1. **Alice can read file X, can append to file Y, and can write to file Z. Bob can append to file X, can write to file Y, and cannot access file Z. Design the access control matrix M that specifies the described set of access rights for subjects Alice and Bob to objects file X, file Y and file Z.**

Answer:

	File X	**File Y**	**File Z**
Alice	Read	Append	Write
Bob	Append	Write	

2. **Write the set of access control lists and the set of capability list for the previous question (Q1).**

Answer:

Access Control List:

- ACL(File X) = Alice: {read}, Bob: {append}
- ACL(File Y) = Alice: {append}, Bob: {write}
- ACL(File Z) = Alice: {write}, Bob: {}

Capability List:

1. CList(Alice) = FileX: {read}, FileY: {append}, FileZ: {write}
2. CList(Bob) = FileX: {append}, FileY: {write}, FileZ: {}

شرح على اليتويب

https://www.youtube.com/watch?v=OdOiu4Y1794&list=PLyepPYpoFSfgx52fS7CBqP7wElumDh5dI&index=12

3. What is port scanning and how is it typically launched?

Answer: Port scanning is not an attack but rather is a means for a cracker to detect a system's vulnerabilities to attack. Port scanning typically is automated, involving a tool that attempts to create a TCP/IP connection to a specific port or a range of ports. Because port scans are detectable, they are frequently launched from zombie systems.

4. What is the difference between symmetric and asymmetric encryption?

Answer: In a symmetric encryption algorithm, the same key is used to encrypt and to decrypt. In an asymmetric encryption algorithm, there are different encryption and decryption keys. Asymmetric encryption is based on mathematical functions instead of the transformations used in symmetric encryption, making it much more computationally expensive to execute.

5. What is a process, and how is it managed in Windows 7?

Answer: A process is an executing instance of an application containing one or more threads. Threads are the units of code that are scheduled by the operating system. A process is started when some other process calls the CreateProcess routine, which loads any dynamic link libraries used by the process, resulting in a primary thread. Additional threads can also be created. Each thread is created with its own stack with a wrapper function providing thread synchronization

6. File Operation ?

1. Create.

2. Write – at write pointer location.

3. Read – at read pointer location.

4. Reposition within file – seek.

5. Delete.

6. Truncate.

7. Open (Fi) – search the directory structure on disk for entry Fi, and move the content of entry to memory.

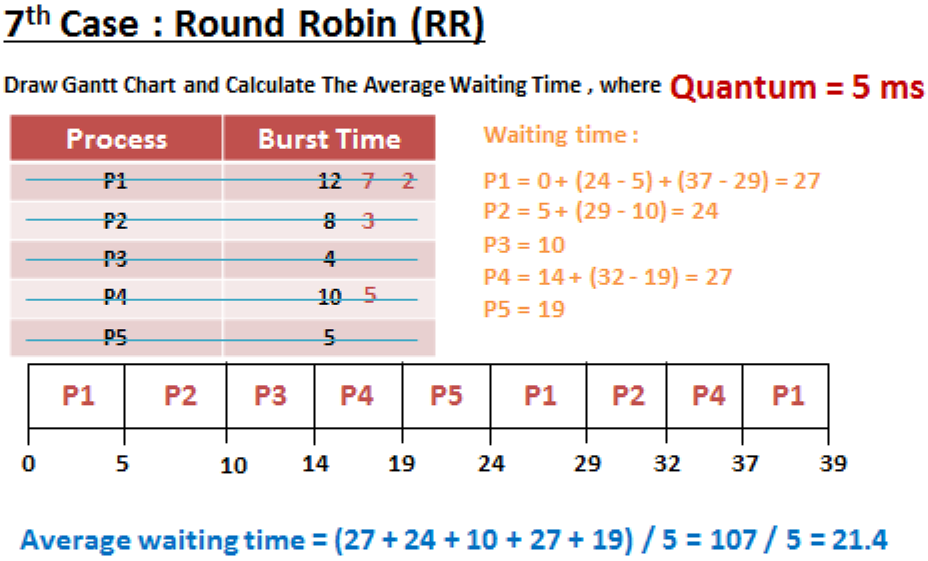
8. Close (Fi) – move the content of entry Fi in memory to directory structure on disk.

7. Bootstrap program?

is loaded at power-up or reboot
1. Typically stored in ROM or EPROM, generally known as firmware
2. Initializes all aspects of system
3. Loads operating system kernel and starts execution

8. round robin scheduling (RR)

https://www.youtube.com/watch?v=w_24qrbcBTs



7th Case : Round Robin (RR)

Draw Gantt Chart and Calculate The Average Waiting Time , where Quantum = 5 ms

Process	Burst Time
~~P1~~	~~12 7 2~~
~~P2~~	~~8 3~~
~~P3~~	~~4~~
~~P4~~	~~10 5~~
~~P5~~	~~5~~

Waiting time :

P1 = 0 + (24 - 5) + (37 - 29) = 27
P2 = 5 + (29 - 10) = 24
P3 = 10
P4 = 14 + (32 - 19) = 27
P5 = 19

Average waiting time = (27 + 24 + 10 + 27 + 19) / 5 = 107 / 5 = 21.4

9. Disk scheduling :

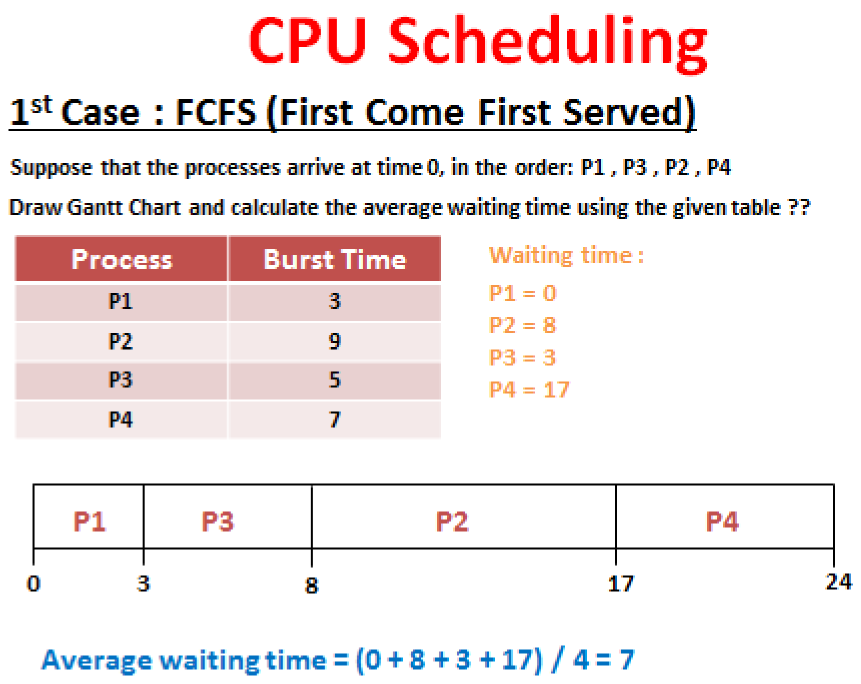
https://www.youtube.com/watch?v=T-Rvkofm19I

10. Page replacement:

https://www.youtube.com/watch?v=lwmqgip-hN4

11. CPU scheduling :

https://www.youtube.com/watch?v=XXrx9ricm_w



CPU Scheduling

1st Case : FCFS (First Come First Served)

Suppose that the processes arrive at time 0, in the order: P1 , P3 , P2 , P4

Draw Gantt Chart and calculate the average waiting time using the given table ??

Process	Burst Time
P1	3
P2	9
P3	5
P4	7

Waiting time :

P1 = 0

P2 = 8

P3 = 3

P4 = 17

P1	P3	P2	P4

0 3 8 17 24

Average waiting time = (0 + 8 + 3 + 17) / 4 = 7

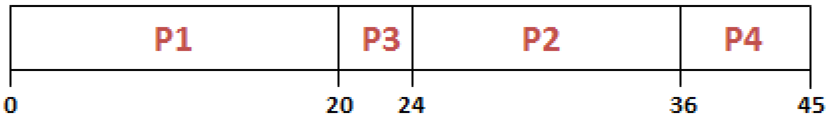
2nd Case : FCFS (First Come First Served)

Draw Gantt Chart and calculate the average waiting time using the given table ??

Process	Burst Time	Arrival Time
P1	20	0
P2	12	3
P3	4	2
P4	9	5

Waiting time : start time – arrival time

P1 = 0 – 0 = 0
P2 = 24 – 3 = 21
P3 = 20 – 2 = 18
P4 = 36 – 5 = 31



Average waiting time = (0 + 21 + 18 + 31) / 4 = 70 / 4

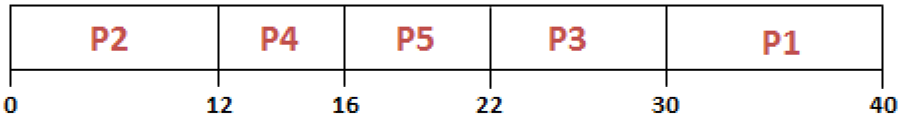
3rd Case : SJF (short job first) non-Preemptive

Draw Gantt Chart and calculate the average waiting time using the given table ??

Process	Burst Time	Arrival Time
P2	12	0
P3	8	3
P4	4	5
P1	10	10
P5	6	12

Waiting time : start time – arrival time

P1 = 30 – 10 = 20
P2 = 0 – 0 = 0
P3 = 22 – 3 = 19
P4 = 12 – 5 = 7
P5 = 16 – 12 = 4



Average waiting time = (20 + 0 + 19 + 7 + 4) / 5 = 50 / 5 = 10

4th Case : SJF (short job first) Preemptive

Draw Gantt Chart and calculate the average waiting time using the given table ??

Process	Burst Time	Arrival Time
~~P2~~	~~12 9~~	~~0~~
~~P3~~	~~8 6~~	~~3~~
~~P4~~	~~4~~	~~5~~
~~P1~~	~~10~~	~~10~~
~~P5~~	~~6~~	~~12~~

Waiting time : start time – arrival time

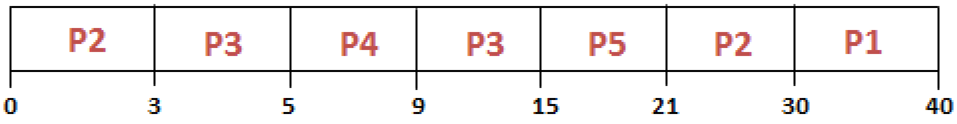
P1 = 30 – 10 = 20

P2 = (0 – 0) + (21 - 3) = 18

P3 = (3 – 3) + (9 - 5) = 4

P4 = (5 – 5) = 0

P5 = 15 – 12 = 3



Average waiting time = (20 + 18 + 4 + 0 + 3) / 5 = 45 / 5 = 9

5th Case : Priority Scheduling non-Preemptive

Draw Gantt Chart and calculate the average waiting time using the given table ??

Process	Burst Time	Priority	Arrival Time
P1	10	3	All Processes Arrived at The Same Time
P2	1	1	
P3	2	4	
P4	1	5	
P5	5	2	

Waiting time :
start time – arrival time

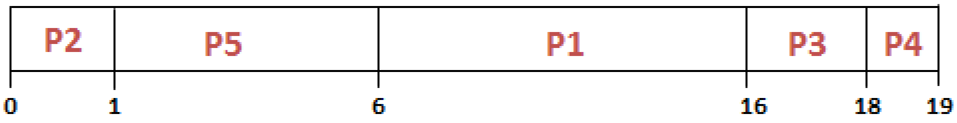
P1 = 6

P2 = 0

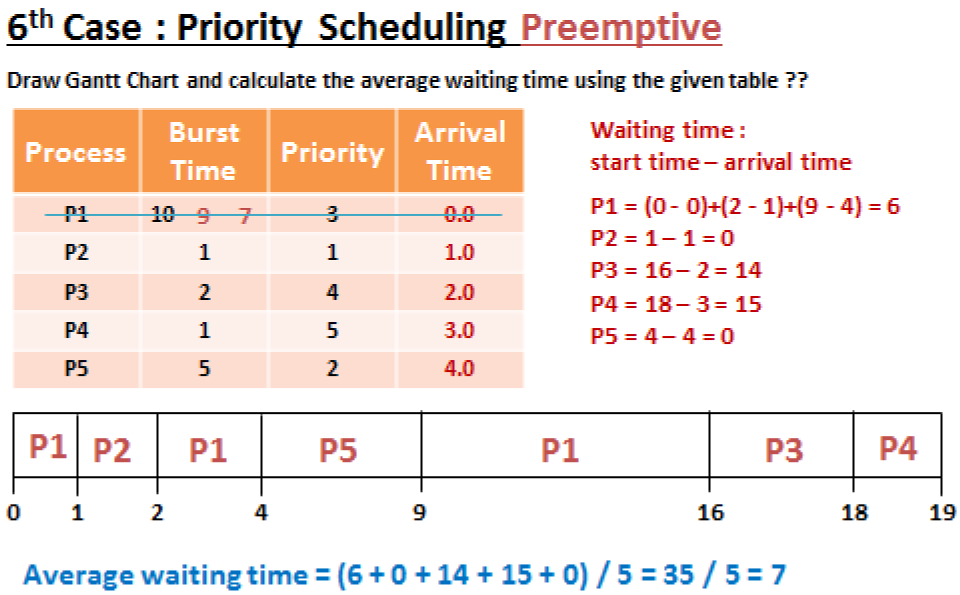
P3 = 16

P4 = 18

P5 = 1



Average waiting time = (6 + 0 + 16 + 18 + 1) / 5 = 41 / 5 = 8.2



6th Case : Priority Scheduling Preemptive

Draw Gantt Chart and calculate the average waiting time using the given table ??

Process	Burst Time	Priority	Arrival Time
~~P1~~	~~10 9 7~~	~~3~~	~~0.0~~
P2	1	1	1.0
P3	2	4	2.0
P4	1	5	3.0
P5	5	2	4.0

Waiting time :
start time – arrival time

P1 = (0 - 0)+(2 - 1)+(9 - 4) = 6
P2 = 1 – 1 = 0
P3 = 16 – 2 = 14
P4 = 18 – 3 = 15
P5 = 4 – 4 = 0

P1	P2	P1	P5	P1	P3	P4

0 1 2 4 9 16 18 19

Average waiting time = (6 + 0 + 14 + 15 + 0) / 5 = 35 / 5 = 7

12. Explain the basic method for implementing paging.

Answer: Physical memory is broken up into fixed-sized blocks called frames while logical memory is broken up into equal-sized blocks called pages. Whenever the CPU generates a logical address, the page number and offset into that page is used, in conjunction with a page table, to map the request to a location in physical memory.

13- What is Context Switching?

Ans. When CPU switches to another process, the system must save the state of the old process and load the saved statefor the new process via Context Switch. context of a process represented in the PCB.