19BCE245 2CS403 4 20/02/2021 1

Q-1 A. System calls for file management:

Interface between a process and operating system is provided by system calls.

System calls are usually made when a process in user mode requires access to a resource. Then it requests the kernel to provide a resource via system calls.

File management:

These system calls are responsible for file manipulation such as creating a file, reading a file, writing into device buffer etc.

There are four system calls for file management → open(), read(), write(), close().

1.) open()

↳ system call which is used to know the file descriptor of user-created files, since read and write use the file descriptor, as their first parameter, so to know the descriptor open() system call is used.

syntax: fd = open (file-name, mode, permission);

here → file-name is the name of the file which we want to open.

→ mode: is used to define file opening mode: create, read, write.

→ permission: used to define permissions.

2.) read().

↳ which is used to read content from the file. It can also be used to read input from the keyboard (by specifying 0 in the file descriptor)

Syntax: read (file-~~name~~ descriptor, buffer, max-length)

file-descriptor → name of the buffer where data is to be stored → maximum amount of data can be read.

2.) Difference between multiprogramming and multitasking

→ Multiprogramming: A computer running more than one programs at a time.

→ Multitasking: Tasks which are sharing a common resource (like One CPU)

Multiprogramming	Multitasking
→ Concept of context switching is used.	→ concept of context switching and time-sharing is used
→ In this system, the OS simply switches and executes another job when current job needs to wait	→ In this, processor is used in time sharing mode. here switching happens when allowed time expires or there is other reason for current process needs to wait
→ here the idea is to reduce CPU idle time for as long as possible.	→ here, The idea is to further extend the CPU utilization by increasing responsiveness time sharing
→ this increases CPU utilization by organising jobs.	→ this also increases CPU utilization with increasement in responsiveness

19BCE245 2CS403 4 20/02/2021

3 days

Q-2 A.) Long Term Scheduler controls the degree of multiprogramming. (also called as Job scheduler).

Q-2 B.) Scheduling criteria.

1.) CPU utilisation.

- The main objective of any CPU scheduling algo is to keep the CPU as busy as possible.
- CPU utilization can range from 0 to 100. but in a real time-system, it varies from 40 to 90 percent which depends on the load upon system.

2.) Throughput

- A measure of the work done by system CPU is the number of processes being executed and completed per unit time. This is called throughput
- The throughput may very depending upon the length or duration of the process.

3.) Turnaround time:

- How long the particular process takes to execute is the important criteria for that process.
- The time elapsed from the time of submission of a process to the time of completion is known as turn around time.
- This is the sum of time spent waiting to get into memory, waiting in ready queue, executing and waiting for I/O.

4.) Waiting time:

- A scheduling algo. doesn't affect the time required to complete the process once it starts execution.

- It only affects the waiting time of the process i.e. time spent by a process waiting in ready queue.

5.) Response time
- Turn-around time is not the best criteria in an interactive system.
- A process may produce some output fairly early and computing new results while previous results are being output to the user.
- Therefore, another criteria is the time taken from submission of the process of request until the first response is produced. This measure is called response time.

Q-2 c) ▸Race condition:
- A race condition is an undesirable situation that occurs when a device or system attempts to perform two or more operations at the same time, but because of the nature of the device or system, the operation must be done in the proper sequence to be done correctly.

▸Avoiding Race
- In order to prevent race conditions from occurring, we would typically put a lock around the shared data to ensure only one thread can share the data at a time.

- To avoid the race condition, we need mutual exclusion here Mutual exclusion is someway of making sure that if one process is using a shared variable or file, the another processes will be excluded from doing the same things.

Q-2 c continue. The part of the program where the shared memory is accessed is called the critical region or critical section.

If we could arrange matters such that no two processes were ever in their critical regions at the same time, we could avoid race conditions.

▸ Rules for avoiding Race conditions:

1. No two processes may be simultaneously inside their critical regions. [Mutual exclusion]
2. No assumptions can be made about speeds or the number of CPUs.
3. No process running outside its critical region may block other processes.
4. No process should have to wait forever to enter its critical region.

1D). An I/O is interface is required whenever the I/O device is driven by processor.

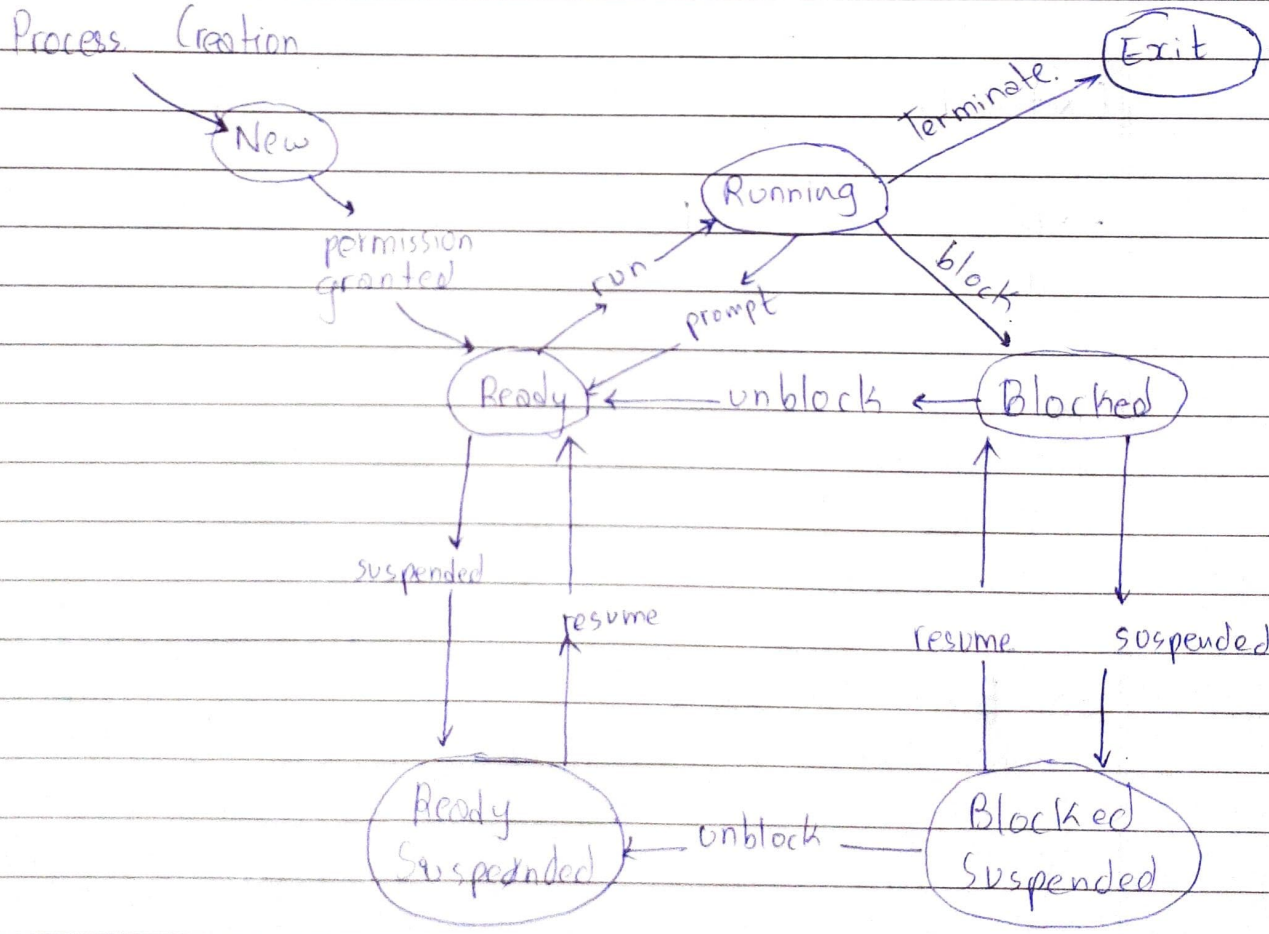
~~Memory~~ Memory mapped memory I/O.

In this case a cartesian portions of the process address space is mapped to the device & communication occurs by reading and writing directly from I/O

19BCE2245 2CS403 20/02/2021 6

Q-2 E Seven states of process model:

1. New
2. ready
3. running
4. Blocked
5. Blocked suspended
6. Ready suspended
7. Exit



New: when the process is newly created

Ready: when the process is created and ready to run

Ready suspended: When the process leaves the ready state and suspended.

Blocked: when the process is blocked after running state.

Blocked suspended: When the process is blocked and also suspended.

Running: when the process is running on the CPU.

Exit: Process leaves the CPU and RAM.

→ Ready suspend: Process that was initially in the ready state but were swapped out of main memory and placed onto external storage by schedular 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.

→ Block suspend:
when work is finished it may go to suspend ready

19BCE245 2CS403 4 20/02/2021

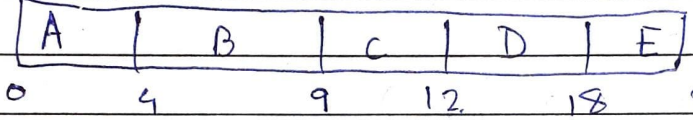
8. Days

Q-3.

Process	Arrival time	Burst time
A	0	4
B	2	5
C	3	3
D	4	6
E	5	2.

1.) First come First serve.

Gantt chart:



CT	TAT	WT.
4	4	0
9	7	2
12	9	6
18	14	8
20	15	13

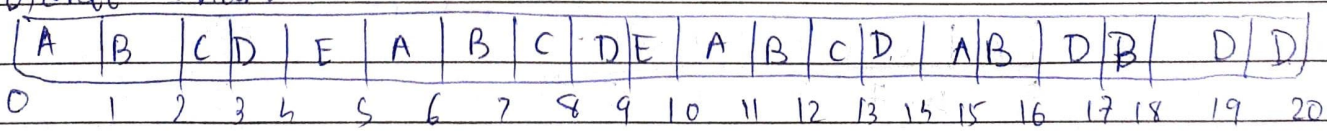
Avg TAT = 9.8 ms.
Avg WT = 5.8 ms

2.) Round Robin (qt = 1 ms.)

Ready queue

A B C D E A B C D E A B C D

Gantt chart.



CT	TAT	WT.
15	15	11
18	16	17
13	10	7
20	15	10
10	5	3

Avg TAT = 12.4 ms.
Avg WT = 8.4 ms.

vision

19BCE245 2CS403 4 20/02/2021 9 Days

3.) Round Robin (qt = 3 ms)

RQ | A | B | C | D | E | A | B | D |

CT	TAT	WT
15	15	11
17	15	10
9	6	3
20	16	10
14	9	7

Avg TAT = 12.2 ms
Avg WT = 8.2 ms

4.) Shortest job first (SJF)

Gantt chart.

A	C	E	B	D
0	4	7	9	14	20

CT	TAT	WT
4	4	0
14	12	7
7	4	1
20	16	10
9	4	2

Avg TAT = 8 ms
Avg WT = 5 ms

6.) SRTN (SRTN)

A	C	E	B	D
0	4	7	9	14	20

→ Gantt chart

A
B
C
D
E

CT	TAT	WT

Avg TAT = 8 ms
Avg WT = 4 ms

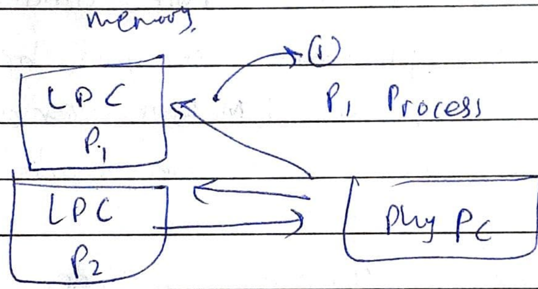
vision

19BCE245 2CS403 4 20/02/2021

8/10

1C.)

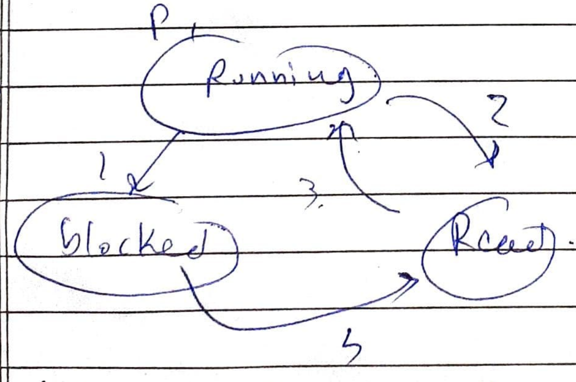
Process ID
State
Priority
Program counter
I/O information
⋮
etc.



order.

(1) CPU allocated to P1

2.D.)



Process state transition.

1.) Running – blocked Process blocks wait for input or wait for an event or resource that is needed

2.) Running – Ready when process is either completed or terminated due to timeout ie. end of time slice

3.) Ready → Running → Schedular picks the process

4.) Blocke – ree input becomes available as discussed in first point.