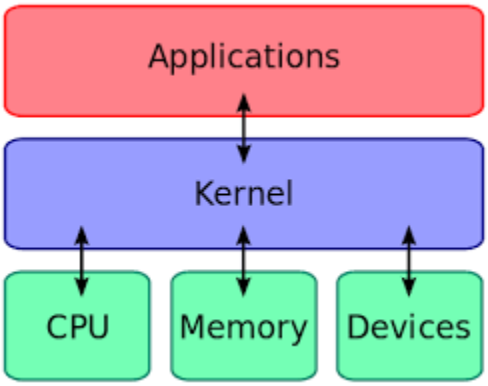
In a uni-processor system, only one process executes at a time. Multiprocessing is the use of two or more CPUs (processors) within a single Computer system. The term also refers to the ability of a system to support more than one processor within a single computer system. Now since there are multiple

processors available, multiple processes can be executed at a time. These multi processors share the computer bus, sometimes the clock, memory and peripheral devices also.

Sr.no	**Multiprogramming**	**Multi-tasking**
1.	These concepts are for single CPU.	These concepts are for single CPU.
2.	Concept of Context Switching is used.	Concept of Context Switching and Time Sharing is used.
3.	In multiprogrammed system, the operating system simply switches to, and executes, another job when current job needs to wait.	The processor is typically used in time sharing mode. Switching happens when either allowed time expires or where there other reason for current process needs to wait (example process needs to do IO).
4.	Multi-programming increases CPU utilization by organising jobs .	In multi-tasking also increases CPU utilization, it also increases responsiveness.
5.	The idea is to reduce the CPU idle time for as long as possible.	The idea is to further extend the CPU Utilization concept by increasing responsiveness Time Sharing.

2. What is the role of kernel in OS? Distinguish clearly the difference between microkernel and monolithic kernel.

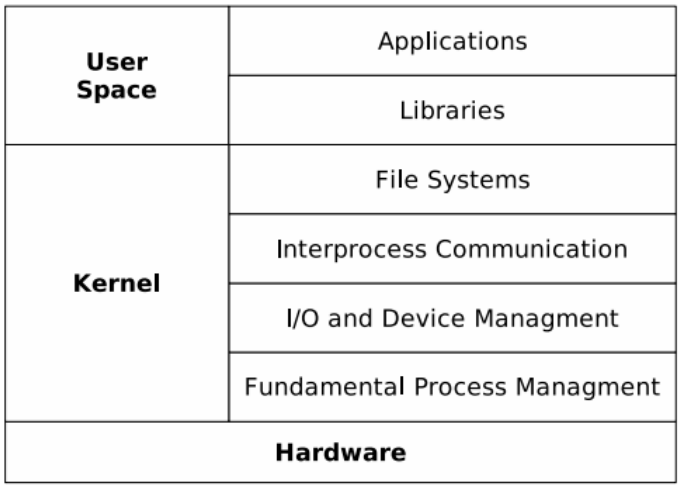
Kernel is the central module of an OS. It is the part of OS that loads first and remains in the main memory. As it stays in the main memory, it is important for the kernel to be as small as possible. Kernel is the main module which provides services to the other part of OS and application software. The Kernel code is usually stored in the protected area of the memory to prevent it from being overwritten by program and other part of OS.



(Kernel of the Operating System)

Kernels are of following type:

1. **Monolithic Kernel**
   - Traditional UNIX OS uses monolithic Architecture.
   - The Entire OS runs as single program in Kernel Mode. Programs contains OS core functions and devices drivers.
   - Most of the operations are performed by kernel via system call.
   - Windows-95/98, Linux and FreeBSD OS use monolithic Kernel.



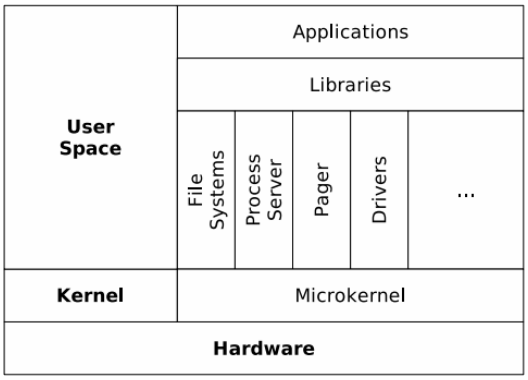
(Monolithic kernel)

- The monolithic is an older kernel used in UNIX, MS-DOS and early MAC-OS.
- In the monolithic kernel, every basic service like the process and memory management, interrupt handling and I/O communication; file system, etc. run in kernel space. It is constructed in a layered fashion, application at the top, then the libraries in user-space.

  To overcome the drawbacks of the monolithic kernel, microkernel came into existence

2. **Micro Kernel**

- Microkernel provides minimal services like defining memory address space, IPC and process management.
- In the micro-kernel, process resides in user-space in the form of servers.
- It has small operating core.



- Communication between kernel space and user space is done via **message parsing** which allows independent communication

Difference between microkernel and monolithic kernel:

Basis for Comparison	Microkernel	Monolithic Kernel
Size	Microkernel is smaller in size	It is larger than microkernel
Execution	Slow Execution	Fast Execution
Extendible	It is easily extendible	It is hard to extend
Security	If a service crashes, it does effects on working on the microkernel	If a service crashes, the whole system crashes in monolithic kernel.
Code	To write a microkernel more code is required	To write a monolithic kernel less code is required
Example	QNX, Symbian, L4Linux etc.	Linux,BSDs(FreeBSD,OpenBSD,NetBSD)etc.

3. Explain Operating system as Resource manager! Clearly mention various components of Operating system.

As a resource Manager: The OS acts a manager of the computer system resources (like software and hardware and more specifically CPU time, memory space, file storage space, I/O devices etc) and allot them to specific program and uses as necessary for their tasks.

Since there may be many possibly conflicting requests for recourses, the operating system must decide which resource to operate the computer system efficiently and fairly.

System Components of Operating System:

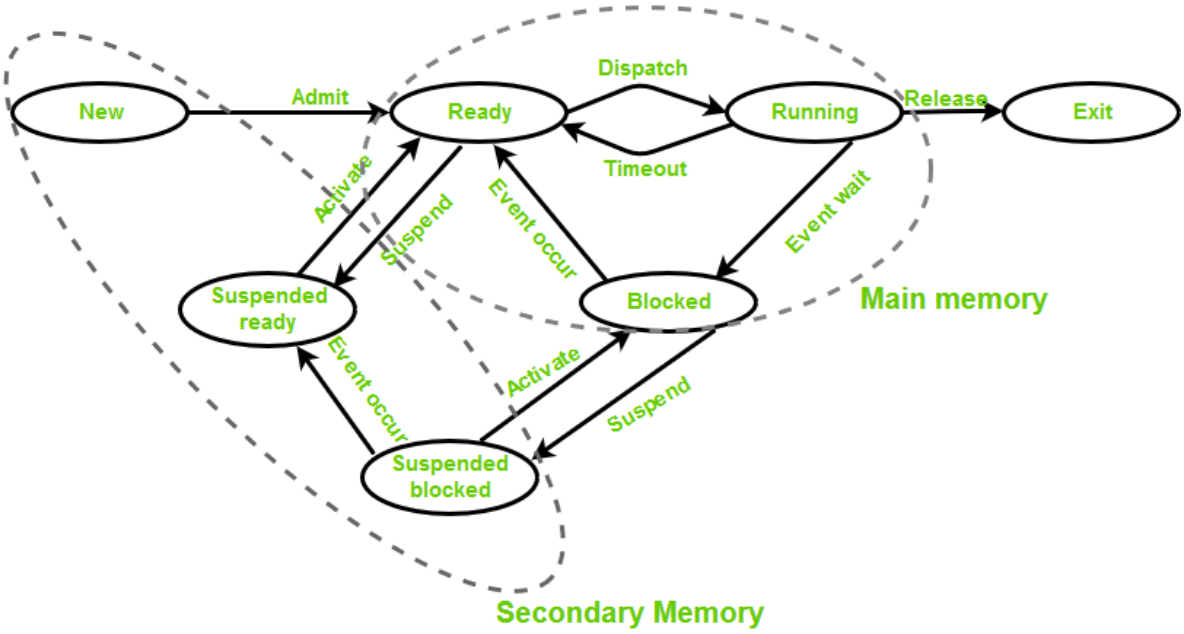
The components of an operating system all exist in order to make the different parts of a computer work together. Each component has its pre-defined functions to handle a specific task for operating system. These components include:

- Process Manager
- Main Memory Manager
- File Manager
- Secondary Storage Manager
- I/O Manager
- Networking Manager
- Protection System Manager
- Command Interpreter

Section – C

Note: Attempt Any Three Questions. 3 x 5 = 15 Marks

1. During the state transition, in which states, the process remains in Secondary memory and main memory? Explain, Seven State process diagram with proper diagram.



- **New (Create) –** In this step, the process is about to be created but not yet created, it is the program which is present in secondary memory that will be picked up by OS to create the process.
- **Ready –** New -> Ready to run. After the creation of a process, the process enters the ready state i.e. the process is loaded into the main memory. The process here is ready to run and is waiting to get the CPU time for its execution. Processes that are ready for execution by the CPU are maintained in a queue for ready processes.
- **Run –** The process is chosen by CPU for execution and the instructions within the process are executed by any one of the available CPU cores.
- **Blocked or wait –** Whenever the process requests access to I/O or needs input from the user or needs access to a critical region(the lock for which is already acquired) it enters the blocked or wait state. The process continues to wait in the main memory and does not require CPU. Once the I/O operation is completed the process goes to the ready state.
- **Terminated or completed –** Process is killed as well as PCB is deleted.
- **Suspend ready –** Process that was initially in the ready state but were swapped out of main memory (refer Virtual Memory topic) and placed onto external storage by scheduler are said to be in suspend ready state. The process will transition back to ready state whenever the process is again brought onto the main memory.
- **Suspend wait or suspend blocked –** Similar to suspend ready but uses the process which was performing I/O operation and lack of main memory caused them to move to secondary memory. When work is finished it may go to suspend ready.

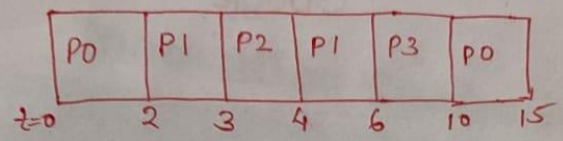
2. Consider the following processes:

Processes	Burst Time	Arrival Time	Priority
P0	7	0	3
P1	3	2	2
P2	1	3	0
P3	4	5	1

Draw the Gantt Chart for SRTF (SJF-Preemptive) and Preemptive Priority CPU Scheduling. Also Calculate the average turnaround time, average waiting time and average response time. (Priority:0>1>2>3)

	Burst Time	Arrival Time	Priority
P0	7	0	3
P1	3	2	2
P2	1	3	0
P3	4	5	1

SRTF



TT

P0 = 15-0 = 15
P1 = 6-2 = 4
P2 = 4-3 = 1
P3 = 10-5 = 5

$\frac{25}{4} = 6.25$

WT

P0 = 15-7 = 8
P1 = 4-3 = 1
P2 = 1-1 = 0
P3 = 5-4 = 01

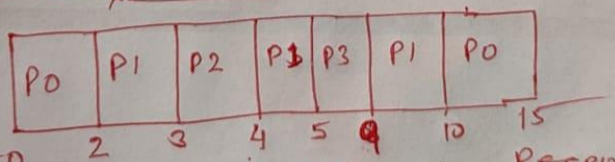
$\frac{10}{4} = 2.5$

Response Time

P0 = 0
P1 = 2-2 = 0
P2 = 3-3 = 0
P3 = 5-1 = 1

$1/4 = 0.25$

Preemphve Prority



TT

P0 15-0=15.
P1 10-2=8.
P2 4-3=01
P3 9-5=4

$\frac{28}{4} = 7$

WT

15-7=8
8-3=5
1-1=0
4-4=0

$\frac{13}{4} = 3.25$

Respm TT

P0 = 0
P1 = 2-2=0
P2 = 3-3=0
P3 = 5-5=0

0

3. What is the role of schedulers? Which scheduler decide the degree of multiprogramming? Explain CPU scheduler with diagram.

SCHEDULERS

Schedulers are special system software which handles process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types −

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

Long Term Scheduler

It is also called a **job scheduler**. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling.

The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the system.

On some systems, the long-term scheduler may not be available or minimal. Time-sharing operating systems have no long term scheduler. When a process changes the state from new to ready, then there is use of long-term scheduler.

Short Term Scheduler

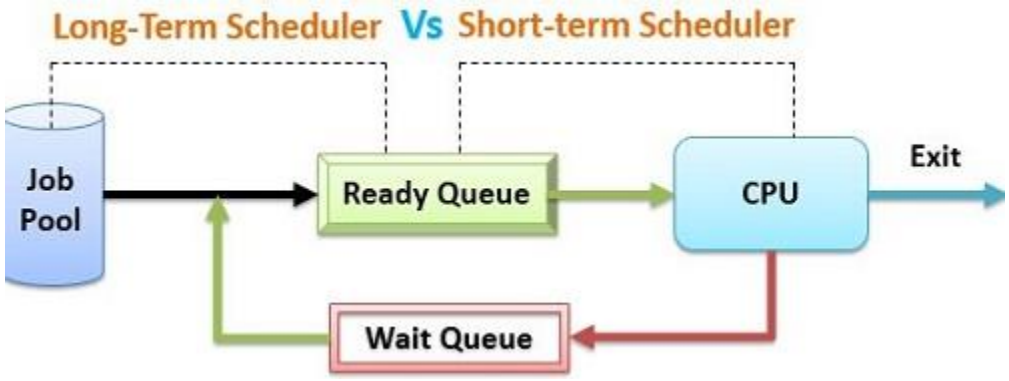
It is also called as **CPU scheduler**. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.

Short-term schedulers, also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers.

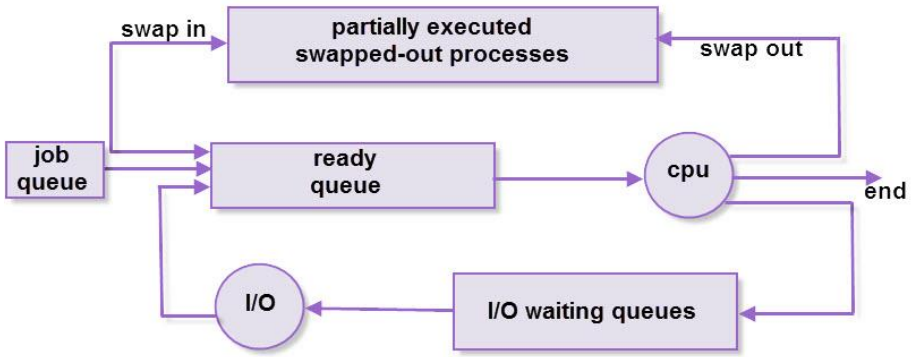
A process is selected from queues by the appropriate scheduler.

- In a batch system, more processes are submitted than can be executed immediately. These process are spooled to a mass storage device (typically a disk), where they are kept for later execution.
- The **short term scheduler** (**CPU SCHEDULER**) select from among the process that ready to execute, and allocate the CPU to one of them.

- The Primary distinction between these schedulers is the frequency of their execution. The short term schedulers must select a new process for the CPU quite frequently. Because of the short duration of time between executions, the short term scheduler must be very fast. Short term scheduler execute at least once in every 100 milliseconds.
- The long term scheduler on the other hand execute much less frequently. There may be minutes between the creations of new processes in the system. The long term scheduler controls the degree of multiprogramming (the no. of processes in the memory). If the degree of multiprogramming is stable, then the average rate of processes creation must be equal to the average departure rate of processes leaving the system.
- Thus, the long term schedulers may need to be invoked only when a process leaves the system. Because of the longer interval between executions the long term scheduler can afford to take more time to decide which process should be selected for execution.
- The long term scheduler select good process mix of I/O bound and CPU Bound processes. If all processes are I/O bound, the ready queue will almost be empty and short term scheduler will have little to do.



- Some Operating System such as Time Sharing System may introduce an additional intermediate level of scheduling. This **Medium Term Scheduler** sometimes removes processes from memory (and From active contention for the CPU) and thus reduce the degree of multiprogramming.
- At Later time, the process can be reintroduced into memory and its execution can be continued where it is left off. This scheme is **Swapping**. The Process is swapped out and swapped in later by **Medium Term Scheduler.**
- **Swapping** may be necessary to improve process mix or because a change in memory requirements has over committed available memory, requiring memory to be freed up.



Addition of medium-term scheduling to the queuing diagram

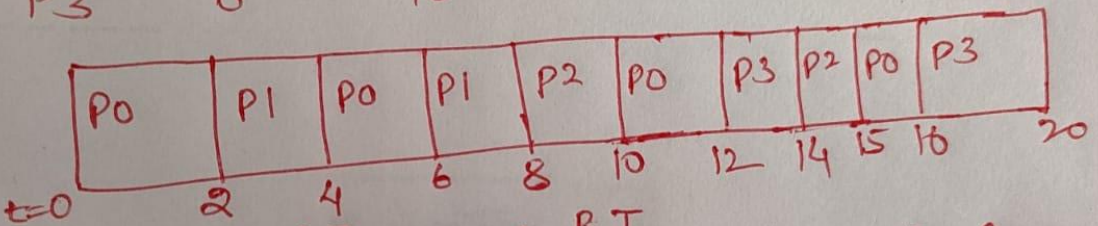
4. Consider the following processes:

Processes	Burst Time	Arrival Time
P0	7	0
P1	4	2
P2	3	5
P3	6	10

Draw the Gantt Chart for Round Robin (Time Slice=2) and calculate the average turnaround time, average waiting time and average response time.

		Arrival	Time
P0	7 5 3 1	0	
P1	4 2 0	2	
P2	3 0	5	
P3	6	10	

TS = 2



	TT	WT	RT
P0	16-0=16	16-7=9	P0 0-0=0
P1	8-2=6	6-4=2	2-2=0
P2	15-5=10	10-3=7	8-5=3
P3	20-10=10	10-6=4	12-10=2
	42/4	10.5	5/4 = 1.25

Ready Queue