PAGE NO.:
DATE:

Assignment: 1

(1) Why are the primary purpose of operating system. Why should operating system prevent user from accessing boot access.

Ans: The primary purpose of an operating system are:

- To provide an environment for a computer user to execute programs on computer hardware in a convenient and efficient manner.
- To allocate the seperate resources of the computer as needed to solve the problem given. The allocation process should be as fair and efficient as possible.

As a control program it serves two major functions:

(i) Supervision of the execution of user program to prevent errors and improper use of the computer and.

(ii) management of the operation and control of I/O devices.

2) Explain various types of OS:

Ans: The various types of operating system are

i) Batch Processing:

In batch processing system, the grouping of similar job which consist of program data and system commands are done. The instructions, data and some controlled information are submitted to the computer operator in the form of job. The users are not allowed to interact with the computer system. The jobs are performed in FIFO manner, so the OS require very simple CPU scheduling techniques. The problems in

DATE:

this system are:
- Lack of interaction between user and the job.
- CPU is often idle, because the speed of mechanical I/O devices is slower than the CPU.
- Difficult to provide the desired priority.

ii) Multi-Processing:

A computer's capability to process more than one task simultaneously is called multiprocessing. A multiprocessing operating system is capable of running many programs simultaneously, and most moder network operating system (NOSs) support multiprocessing. These operating systems include Windows NT, 2000, XP and UNIX.

Advantages:
- Increased throughput
- Economy of scale increased.

Disadvantages:
- If one processor fails then it will affect the speed.
- Multiprocessor systems are expensive.
- Complex OS is required.
- Large main memory required.

iii) Time sharing operating system:

It is a technique which enables many peoples located at various terminals, to use a particular system at the same time. Time-sharing or multitasking is a logical extension of multiprocessing. Processor's time which is shared among multiple user simultaneously is termed as time sharing.

Advantages:
- Provides the advantage of quick response.
- Avoids duplication of software.
- Reduces CPU idle time.

Disadvantages:
- Problem of reliability
- Question of security and integrity of user program and data
- Problem of data communication.

iv.) Real time system:

A real-time system is defined as data processing system in which the time interval required to process and respond to inputs is so small that it controls the environment. It is a multitasking operating system that aims at executing real-time application.

It is of two types:

a.) Hard real-time system

It guarantees that critical task complete on time.

b.) Soft real-time system

It is less restrictive.

v.) Network Operating System

A network OS runs on a server and provides the server the capability to manage data, users, groups, security applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multi computers in a network.

It is further divided into two types:

a. Peer-to-peer

It allows users to share resources and files located on their computers and to access shared resource found on other computers.

b. Client/server

It allows the network to centralize functions and applications in one or more dedicated file servers.

E.g: Novell Netware, Windows 2000 Server.

vi) Distributed Operating System

A distributed OS in an operating system that runs on several machines. Its purpose is to provide a useful set of services, generally to make the collecti of machines behave more like a single machine.

Advantages:

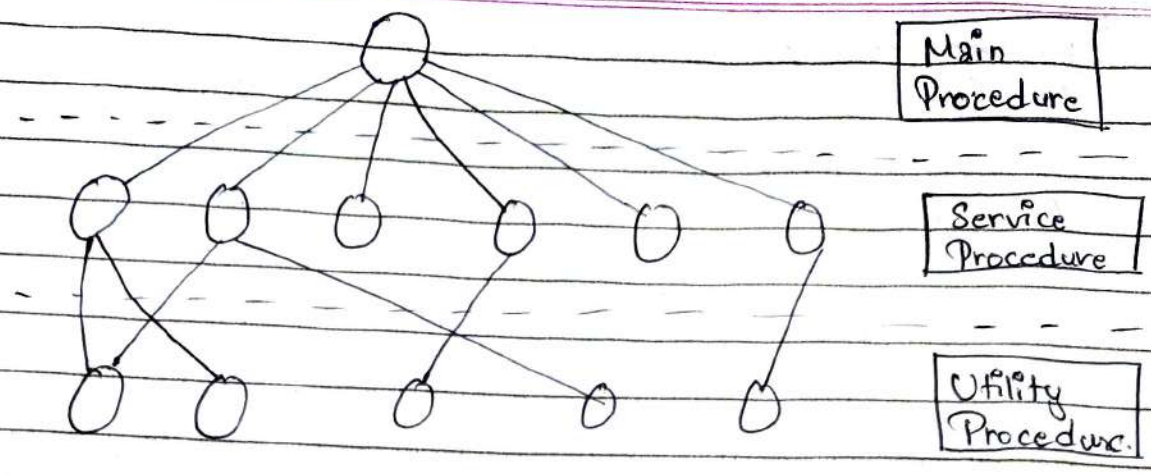
- Sharing of resources
- Reliability
- Communication
- Computation speedup.

(3.) Explain the structure of OS with its advantages and disadvantages.

Ans: The common system structures are given as:

i. Monolithic:

- Here, the kernel is a single large program.
- Functionality of the OS is invoked with simple function calls with the kernel.
- Device drivers are loaded into the running kernel and become part of the kernel.
- System calls from the user programs are kept in trap table which are executed in kernel mode switching from user mode.



ii. Layered structure:

- Hierarchy of layers, each constructed upon another below it.
- Layer 0 (hardware) to layer N (user interface).
- Provide modularity.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers.
- But difficult in differentiation and less efficient.
- First system constructed in this way --- 'THE SYSTEM'
- 'THE SYSTEM' was simple batch system consisting six layers.

The six layers are:

Layer 5	User program
Layer 4	Buffering for input and output
Layer 3	Process management
Layer 2	Memory management
Layer 1	CPU scheduling
Layer 0	Hardware.

iii. Microkernel

- Basic idea is to achieve high reliability by splitting of OS into small, well defined modules.
- Only one among those modules always reside in memory &

DATE:

always run on kernel mode.
- Others run as user process (device driver, file system).

Advantages:
- Easier to extend a microkernel.
- Easier to port the OS to new architecture.

iv. Client-server architecture.
- Two classes of processes - Server & Client
- Communication between client and server is via message passing.
- Client and server can run on different computers connected by LAN/WAN.
- Servers run as user mode. Hence, no system down if the server crashed.
- Well adapted in distributed system.

v. Virtual Machine
- It is an illusion of a real machine operating system, which make a single real machine appear to be several real machine.
- In this system, each user can choose a different OS.

Advantages:
- Complete protection of system resource.
- No direct sharing of resources.

Disadvantages:
- Difficult to implement.

PAGE NO.:
DATE:

4. Explain the views on Operating System

⇒ The views on operating system are given as:

a. User mode

The user view depends on the system interface that is used by the users.

- If the user is using a personal computer, the operating system is largely designed to make interaction easy.
- If the user is using a system connected to a mainframe computer or a minicomputer, the operating system is largely connected with resource utilization.
- If the user is sitting on a workstation connected to other workstations through networks, then OS needs to focus on both individual ~~use~~ usage of resources & sharing through the network.
- If the user is using a hand held computer such as a mobile, then the OS handles the usability of the device including some remote application

b. System view

According to computer system, the OS is the bridge between application and hardware.

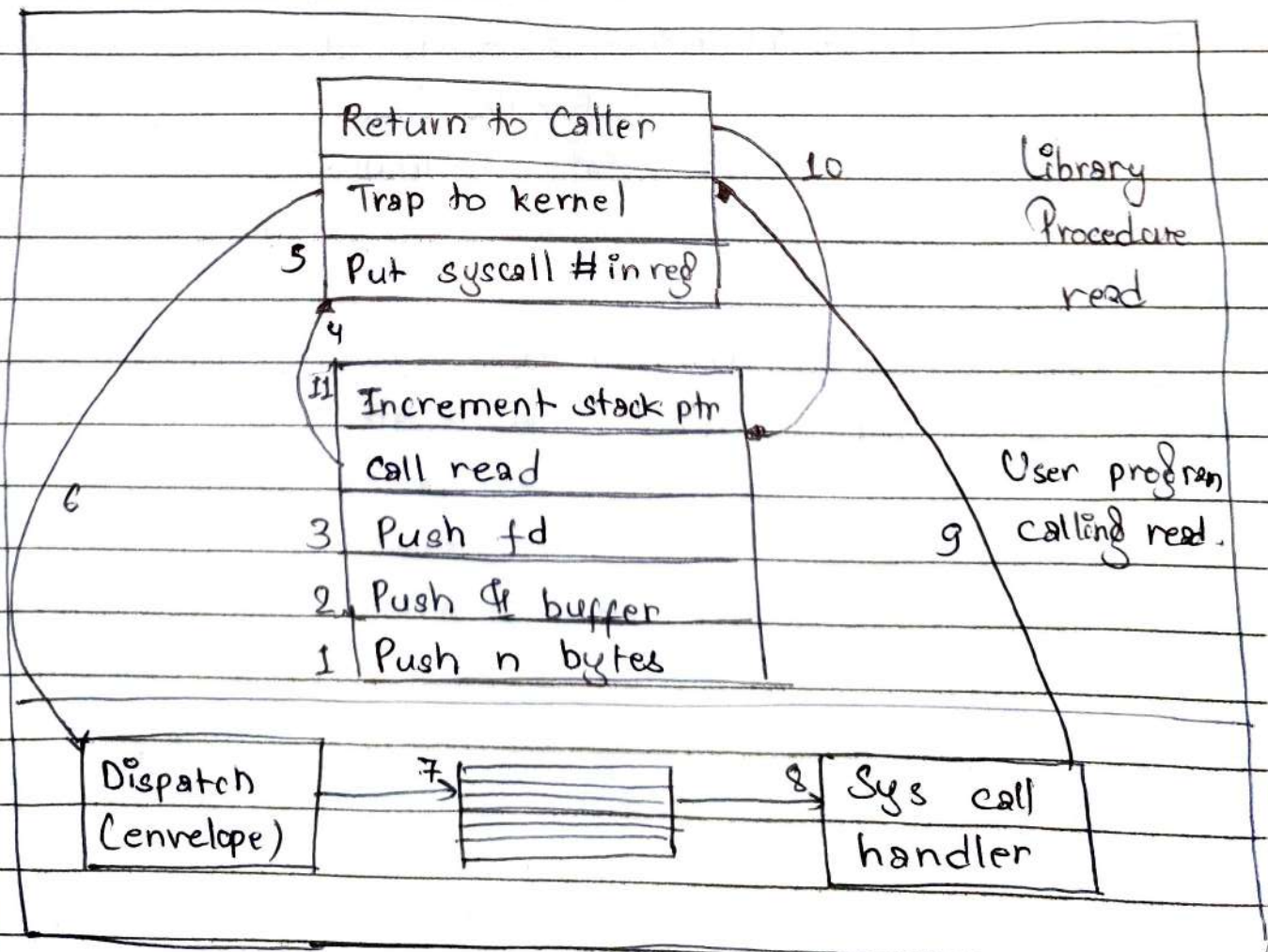
- The system view the OS as a resource allocator.
- The OS can also work as a control program
- OS can also be viewed as a way to make hardware easier.
- Operating system were developed to easily communicate with hardware.

DATE:

5.) Define system calls. Illustrate the working of a system call with appropriate example.

Ans: A system call is how a program requests to a service from an operating system's kernel. This may include hardware related services, creating and executing new processes and communicating with integral kernel services. System call provides an essential interface between a process and the Operating system.

For read (fd, buffer, n bytes);
The system call returns the number of bytes actually read in count. This value is normally the same as n bytes, but may be smaller, if for example end-of-file is encountered while reading.
It is represented in figure below:



PAGE NO.: _____
DATE:

6) Define shell and kernel along with its function.

Ans: Shell & kernel

When a user gives his command for performing any operation, then the request will go to the shell parts, the shell parts is also called as the interpreter which translates the human program into the machine language and then the request will be transferred to the kernel that means shell is just as the interpreter of the commands which converts the request of user into the machine language.

Functions of kernel:

- It controls the state of the process ie. it checks whether the process is running or is waiting for the request of the user.
- It provides the memory for the processes those are running on the system ~~mean~~ ie. kernel runs the allocation and deallocation process.
- It also maintains a time-table for all the processes those are running.

7) Operating system provides abstraction. Justify.

Ans: Operating system provides a set of basic commands or instructions to perform various operations such as read, write, modify, save or close. Also dealing with them is easier than directly dealing with hardware. Thus, operating system hides the complexity of hardware and presents a beautiful interface to the user. Just as the operating system shields the programmer from the disk hardware and

presents a simple file-oriented interface, it also conceeds a lot of unpleasent business concerning interrupts, memory management and other low-level factors. In each case, the abstraction offered by the operating system is simple and easier to understand than offered by underlying hardware.

(8) Explain the various components of OS:

Ans: The components of operating system are:

i) Process management (Scheduling)

A process is the unit of work in a system. Operating system manages the execution of those process. OS is responsible for the following activities in connection with process managed.

- The creation and deletion of both user & system proceesses.
- The suspension is resumption of proccesses.
- The provisim of mechanism for process synchronisation.
- The provisim of mechanism for deadlock handling.

ii) I/O management:

One of the important job of OS is to manage various I/O devices. I/O system requires to take an application I/O request and send it to physical device, then take whatever response come back from the device and send it to application.

iii) Main memory management:

The operating system is responsible for the following activities in connection with memory management:

- keeping track of which part of memory is being used.
- Decide which process are to be loaded into memory when memory space becomes available.
- Allocate and deallocate memory space as needed.

iv) Secondary storage management:

The operating system is responsible for the following activities in connection with disk management
- free space management
- Storage allocation
- Disk scheduling

v) File management system

The operating system is responsible for the following activities in connection with file management
- The creation and deletion of files.
- The creation and deletion of directory
- The support of primitives for manipulating files and directories
- The mapping of files onto disk storage.
- Backup of files on stable storage.

vi) Security & Protection

The various processes in an OS must be protected from each other's activities. For that purpose, various mechanisms which can be used to ensure that the files,

memory segment, CPU and other resources can be operated by the processes that have gained proper authorization from the operating system.

(9.) Write short notes on:

(a) Windows

Windows is a personal computer's operating system from Microsoft Corporation that together comes with some commonly used applications. Windows have become a 'standard' for common users in most organizations as well as in most homes. Microsoft Windows is a series of operating system and graphical user interface produced by Microsoft. Microsoft first introduced an operating environment named Windows as MS-DOS.

(b) UNIX

UNIX is a computer operating system originally developed in 1969 by a group of AT&T employee at Bells Lab.

UNIX was originally meant to be a convinient platform for programmers developing software to be run on it and on other systems, rather than non-programmers.

(c) Under UNIX, the operating system consist of many libraries and utilities along with the master control program, the kernel.

PAGE NO.:
DATE:

(c) Linux:

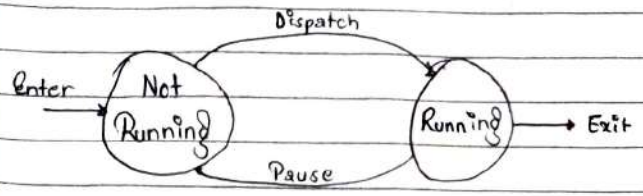
A linux-based system is a modular Unix-like operating system, deriving much of its basic design from principles established in Unix during 1970s & 1980s. Such a system uses monolithic kernel, the Linux kernel, which handles process control, networking access to the peripherals and file system. Device drivers are either integrated directly with the kernel, or added as modules that are loaded with the system is running.

Assignment - 2

① Define process. Explain the process model along with the transition.

Ans. A process is defined as an entity which represents the basic unit of work to be implemented in the system i.e. a process is a program in execution. There are two types of process models:

i) Two state model:



State transition diagram

The two states are:

a. State 1: Process is running on CPU
State 2: Process is "Not Running" on CPU

New: First of all, when a new process is created, then it is in Not Running State. Suppose a new process P2 is created then P2 is in NOT Running state

CPU: When CPU becomes free, Dispatcher gives control of the CPU to P2 that is in NOT Running state & waiting in queue.

Dispatcher: Dispatcher is a ~~person~~ program that gives control of CPU to the process selected by the CPU schedules.

Running: When dispatcher allows P2 to execute on CPU then P2 starts its execution.

ii) Five state model

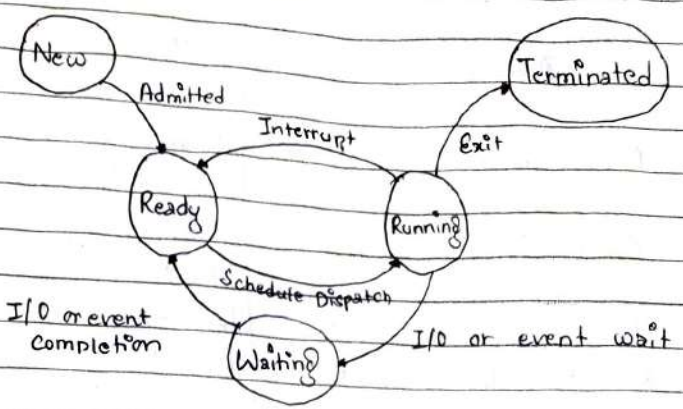


Fig: Five state model.

The five states are:

1. New: A process that has just been created but has not yet been admitted to the pool of executable process by the OS.
2. Ready: Process that is prepared to execute when given the opportunity. That is, they are not waiting on anything except the CPU availability.
3. Running: The process that is currently being executed
4. Blocked (Waiting): A process that cannot execute until some event occurs.
5. Exit: A process that has been released from the pool of executable processes by the OS.

(2) Differentiate between process & thread:

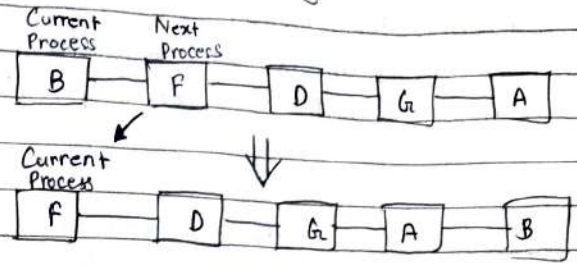
Process	Thread
1. Doesn't share memory Closely coupled)	Shares memories and files (Tightly coupled)
2. Creation is time consuming.	Fast.
3. Execution slow	Fast.
4. More time to terminate	less time.
5. More time to switch between process	less time.
6. System calls are required for communication	Not required.
7. More resources are required.	Fewer resources are required.
8. Not suitable for parallelism	Suitable for parallelism.

(3) What is context switching? Explain the processes with example.

Ans: Context switching:

A context switching is the mechanism to store and restore the state or context of a CPU in process control block so that a process execution can be resumed from the same point at a later time.

Using this technique, a context switcher enables multiple processes to share a single CPU. When the scheduler switches the CPU from executing one process to execute another, the state from the current process is stored into its control block. After this, the state for the process to run next is loaded from its own PCB and used to set the PC, registers, etc. At that point, the second process can start executing.



(4) Differentiate between user-level thread and kernel-level thread:

User-level thread	Kernel-level thread
1. User-level threads are faster to create and manage.	Kernel-level threads are slower to create and manage.
2. Implementation is by a thread library at the user-level.	Operating system supports creation of Kernel threads.
3. User-level thread is generic & can run on any OS.	Kernel-level thread is specific to the OS.
4. Multi-threaded applications can't take advantage of multiprocessing.	Kernel routines themselves can be multithreaded.

PAGE NO.:
DATE:

(5) Define scheduling. Explain scheduler & its types. Define dispatcher.

Ans: Scheduling:

The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

Scheduler:

It is a special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system & to decide which process to run.

There are three types of scheduler:

- Long term scheduler
- Mid term schedules
- Short term schedules.

1. Long Term Scheduler:

The long term scheduler basically decides the priority in which processes must be placed in main memory. Processes of long term scheduler are placed in the ready state because in this state the process is ready to execute waiting for calls of execution from CPU which takes time that's why this is known as long-term schedule.

Mid Term Scheduler:

It places the blocked and suspended processes in the secondary memory of a computer system.

3. Short Term Scheduler:

It decides the priority in which processes is in the ready queue are allocated the CPU time for their execution.

Dispatcher:

Dispatcher is the module that gives control of the CPU to the process selected by the short time scheduler (selects from among the processes that are ready to execute).

(6) Define the following terms:

a. CPU utilization

Keeping the CPU as busy as possible. Theoretically, it's value ranges from 0% to 100% but in practice, it is 40% to 90%.

b. Throughput:

Throughput is the rate at which processes are completed per unit of time.

c. Turnaround time:

It is the time taken by a process to execute. It is the difference between the completion time and the submission time.

d. Waiting time:

It is the sum of the time period spent waiting in queu

e. Response time:

It is the time taken to start responding from submission time.

(7) f) Burst time
The duration for which a process gets control of the CPU is burst time.

(7) Differentiate between preemptive & non-preemptive scheduling.

Ans:

	Preemptive Scheduling	Non-preemptive scheduling
1.	The resources are allocated to a process for a limited time	Once resources are allocated to a process, the process holds it till it completes its burst time or switches to waiting state.
2.	Process can be interrupted in between.	Process cannot be interrupted till it terminates or switches to waiting state.
3.	Preemptive scheduling has overheads of scheduling the process.	It does not have overheads
4.	It is flexible.	It is rigid.

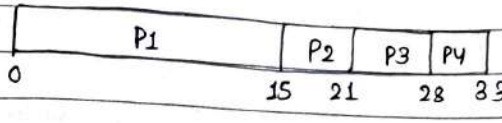
(8) Calculate the Average Turnaround Time & Average Waiting Time for following.
Using: FCFS, SJN, SRTN, Priority, Round-Robin, HRRN
Quantum = 2

★ First Come First Serve (FCFS)

(a)

Process	Arrival Time	Burst time
P1	0	15
P2	2	6
P3	3	7
P4	5	5

i) FCFS:
Gantt Chart



Process	Arrival Time (AT)	Burst Time (BT)	Finish Time (FT)	Turnaround Time (TAT)	Waiting Time (WT)
P1	0	15	15	15	0
P2	2	6	21	19	13
P3	3	7	28	25	18
P4	5	5	33	28	23

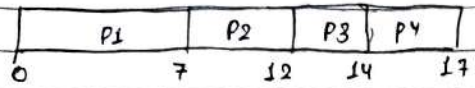
∴ Average Turnaround Time ($\overline{TAT}$) = $\frac{15+19+25+28}{4}$ = 21.75

Average Waiting Time ($\overline{WT}$) = $\frac{0+13+18+23}{4}$ = 13.5 //

(b)

Process	AT	BT
P1	0	7
P2	1	5
P3	3	2
P4	4	3

Gantt Chart:



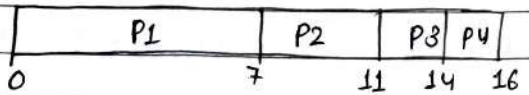
Process	AT	BT	FT	TAT	WT
P1	0	7	7	7	0
P2	1	5	12	11	6
P3	3	2	14	11	9
P4	4	3	17	13	10

∴ Average Turnaround Time = $(7+11+11+13)/4 = 10.5$ ■
Average Waiting Time = $(0+6+9+10)/4 = 6.25$ //

(c)

Process	AT	BT
P1	0	7
P2	1	4
P3	3	3
P4	4	2

⇒ Gantt Chart:



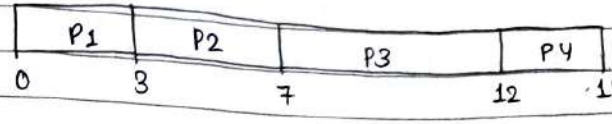
Process	AT	BT	FT	TAT	WT
P1	0	7	7	7	0
P2	1	4	11	10	6
P3	3	3	14	11	8
P4	4	2	16	12	10

∴ $\overline{TAT} = 10$
$\overline{WT} = 6$ //

(d)

Process	AT	BT
P1	0	3
P2	2	4
P3	3	5
P4	4	2

⇒ Gantt Chart:



Process	AT	BT	FT	TAT	WT
P1	0	3	3	3	0
P2	2	4	7	5	1
P3	3	5	12	9	4
P4	4	2	14	10	8

∴ Average Turnaround Time = $\frac{3+5+9+10}{4} = 6.75$

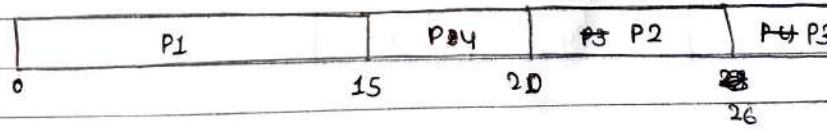
Average Waiting Time = $\frac{0+1+4+8}{4} = 3.25$ //

★ Shortest Job Next (SJN):

(a)

Process	Arrival Time	Burst Time
P1	0	15
P2	2	6
P3	3	7
P4	5	5

⇒ Gantt Chart:



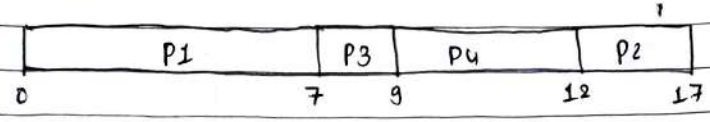
Process	AT	BT	FT	TAT	WT
P1	0	15	15	15	0
P2	2	6	26	24	18
P3	3	7	33	30	23
P4	5	5	20	15	10

∴ Average Turnaround Time = 21

Average Waiting Time = 12.75 //

(b) Here,

Gantt Chart:



Process	AT	BT	FT	TAT	WT
P1	0	7	7	7	0
P2	1	5	17	16	11
P3	2	2	9	7	5
P4	3	3	12	9	6

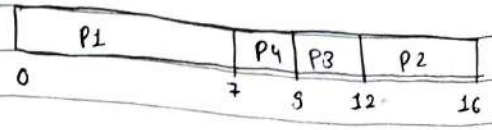
∴ Average Turnaround Time ($\overline{TAT}$) = (7+16+7+9)/4 = 9.75

Average Waiting Time ($\overline{WT}$) = (0+11+5+6)/4 = 5.5 //

(c)

Process	AT	BT
P1	0	7
P2	1	4
P3	3	3
P4	4	2

Gantt Chart:



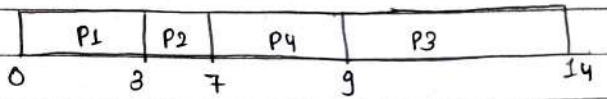
Process	Arrival Time (AT)	Burst Time (BT)	Finish Time (FT)	TAT	WT
P1	0	7	7	7	0
P2	1	4	16	15	11
P3	3	3	12	9	6
P4	4	2	9	5	3

∴ Average Turnaround Time ($\overline{TAT}$) = (7+15+9+5)/4 = 9

Average Waiting Time ($\overline{WT}$) = (0+11+6+3)/4 = 5 //

(d) Here,

Gantt Chart:



Process	AT	BT	FT	TAT	WT
P1	0	3	3	3	0
P2	2	4	7	5	1
P3	3	5	9	6	1
P4	4	2	14	10	8

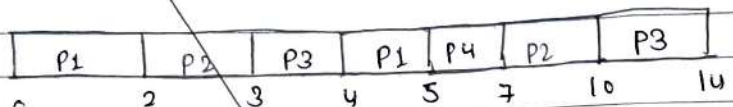
∴ Average Turnaround Time = $\frac{3+5+6+10}{4}$ = 6

Average Waiting Time = $\frac{0+1+1+8}{4}$ = 2.5 //

④

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

⇒ Gantt Chart:



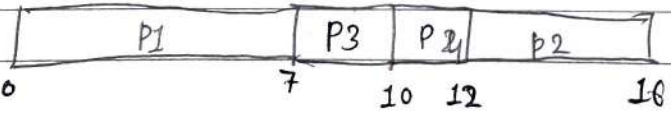
Process	AT	BT	FT	TAT	WT
P1	0	3	5	5	2
P2	2	4	10	8	4
P3	3	5	14	11	6
P4	4	2	7	3	1

∴ Average TAT = 6.75
Average WT = 4.25 //

★ Priority Scheduling:

Process	AT	BT	Priority
P1	0	7	4
P2	1	4	3
P3	3	3	1 (High)
P4	4	2	2

⇒ Gantt Chart:



Process	AT	BT	FT	TAT	WT
P1	0	7	7	7	0
P2	1	4	16	15	11
P3	3	3	10	7	4
P4	4	2	12	8	6

∴ Average TAT = 9.25
Average WT = 5.25 //

★ Round Robin Scheduling:

⑤

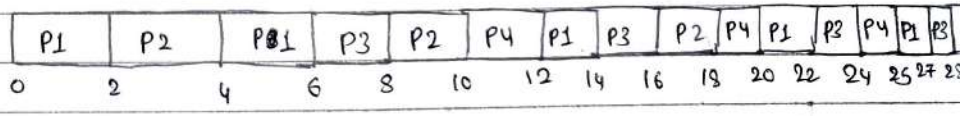
Process	Arrival Time	Burst Time
P1	0	15
P2	2	6
P3	3	7
P4	5	5

Quantum = 2ms

⇒ Queue:

P1 P2 P1 P3 P2 P4 P1 P3 P2 P4 P1 P3 P4 P1 P3 P1

Chart:



Process	AT	BT	FT	TAT	WT
P1	0	15	33	33	18
P2	2	6	18	16	14
P3	3	7	28	25	18
P4	5	5	25	20	15

$\therefore \overline{TAT} = 23.5$

$\overline{WT} = 16.25$ //

(b)

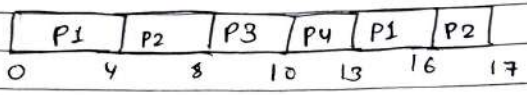
Process	AT	BT
P1	0	7
P2	1	5
P3	3	2
P4	4	3

Quantum = 4 sec

Queue:

~~P1 P2 P3 P4 P1~~ P2

Chart:



Process	AT	BT	FT	TAT	WT
P1	0	7	16	16	9
P2	1	5	17	16	11
P3	3	2	10	7	5
P4	4	3	13	9	6

$\therefore$ Average TAT = 12

Average WT = 7.75 //

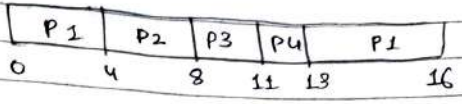
Process	AT	BT
P1	0	7
P2	1	4
P3	3	3
P4	4	2

Quantum = 4.

Queue:

~~P1 P2 P3 P4 P1~~

Chart:



Process	AT	BT	FT	TAT	WT
P1	0	7	16	16	9
P2	1	4	8	7	3
P3	3	3	11	8	5
P4	4	2	13	9	7

$\therefore$ Average TAT = 10

Average WT = 6 //

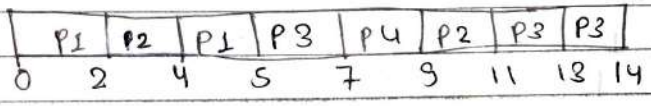
(d)

Process	AT	BT
P1	0	3
P2	2	4
P3	3	5
P4	4	2

Quantum = 2.

Queue: ~~P1 P2 P1 P3 P4 P2 P3~~

Chart:



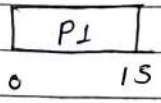
Process	AT	BT	FT	TAT	WT
P1	0	3	5	5	2
P2	2	4	11	9	5
P3	3	5	14	11	6
P4	4	2	9	5	3

∴ Average TAT = 7.5
Average WT = 4 //

★ HRRN

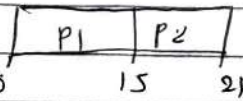
(a)

Process	AT	BT
P1	0	15
P2	2	6
P3	3	7
P4	5	5



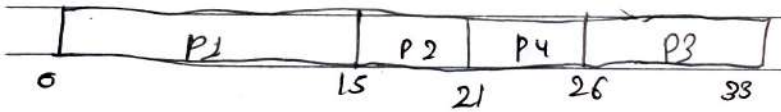
Response Ratio:

P2 = (13+6)/6 = 3.166
P3 = (12+7)/6 = 2.714
P4 = (10+5)/5 = 3



Response Ratio:

P3 = (18+7)/7 = 8.571
P4 = (16+5)/5 = 4.2



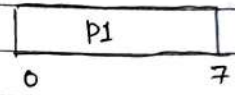
Process	AT	BT	FT	TAT	WT
P1	0	15	15	15	0
P2	2	6	21	19	13
P3	3	7	33	30	23
P4	5	5	26	21	16

∴ Average TAT = 21.25
Average WT = 13.11

(b)

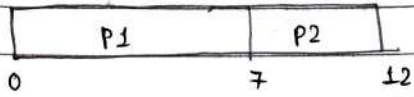
Process	AT	BT
P1	0	7
P2	1	5
P3	3	2
P4	4	3

Chart:



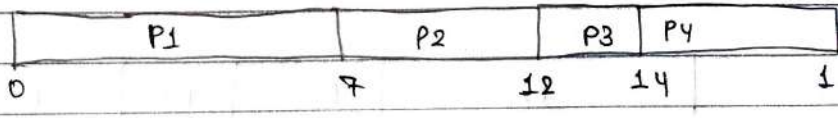
Response Ratio:

P2 = ((7-1)+5)/5 = 5.5
P3 = ((7-3)+2)/2 = 3
P4 = ((7-4)+3)/3 = 2



Response Ratio:

P3 = (9+2)/2 = 5.5 ; P4 = (8+3)/2 = 5.5



Process	AT	BT	FT	TAT	WT
P1	0	7	7	7	0
P2	1	5	12	11	6
P3	3	2	14	11	9
P4	4	3	17	13	10

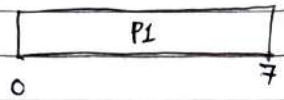
∴ Average TAT = (7+11+11+13)/4 = 10.5

Average WT = (0+6+9+10)/4 = 6.25 //

(c)

Process	AT	BT
P1	0	7
P2	1	4
P3	3	3
P4	4	2

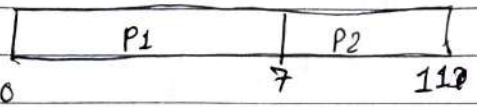
Chart:



Response Ratio: P2 = (6+4)/4 = 2.5

P3 = (4+3)/3 = 2.33

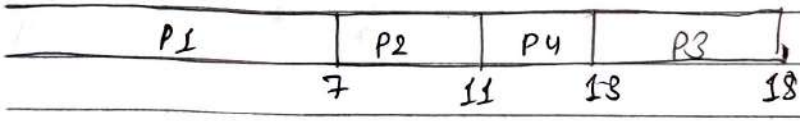
P4 = (3+2)/2 = 2.5



Response Ratio:

P3 = $\frac{8+3}{3}$ = 3.667

P4 = $\frac{7+2}{2}$ = 4.5



Process	AT	BT	FT	TAT	WT
P1	0	7	7	7	0
P2	1	4	11	10	6
P3	3	3	16	13	10
P4	4	2	13	9	7

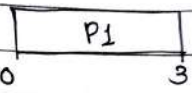
∴ Average Turnaround time = 9.75

Average Waiting time = 5.75 //

(d)

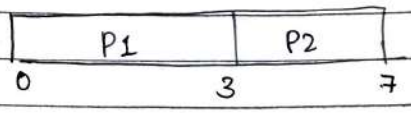
Process	Arrival Time	Burst Time
P1	0	3
P2	2	4
P3	3	5
P4	4	2

Chart:



Response Ratio: P2 = (1+4)/4 = 1.25

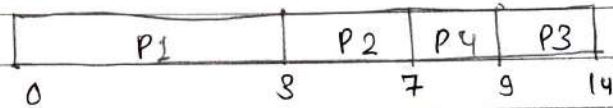
P3 = (0+5)/5 = 1



Response Ratio:

P3 = $\frac{4+5}{5}$ = 1.8

P4 = $\frac{8+2}{2}$ = 2.5



Process	AT	BT	FT	TAT	WT
P1	0	3	3	3	0
P2	2	4	7	5	1
P3	3	5	14	11	6
P4	4	7	9	5	3

∴ Average Turnaround time = $\frac{(3+5+11+5)}{4} = 6$

Average Waiting time = $\frac{0+1+6+3}{4} = 2.5$ ms

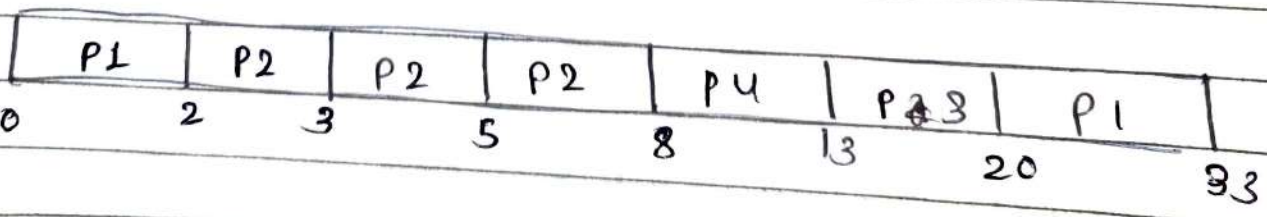
posted

★ Shortest Remaining Time Next:

a)

Process	Arrival Time	Burst Time
P1	0	15
P2	2	6
P3	3	7
P4	5	5

⇒ Gantt Chart.



Process	AT	BT	FT	TAT	WT
P1	0	15	33	33	17
P2	2	6	8	6	0
P3	3	7	20	17	10
P4	5	5	13	8	3

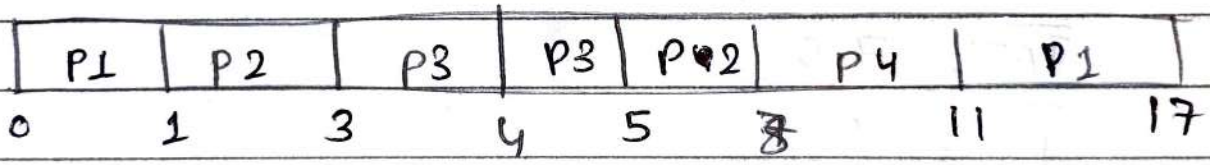
$\therefore \overline{TAT} = 16$

$\overline{WT} = 7.5$ //

(b)

Process	AT	BT
P1	0	7
P2	1	5
P3	3	2
P4	4	3

Gantt Chart:



Process	AT	BT	FT	TAT	WT
P1	0	7	17	17	10
P2	1	5	7	6	1
P3	3	2	5	2	0
P4	4	3	11	7	4

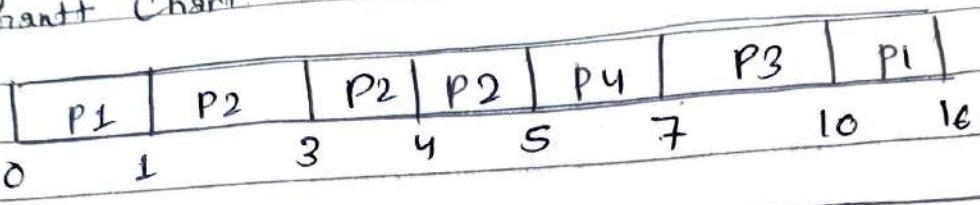
$\therefore \overline{TAT} = 8$

$\overline{WT} = 3.75$ //

c)

Process	AT	BT
P1	0	7
P2	1	4
P3	3	3
P4	4	2

Gantt Chart:



Process	AT	BT	CT	TAT	WT
P1	0	7	16	16	9
P2	1	4	5	4	0
P3	3	3	10	7	4
P4	4	2	7	3	1

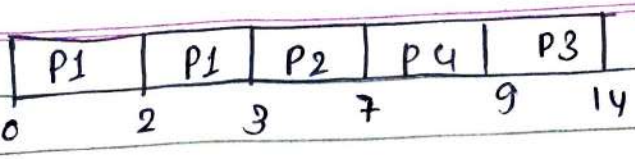
$\therefore \overline{TAT} = 7.5$

$\overline{WT} = 3.5$

d)

Process	AT	BT
P1	0	3
P2	2	4
P3	3	5
P4	4	2

Gantt Chart:



Process	AT	BT	FT	TAT	WT
P1	0	3	3	3	0
P2	2	4	7	5	1
P3	3	5	14	11	6
P4	4	2	9	5	3

$\therefore \overline{TAT} = 6$

$\overline{WT} = 2.5$.

DATE:

Chapter 8: Process Synchronization & Communication.

2075 Bhadra

Q. What is critical section problem? Why must the executing the critical section be mutually exclusive? Describe how semaphores can be used to solve the critical section problem?

Ans: Critical section is the part of a program which tries to access shared resources. That resource may be any resource in a computer like memory location, data structure, CPU or any IO device. The critical section cannot be executed by more then one process at the same time; operating system faces the difficulte in allowing and disallowing the processes from ~~entering~~ entering the critical section.

The critical section problem is used to design a set of protocols which can ensure that the Race condition among the processes will never arise.

Mutual exclusion means if a process is executing in its critical section, then no other process is allowed to execute in the critical section. Since, shared resources are kept in the critical section and if more than one process is access the critical section, the alteration caused by one may cause error in the execution of another. So, executing the critical section must be mutually exclusive.

```
int s = 1;
wait( Semaphore s)
{
    while (s == 0);
```

```
    s = s - 1;
}
signal (Semaphore s)
{
    s = s + 1;
}
```

Implementation of semaphore:

```
do
{
    wait(s); // critical section.
    signal(s); // remainder section.
} while (1);
```

Let there be two process P1 and P2 and a semaphore initialised as 1. Now if suppose P1 enters in its critical section then it will wait unit $s > 0$, this call only happen when P1 finishes its critical section and calls signal() operation on semaphore s. This way mutual exclusion is achieved and critical section problem is solved.

2074 Bhadra

Q. What is race condition? Explain how Sleep and Wakeup() solution is better than busy waiting solution for critical section problem.

Ans. When two or more processes are reading or writing some shared data and the final results depends on who runs precisely when, are called race condition.

In busy waiting solution, processes waiting to enter their critical sections waste processor time

checking to see if they can proceed. Busy waiting has the following disadvantages

- waste of processor time
- Possibility of deadlock/ starvation in systems with multipriority or scheduling.

Sleep() and wakeup() eliminates this problem. By calling sleep() the calling process is suspended till being woken by other process called wakeup()

Q. What is TSL? Why it is used? Explain the major operations of semaphore with a simple implementation as a class.

Ans: Test and Set lock (TSL) is a synchronization mechanism. It uses a test and set instruction to provide the synchronization among the processes executing concurrently.

It is used to implement mutual exclusion.

The major operations of semaphore are:

- wait(): called when a process wants access to a resource.
- signal(): called when a process is done using a resource.

```
class semaphore {
public:
    semaphore (int n): n-(n) {
    }
    void wait() {
        --n;
    }
```

DATE :

```
void signal() {
        ++n;
    }
private:
    std :: atomic <int> n_;
};
```

2073 Bhadra

Q. Explain critical section problem. Why is it important for a thread to execute a critical section as quickly as possible?

Ans. A thread must acquire a lock prior to executing a critical section. The lock can be acquired by only one thread. So, in order to release the lock, so that the other thread could access that critical section, a thread must execute a critical section as quickly as possible.

Q. Define Semaphore and explain the major operations in semaphore including pseudocode.

Ans: Semaphore is a variable used to solve the critical section problem and to achieve process synchronization in the multiprocessing environment. The major operations of a semaphore are:

- wait()
- signal()

Pseudocode.

```
int s = 1;
wait (Semaphore s)
{
```

```
    while (s==0);
    s=s-1;
}
signal (Semaphore s)
{
    s=s+1;
}
```

2073 Magh:

Q. What are the requirements of mutual exclusion? Solve producer - consumer problem using semaphore and message passing.

Ans. Requirements of mutual exclusion

1. No two processes may be simultaneously inside their critical region.
2. No assumption may be made about speeds or 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.

Solution of Consumer-Producer problem using semaphore.

```
#define N 100
typedef int semaphore
semaphore mutex = 1;
semaphore empty = N;
semaphore full = 0;
void producer (void)
{
```

```
int item;
while (TRUE)
{
    item = produce_item();
    down(&empty);
    down(&mutex);
    insert_item(item);
    up(&mutex);
    up(&full);
}
}
void consumer(void)
{
    int item;
    while (TRUE)
    {
        down(&full);
        down(&mutex);
        item = remove_item();
        up(&mutex);
        up(&empty);
        consumer.item(item);
    }
}.
```

2072 Ashwin:

Q. Solve producer - consumer problem using monitors.

Ans Monitors make solving the producer - consumer a little easier. Mutual exclusion is achieved by placing the critical section of a program inside a monitor. In the code below, the critical sections of

DATE:

the producer and consumer are inside the monitor, Producer Consumer. Once inside the monitor, a process is blocked by the Wait and Signal primitives if it cannot continue.

```
monitor Producer Consumer
    condition full, empty;
    int count;
producer enter();
{
    if (count == N)
```

```
monitor ProducerConsumer
{
    int itemCount = 0;
    condition full;
    condition empty;
    procedure add (item)
    {
        if (itemCount == BUFFER_SIZE)
        {
            wait (full);
        }
        putItemIntoBuffer (item);
        itemCount = itemCount + 1;
        if (itemCount == 1)
        {
            notify (empty);
        }
    }
```

```
procedure remove( )
{
    if (itemCount == 0)
    {
        wait(empty);
    }
    item = removeItemFromBuffer();
    itemCount = itemCount - 1;
    if (itemCount == BUFFER_SIZE - 1)
    {
        notify(full);
    }
    return item;
  }
}

procedure producer( )
{
    while (true)
    {
        item = produceItem();
        ProducerConsumer.add(item);
    }
}

procedure consumer()
{
    while (true)
    {
        item = ProducerConsumer.remove()
        consumeItem(item);
    }
}
```

PAGE NO.:
DATE:

2072 Magh:

Q. Why processes need to be synchronized ? Explain Peterson's Solution and TSL instruction approaches used in mutual exclusion with busy waiting.

Ans: Process synchronization means sharing system resources by processes in a such a way that, Concurrent access to shared data is handled thereby minimizing the chance of inconsistent data. Processes need to be synchronized for sharing of resources without interference using mutual exclusion.

Peterson's algorithm is a concurrent programming algorithm. It is used for mutual exclusion and allows two processes to share a single-use resource without conflict. It is given as.

```
#define FALSE 0
#define TRUE 1
#define N 2
int turn;
int interested[N];
void enter_region(int process);
{
    int other;
    other = 1 - process;
    interested[process] = TRUE;
    turn = process;
    while (turn == process && interested[other] == TRUE);
}
void leave_region(int process)
{
```

interested [process] - FALSE;
}

Before using the shared variables (i.e. before entering its critical region), each process calls enter_region with its own process number, 0 or 1, as parameter. This call will cause to wait, if needed be, until it is safe to enter. After it has finished with the shared variables, the process calls leave_region to indicate that it is done and to allow the other process to enter, if it so desires.

TSL Instruction:

```
enter region:
    TSL REGISTER LOCK
    CMP REGISTER, #0
    JNE enter region
    RET
leave region:
    MOVE LOCK, #0
    RET.
```

To use the TSL instruction, we will use a shared variable, lock to coordinate access to shared memory. When lock is 0, any process may set it to 1 using the TSL instruction and then read or write the shared memory. When it is done, the process set lock back to 0 using an ordinary move instruction.

The first instruction copies the old value of lock to the register and then set lock to 1. Then the old value is compared with 0. If it is nonzero,

the lock was already set, so the program just goes back to the beginning and tests it again. Sooner or later it will become 0 (when the process currently in its critical region is done with its critical region), and the subroutine returns, with the lock set. Clearing the lock is very simple. The program just stores a 0 in lock. No special synchronization instructions are needed.

2071 Magh

Q. Why do we need pipe() function? Define Semaphore and explain the major operations in semaphore. Can semaphores be used in distributed system? Explain why or why not.

Ans: We need pipe() function in order to create a connection between two processes, such that the standard output from one process becomes the standard input of the other process. Pipes are useful for communication between related processes (inter-process communication). Pipe can be used by the creating process, as well as all its child processes, for reading and writing.

Yes, semaphores can be used in distributed systems. A distributed system is a network of processors interconnected by a communication network. The processors do not share memory and exchange information through messages. The lack of shared memory makes implementation of semaphores very difficult in a distributed system. A

implementation of semaphores in such a system must rely on message passing. To protect a critical section in a distributed system, several algorithms based on message passing have been proposed. Those algorithms indirectly implement binary semaphore in distributed system.

2070 Bhadra

Q. Explain all possible approaches to handle the situation "while one process is busy updating shared memory, no other process will enter its critical section and cause trouble".

Ans. The various approaches to handle the situation are:

i) Disabling Interrupts:

On a single-processor system, the simplest solution is to have each process disable all interrupts just after entering its critical region and renable them just before leaving it. With interrupts disabled, no clock interrups can occur. The CPU is only switched from process to process as a result of clock or other interrupts, after all, and with interrupts turned off the CPU will not be switched to another process.

This approach is generally unattractive because it is unwise to give user processes the power to turn off interrupts.

ii) Lock variables:

This approach is based on software. Consider having a single, shared (lock) variable, initially 0. When a process want to enter its critical region, it first test the lock. If the lock is 0, the process sets it to 1 and enters the critical region. If the lock is already 1, the process just waits until it becomes 0. Thus, a 0 means that no

process in its critical region, and a 1 means that some process is in its critical region.

iii) Strict Alternation.

Two Process Solution:

```
P0: #define FALSE 0
    #define TRUE 1
    while (TRUE)
    {
        while(turn != 0);
        critical_region();
        turn = 1;
        noncritical_region();
    }
P1: #define FALSE 0
    #define TRUE 1
    while (TRUE)
    {
        while (turn != 1);
        critical_region();
        turn = 0;
        noncritical_region();
    }
```

A proposed solution to the critical region problem for Process '0' and Process '1'.

The integer variable turn, initially 0, keeps track of whose turn it is to enter the critical region and examine or update the shared memory. Initially, process 0 inspects turn, finds it to be 0 and enters its critical region. Process 1 also finds it to be 0 and therefore sits in a tight loop continually

testing turn to see when it becomes 1.

iv) Peterson's Solution:

Peterson's algorithm is a concurrent programming algorithm for mutual exclusion that allows two or more processes to share a single-use resource without conflict, using only shared memory for communication.

v) TSL Instruction:

Test and Set Lock (TSL) is a synchronisation mechanism. It uses a test and set instruction to provide the synchronization among the process executing concurrently.

It is an instruction that returns the old value of a memory location and sets the memory location value to 1 as a single atomic operation. If one process is currently executing a test-and-set, no other process is allowed to begin another test-and-set until the first process test-and-set is finished.

Chapter 4: Memory management.

PAGE NO.:
DATE:

2075 Bhadra

Q. What are the differences between fixed partitioning and variable partitioning system of memory for multiprogramming?

Given reference to the following pages by a program
0, 9, 0, 1, 8, 1, 8, 7, 8, 7, 1, 2, 8, 2, 7, 8, 2, 2, 8, 3
How many page faults will occur if the program has 4 frames for Optimal Page Replacement algorithm?

Ans:

Fixed Partitioning	Variable Partitioning
1. In fixed partitioning, each partition is of fixed size and contains only one process.	In variable partitioning, the amount of memory allocated is exactly the amount of memory the process requires
2. No. of process depends upon no. of partitions.	No. of processes in memory doesn't depend upon no. of partitions.
3. Memory cannot be used efficiently for small size processes.	Memory can be used efficiently for small size processes.

Here,
Let the four frames be f_1, f_2, f_3 and f_4.

f1	0	0	-	0	0			0				2		-			-	-		2
f2		9		9	9			7		-		7			-					7
f3				1	1	-		1			-	1								3
f4					8		-	8	-			8	-			-			-	8

PAGE NO.:
DATE:

No. of page faults: (20-13) = 7 //

2074 Bhadra.

Q What is thrashing? Consider the following page-reference storing -
7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1. How many page faults could occur for the following page replacement algorithms, assuming 3 frames :-
a) FIFO b) Optimal c) LRU d) LFU

Ans: Thrashing is the condition when excessive paging operations are taking place. When multiple processes are competing for same resource, then excessive swapping takes place and page faults will occur frequently. It results in poor system performance.

a) FIFO.

f1	7	7	7	2		2	2	4	4	4	0			0	0	-		7	7	7
f2		0	0	0	-	3	3	3	2	2	2		-	1	1		-	1	0	0
f3			1	1		1	0	0	0	3	3	-		3	2			2	2	1

No. of page faults = 15

b) Optimal.

f1	7	7	7	2		2		2	-		2		-	2	-			7		
f2		0	0	0	-	0	-	4			0			0		-		0	-	
f3			1	1		3		3		-	3	-		1			-	1		=

No. of page faults: 9

c) LRU

f1	7	7	7	2		2		4	4	4	0			1		1	-	1		-
f2		0	0	0	-	0	-	0	0	3	3	-		3		0		0	-	
f3			1	1		3		3	2	2	2		-	2	-	2		7		

No. of page fault = 12

d) LFU : 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

Frequency: ✗ 1 ✗ ✗ 2 ✗ 3 ✗ 1 1 4 2 ✗ ✗ ✗ 5 ✗ ✗ 5

f1	7	7	7	2		2		4	4	3		-		3	3		3	3		3
f2		0	0	0	-	0	-	0	0	0	-			0	0	-	0	0	-	0
f3			1	1		3		3	2	2			-	1	2		1	7		1

No. of page fault = 13.

2073 Bhadra.

Q. Differentiate compaction and coalescing technique. How logical address is mapped to physical address by Paging Technique? Explain with suitable example.

Ans.

Compaction	Coalescing
1. The process of combining all the holes into a big one by moving all the processes downward as far as possible is compaction.	The process of merging two adjacent holes to form a single larger hole is called coalescing.
2. It requires more CPU time.	It requires lesser CPU time.

PAGE NO.: _____
DATE:

Physical memory is divided into fixed size blocks known as page frames.
Logical address is divided into fixed size blocks known as pages. Size of page & page frames are same. During execution pages are loaded to frames.
For mapping of logical address to physical address.
Let, page number = p
frame number = f
page offset = d
Logical address (L) = p+d

- Page number is used to index into page number and fetch corresponding frame number (f).
- The physical address is obtained by combining f and d.
- For m bit processor, logical address will be m bits long
  Page size = 2^n bytes
  Then, n bits will specify offset and m-n bits specify page number.

Example: For logical address of 16 bits & page size 4kB.
Size of logical address space = 2^{16}
Page size = 4kB = 2^{12}
i.e. 12 bit offset
4 bit page number.

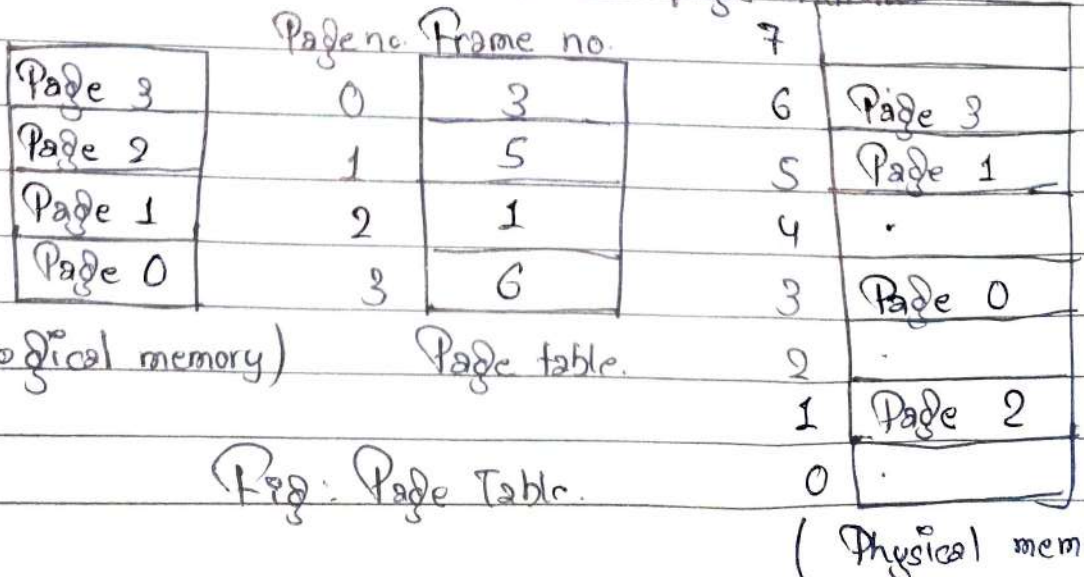


Fig: Page Table.

Q. With an example, show that FIFO page replacement algorithm suffers from Belady's anomaly

Ans. Belady's anomaly is the phenomenon in which increasing the number of page frame results in an increase in the number of page faults.

Let us consider a page sequence:
3, 2, 1, 0, 3, 2, 4, 3, 2, 1, 0, 4, 2, 3, 2, 1, 0, 4

When no. of frames = 3

f1	3	3	3	0	0	0	4			4	4	4	2	2	2	2	0	0
f2		2	2	2	3	3	3	-		1	1	1	1	3	-	3	3	4
f3			1	1	1	2	2		-	2	0	0	0	0		1	1	1

No. of page faults = 14

When no. of frames = 4

f1	3	3	3	3	-		4	4	4	4	0	0		0	0	1	1	1
f2		2	2	2		-	2	3	3	3	3	4		4	4	4	0	0
f3			1	1			1	1	2	2	2	2	-	3	3	3	3	4
f4				0			0	0	0	1	1	1		1	2	2	2	2

No. of page faults = 15

So, as no. of frame was increased, faults were also increased.

PAGE NO.:
DATE:

2073 Magh

Q. Under what circumstances do page fault occur?
Page reference string:
1, 2, 3, 4, 2, 1, 5, 6, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6
i) LRU
ii) FIFO
iii) Optimal replacement. , No. of frames = 3

i) LRU:

f1	1	1	1	4		4	5	5	5	3	3	3	-	3	3		-	3
f2		2	2	2	-	2	2	6	6	6	7	7		2	2	-		2
f3			3	3		1	1	1	2	2	2	6		6	1			6

No. of page fault = 14

ii) FIFO:

f1	1	1	1	4		4	4	6	6	6	7	7		7	1		1	1
f2		2	2	2	-	1	1	1	2	2	2	6		6	6		3	3
f3			3	3		3	5	5	5	3	3	3	-	2	2	-	2	6

No. of page fault = 15

iii) Optimal Replacement

f1	1	1	1	1		-	1	1		3	3		-	3	3		-	6
f2		2	2	2	-		2	2	-	2	7			2	2	-		2
f3			3	4			5	6		6	6	-		6	1			3

No. of Page fault = 11.

A page fault occurs when an access to

a page that has not been brought into main memory takes place. The OS verifies the memory access, aborting the program if it is invalid. If it is invalid, a free frame is located and I/O is requested to read the needed page into the free frame. Upon completion of I/O, the process table and page table are updated and instruction is restarted.

2072 Ashwin.

Q. Define page fault and demand paging. Consider a paged memory system with eight pages of 8kB page size each and 16 page frames in memory. Using the given page table, compute the physical address for logical address 18325.

7	10
6	4
5	0
4	7
3	13
2	11
1	14
0	5

Ans: A page fault is a type of exception raised by computer hardware when a running program accesses a memory page that is not currently mapped by the memory management unit (MMU) into the virtual address space of a process.

PAGE NO.: _____
DATE:

Demand paging is a method of virtual ~~mem~~ memory management. In a system that uses demand paging, the OS copies a disk page into physical memory only if an attempt is made to access it and that page is not already in memory.

Here,

Logical address (m) = 16

Total pages each = 8 = 2^3

Page number (p) = 3

Size of page = 8kB = 2^{13}

$\therefore$ n = 13

Offset bit = 13 bits

Page bit = 3 bit

No. of bits per frame = 4 bits

Logical address = 18325

= $\underbrace{0100}_{p}\ \underbrace{0111\ \ 1001\ \ 0101}_{d}$ [in Binary]

p = 010 = 2

From page table, the equivalent frame number for the given page is f = 11 = 1011

$\therefore$ Physical address (PA) = $\underbrace{1011}_{f}\ \underbrace{00111\ \ 1001\ \ 0101}_{d}$

= 92053 Ans

2072 Magh

Q. Consider the following page reperence string
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.

⇒ 2073 Magh.

2071 Bhadra

Q. What is the role of TLB?

Ans: A translation lookaside buffer (TLB) is a memory cache that is used to reduce the time taken to access a user memory location. The TLB stores the recent translations of virtual memory to physical memory.

2071 Magh.

Q. What is residence monitor? Consider logical address spaces of eight pages of 1024 words, each mapped onto a physical memory of 32 frames then.

a.) How many bits are in logical address?

b.) How many bits are in physical address?

Ans: Residence monitor is a type of system software program that was used in many early computers from 1950s to 1970s. It can be considered a percursor to operating system. The name is derived from a program which is always present in the computer's memory thus being "resident". Because memory was very limited on these systems the resident monitor was often little more than a stub which would gain control at the end of a job and load a non-resident portion to perform job cleanup and setup task.

Here,

No. of page bits : 3 bits

No. of offset bits : 10 bits

No. of bits for frame : 5 bits

PAGE NO.:
DATE:

∴ Bits in physical address = (10+5) bits = 15 bits
Bits in logical address = (10+3) bits = 13 bits

2070 Bhadra

Q. Sequence.

2 3 5 4 2 5 7 3 8 7

i) FIFO
ii) LRU
iii) Optimal, No of frames = 3

i) FIFO:

f1	2	2	2	4	4		4	3	3	
f2		3	3	3	2		2	2	8	
f3			5	5	5	-	7	7	7	-

No. of page hit = 2
No. of page fault = 8.

ii) LRU.

f1	2	2	2	4	4		7	7	7	-
f2		3	3	3	2		2	3	3	
f3			5	5	5	-	5	5	8	

No. of page hit = 2
No of page fault = 8

(iii) Optimal

f1	2	2	2	2	-		2	3	3		
f2		3	3	4			7	7	7	-	
f3			5	5		-	5	5	8		

No. of page hit = 8
No. of page fault = 7.

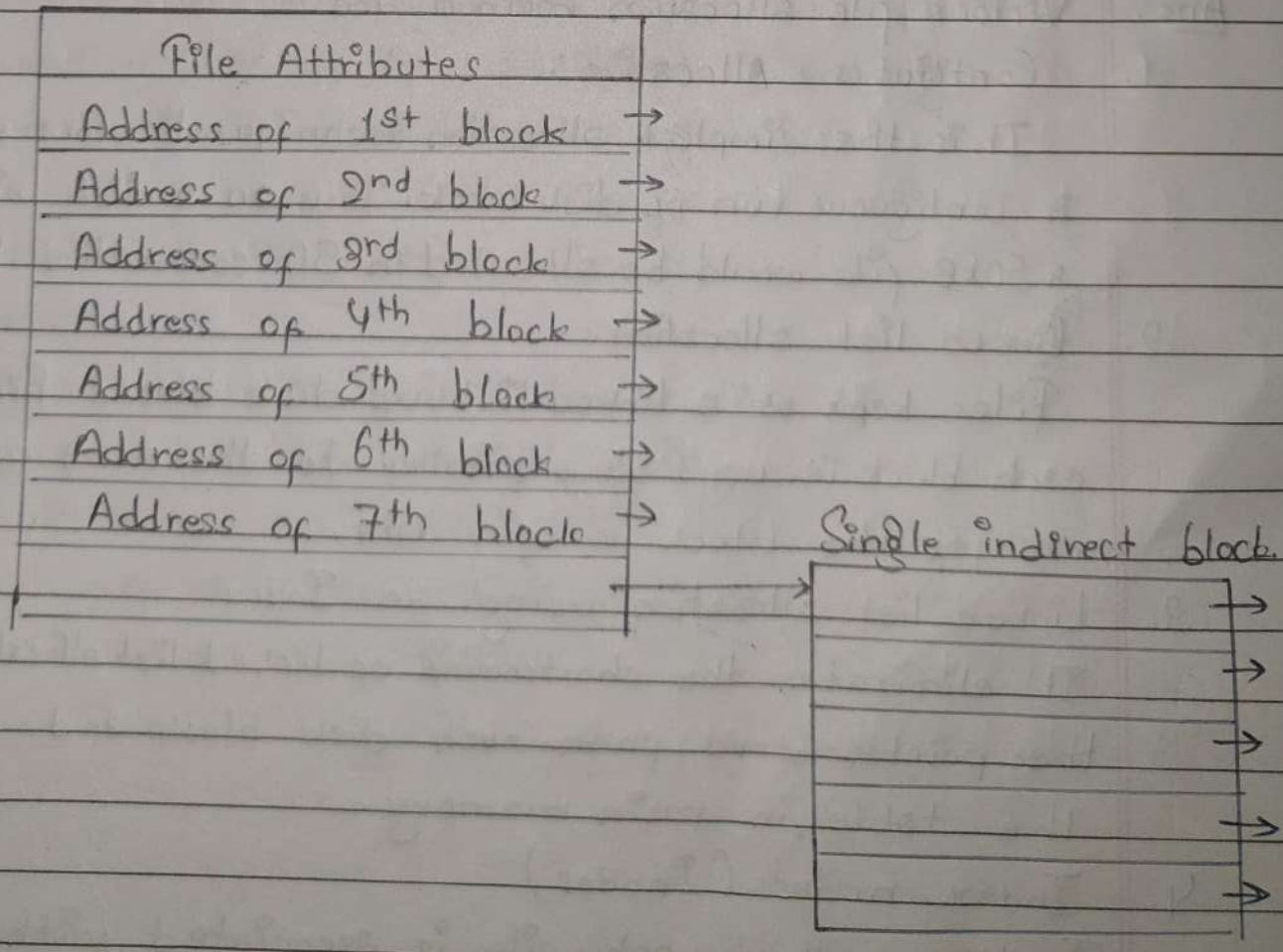
Chapter 5: File Systems.

2075 Bhadra:

Explain inode approach of file system implementation with its advantages and disadvantages.

Ans: Inode approach is an allocation method in which each file is associated with data structures known as i-node (index node). I-node of specific files contains around 40 seperate pieces of information. Some of them are:

- UID (User-Id) and gid (Group Id) of file.
- File type
- File creation, access and modification time.
- Inode modification time
- No. of links to the file
- Size of file.
- Disk address, specifying or leading to the actual disk location for disk blocks that make up file.



Advantages:

- At any instant, only i-nodes of opened files need to be in main memory, there by occupying much smaller space in main memory
- Larger files can be accessed efficiently.

Disadvantages:

- As inode information is kept seperately from data, access of data often requires a long seek when file is initially accessed.
- Inodes of files in a common directory if not kept together leads to low performance when searching directories.

2074 Bhadra:

Q. Discuss various file allocation and access method. Compare their advantages and disadvantages.

Ans: Various file allocation method are:

1. Contiguous Allocation:
   It is the simplest allocation technique to store each file as a contiguous run of disk block. Thus on a disk with 1kB block, a 50kB file would be allocated in 50 consecutive blocks.
2. Linked list allocation:
   Files kept as a linked list of disk blocks. First word of each block is used as a pointer to the next one. The rest of the block is for data.
3. Linked list allocation using an Index
   It eliminates the shortcoming of linked list allocation by taking the pointer word from each disk block and putting it in the table in main memory.
4. Index nodes (Inodes)
   In this method, each file is associated with data structure known as inode or index node.

Advantages & disadvantages:

i) Contiguous allocation

Advantage:
- Sequential and direct access are supported
- Simple to implement.

Disadvantage
- Suffers internal as well as external fragmentation.
- Inefficient memory utilization.
- Difficult to increase the file size.

ii) Linked list allocation:

Advantage:
- Flexible in terms of file size.
- No problem regarding the contiguous chunks of memory.

Disadvantage
- Random access is slow
- Comparatively slower than contiguous allocation
- Random or direct access is not supported.

iii) Linked list allocation using an index.

Advantage:
- Random access is easier
- Large file can be accessed easily

Disadvantage:
- Entire mei table must be in memory all the time to make it work.

PAGE NO.:
DATE:

iv.) Index nodes (Inodes)

Advantages:

- The scheme supports random access of the file.
- The scheme supports fast access to the file blocks
- The scheme is free from the problem of external fragmentation.

Disadvantages:

- The pointer head is relatively greater than the linked allocation of file.
- Index allocation suffers from the wasted space.

File access methods are:

1. Sequential Access:

When the information in the file is accessed in the sequential order i.e. one record after another it is called sequential access.

Advantages:

I. Easiest file access method

Disadvantages

I Lengthy & Slow process

2. Direct Access:

In direct access, bytes or ~~be~~ records can be accessed in any order. Access is based on a key then a position.

Advantage:

I. Faster than sequential access.

Chapter 5: 2073 Bhadra.

Qno. 6 (a)

In what ways is file system management similar to virtual memory management? When which file organization technique is most appropriate for tape storage?

Ans: File systems and virtual memory perform similar functions in different spheres virtual memory creates apparent contiguous memory regions from the combination of physical memory frames, backing mass storage and translation pages. Thus, user perceived memory can both grow without affecting other processes on the same system, and within reason grow beyond the capacity of physical memory.

A file system translates discontiguous collections of mass storage space into virtually contiguous files, a significant simplification. File system thus allows multiple files ~~reading~~ residing on the same storage device to grow without requiring data migration. From user perspective there does not appear to be any segmentation.

The best file organization technique for tape storage is sequential access because

- Data is accessed one record after another in an order.
- Read command cause a pointer to be ahead ~~of~~ by one.

Qno. 6(b)

List the file system performance indicators with brief explanation.

Ans: The file system performance indicator are:

1. Block cache or buffer cache:

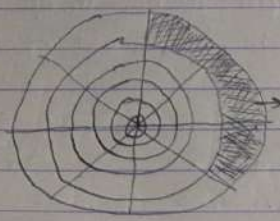
Cache is a collection of blocks that logically belongs

a block a disk but are kept in memory for performance reason. First check whether the required block is in cache or not. If it is, then request can be satisfied without disk access. If a block is not in the cache, it is free read into a cache, the copied to whenever it is needed.

2. Block Read Ahead:
- Get block into cache before they are needed to increase hit rate.
- A file system ask to produce block k of a file, it doest that, but when it is finished, it makes a sneaky checks in the cache to see if block k+1 is there or not. If it is not there, it brings block k+1 to memory in the hope that when it is needed, it will have already present in memory.

3. Reduce Disk Arm Motion
- Keep the block that are likely to be accessed in a sequence close to each other, preferably in same cylinder.
- Another performance bottle neck is system that use i-nodes requires reading over a short file requires two disk access.

a) One for inode
b) One for block



→ Inodes are located near the starting edge of disk.

In the figure, it is shown that all the inodes are near the begining of disk, so average distance between an inode and its block will be about half the no. of cylinders, requiring long seek.

Q. Write short notes on:
UNIX File System

UNIX file system is a file system supported by UNIX and UNIX-like OS. It is a distant descendant of the original file system used by version 7 UNIX.

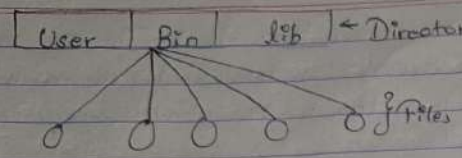
A UNIX file system volume is composed of the following parts:
- A few blocks at the begining of the partition reserved for boot blocks
- A superblock, containing a magic number identifying this as a UFS file system, and some other vital number describing filesystem geometry, statistics etc
- A collection of cylinder groups.

2073 Magh
What is File Attribute? Write the difference between single level directory system and Hierarchial directory system. Explain how OS manages free block of secondary storage?

Ans: File attributes are ~~secondary storage~~ settings associated with computer files that grant or deny certain rights to how a user or the OS can access the file.

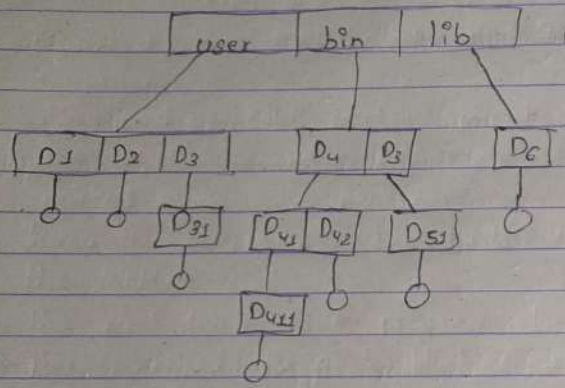
Single level directory system.

It is simplest directory structure. There is only one directory that holds all the files. Sometimes it is also called root directory.



Hierarchial directory system

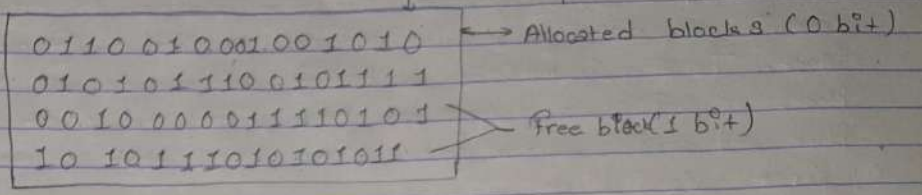
Also known as tree of directory or tree-structured directory. It allows user to have sub directories under their directories, thus making the file system more logical and organised for user.



OS manages free blocks of secondary storage by following methods

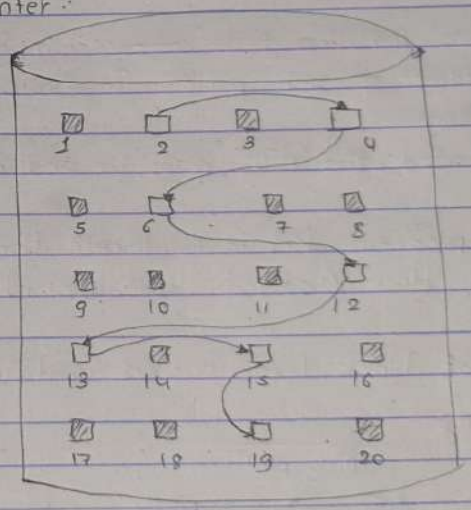
Bit vector:

Bit vector known as bit map is widely used to keep tracks of free blocks on a disk. To track all the free and used block on a disk with total n block, bit map having n bits is required. Each bit is bit map represented disk block, where a '0' in it represents an allocated block and a '1' in a bit represent a free block.



2. Linked list:

Linked list entry of free space management create a link list of all free blocks on the disk. A pointer to the first free block is kept in a special location on the disk and is cached in memory. The first free block contains the pointer.



3. Grouping

Grouping is a modification to the free-list approach in the sense instead of having a pointer in each free block to the next free block.

4 Counting

When a contiguous or clustering approach is used, creating or deleting a file allocates multiple contiguous block. Therefore instead of having address of all free blocks as in grouping, we can have pointer to first free block and count of contiguous free block that follow the first free block.

2072 Ashwin

Q What is file system layout?

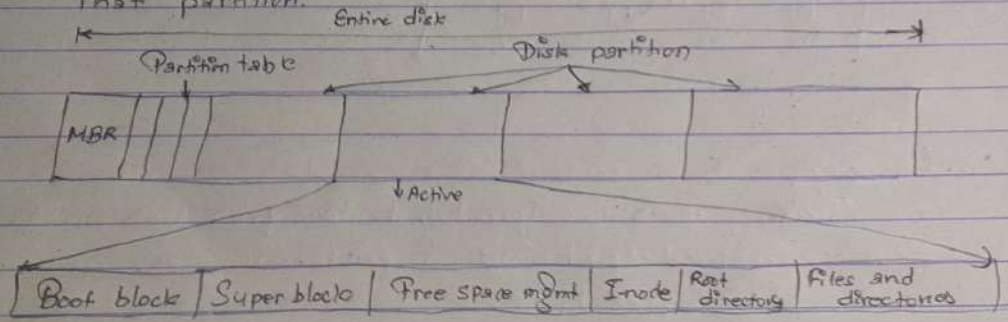
Ans. Files are stored on a disk. Sector 0 of the disk is called the MBR (Master Boot Record) and is to boot the computer.

The end of MBR contains partition table which gives the starting & ending address of each partition in table is marked as an active partition.

When the computer is booted, the BIOS reads in and executes the MBR.

The first thing MBR program does is to locate the active partition, reads its first block (called boot sector) and executes it.

The program in the boot block loads the OS contained in that partition.



2071 Magh

Q. What is file?

Ans: File is a collection of related information defined by its creator. The abstraction used by the kernel to represent and organize the system non-volatile storage resources, including hard disks, floppy disks, CD-ROM and optical disks.

2070 Bhadra

Q. Explain the system layout in detail. What are the major differences between file system interferences and file system implementation

Ans: File system implementation deals with:

- How files and directories are stored?
- How disk space is managed?
- How to make everything work, efficiently and reliably?

The main objective of file system implementation:

> To describe the details of implementing local file system & directory structures.
> To describe the implementation of remote file systems
> To describe discuss block allocation & free-block algorithms & trade-o

Some of the allocation methods in file system implementation are

1. Contiguous allocation.
2. Linked allocation.
3. Indexed allocation.

File system interface provide application with various system calls and commands such as open, write, read, seek, etc. Since main memory is usually too small, the computer system must provide secondary storage to back up main memory. The file system provides the

mechanism for storage of and access to both data and programs residing on the disks. Under this, we describe following topics:

- Access method
- Directory structure
- File system mounting
- File sharing
- Protection.

Chapter 6 : I/O Management & Disk Scheduling

~~Q. Tota~~

2075 Baisakh

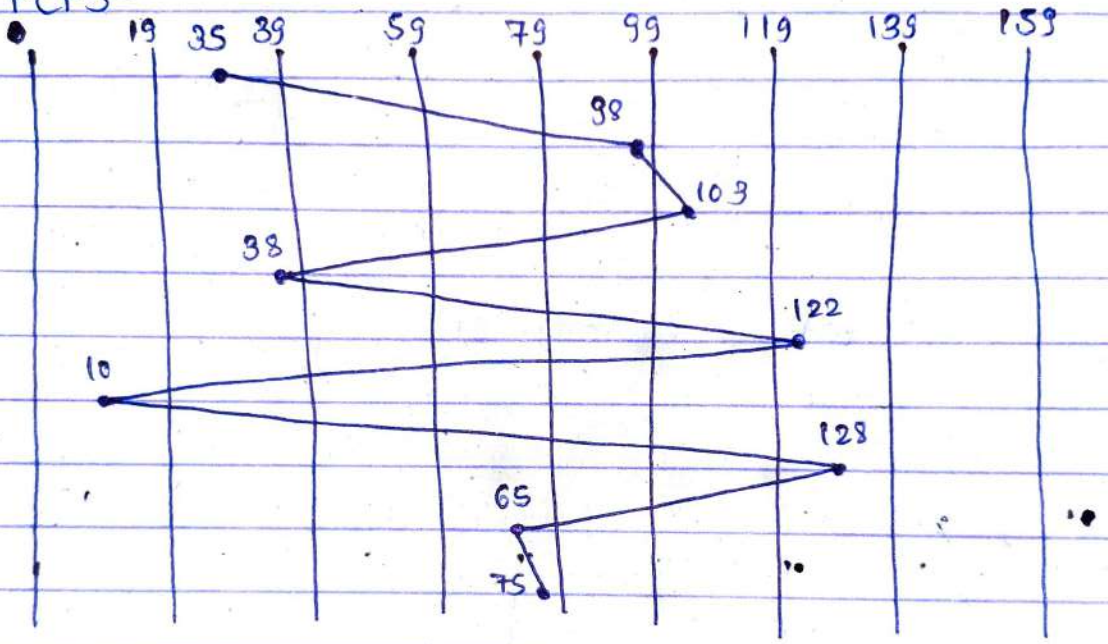
Q. : Total 150 cylinder.

Currently at 35 and previous at 120.

Queue: 98, 103, 38, 122, 10, 128, 65, 75

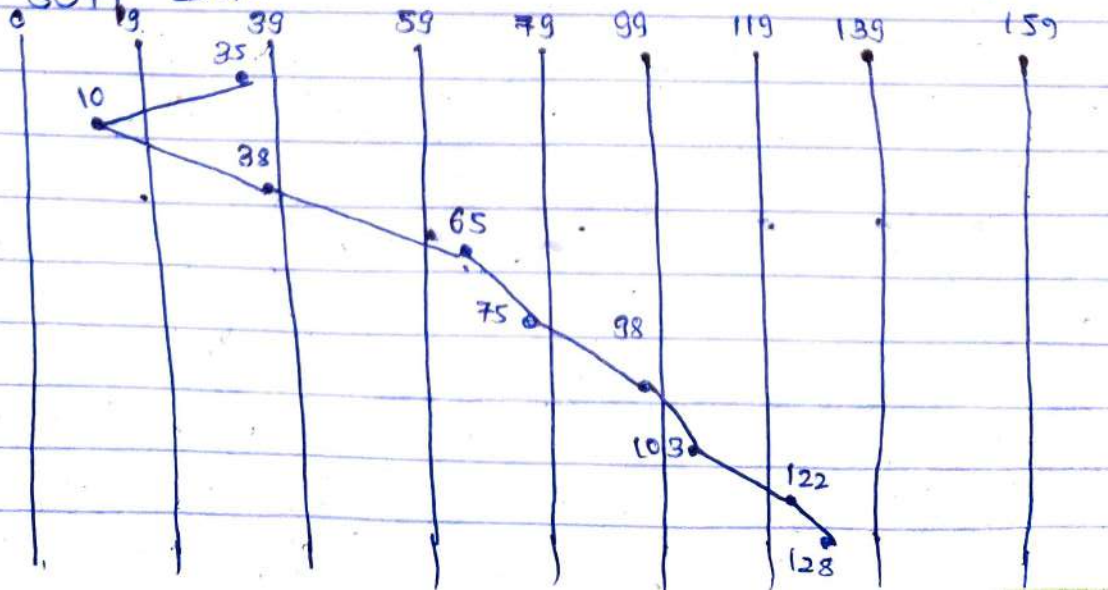
Calculate total head movement (in cylinder) for SSTF, SCAN, C-SAN, LOOK, FCFS.

i) FCFS.



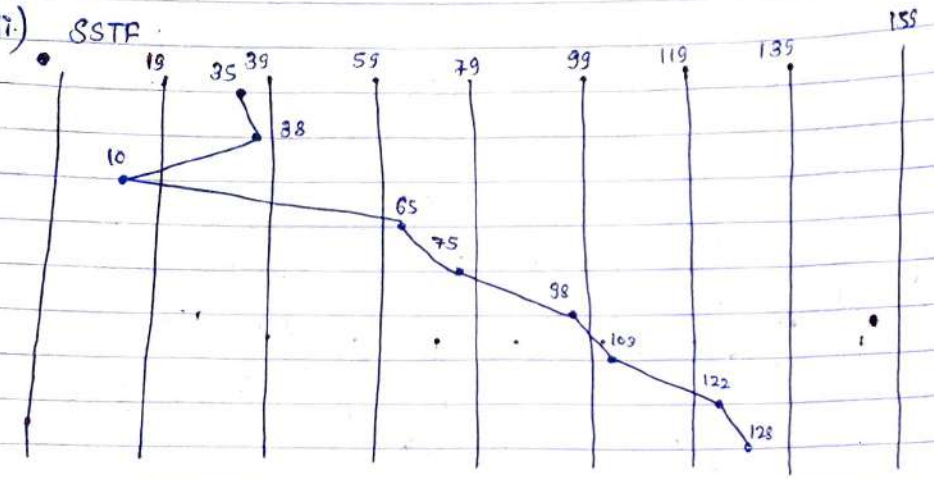
Total head movement : (98-35) + (103-98) + (103-38) + (122-38) + (122-10) + (128-10) + (128-65) + (75-65) = 520 cylinders

ii) ~~SSTF~~ LOOK



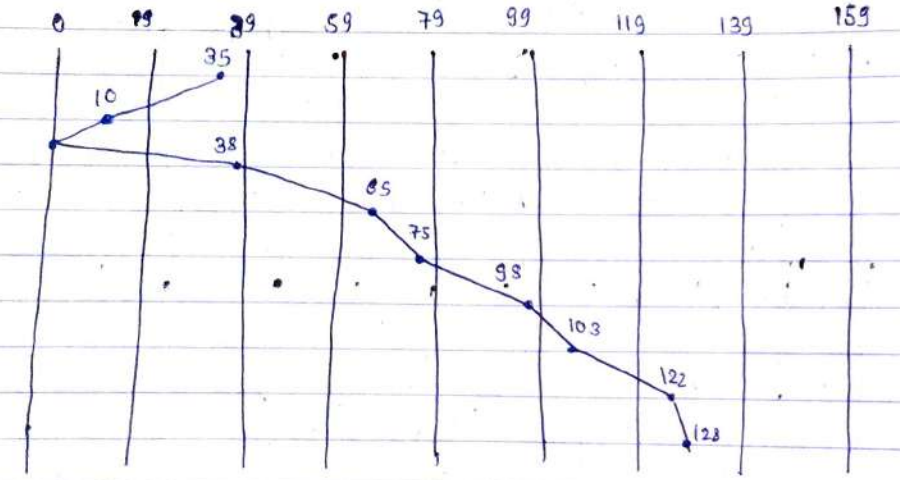
Total distance in cylinder. = 143 cylinders.

iii) SSTF



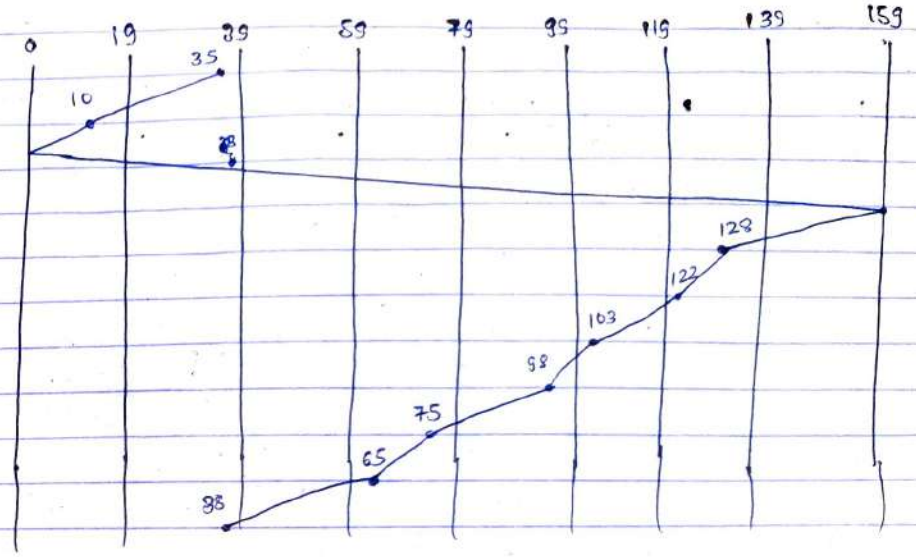
Total head movement in cylinders = 149 cylinder.

iv) ~~LOOK~~ SCAN



Total head movement = 169 cylinders

v) C-SCAN:

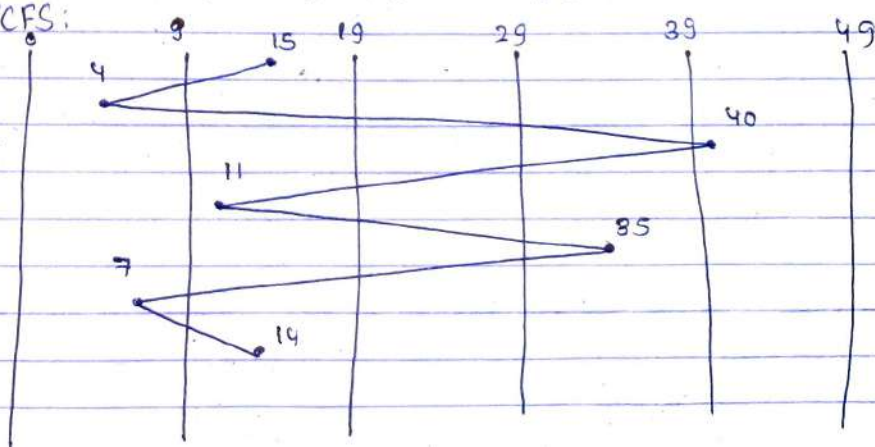


Total head movement = 315 cylinders //

2074 Bhadra

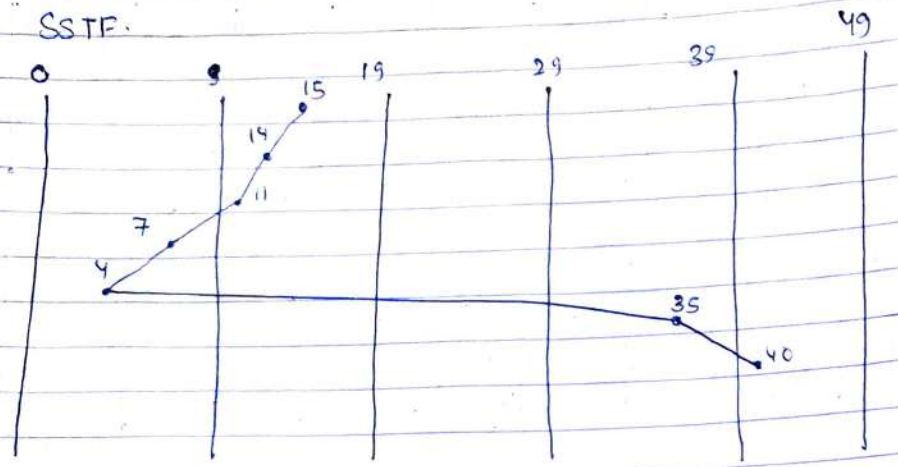
Q. Currently at cylinder 15
Total 50 cylinders
Queue: 4, 40, 11, 35, 7, 14.
FCFS, SSTF, SCAN, LOOK, C-SCAN, C-LOOK

i) FCFS:



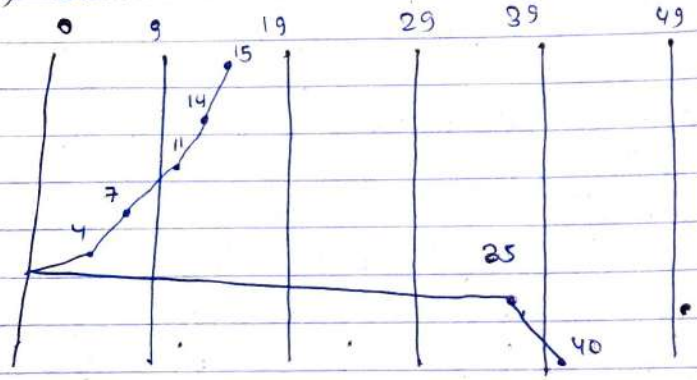
Total distance in cylinder = 135 cylinder

ii) SSTF.



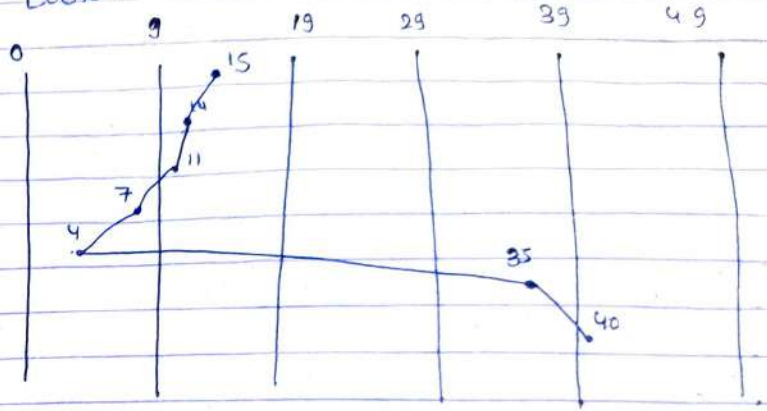
Total distance in cylinder : 47.

iii) SCAN:



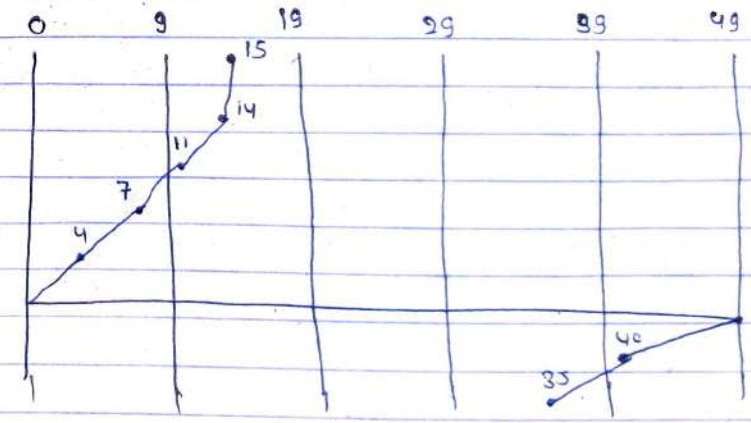
Total distance in cylinder = 55.

iv) LOOK



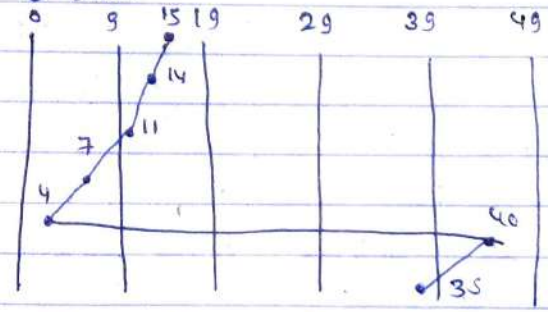
Total distance in cylinder = 47.

v) C-SCAN:



Total distance = 77

vi) C-LOOK

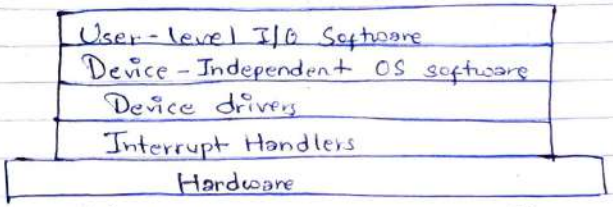


Total distance in cylinder : 51.

2073 Bhadra

Q. Briefly mention the structured I/O software with suitable diagram. Compare the throughput of SCAN and SSTF.

Ans. I/O software is organised in four layer. Each layer has a coell defined function to perform and a well define interface to the adjacent layers. The functionality and interface differ from system to system



- User-level I/O Software:

It provides simple interface to the user program to perform input and output.

- Device Independent OS software:

The basic function of the device-independent software is to perform the I/O functions that are common to all devices and to provide a uniform interface to the user-level software

- Device driver:

Device drivers are software modules that can be plugged into an OS to handle a particular device.

- Interrupt Handler.

An interrupt handler is a piece of software or more specifically a callback function in an OS or more specifically in a device driver, whose execution is triggered by the reception of an interrupt.

Throughput of SCAN and SSTF:

In 2074 Bhadra, the total distance in cylinder is 47 and 55 respectively for SSTF and SCAN respectively. So, SSTF has higher throughput than that for SCAN.

Q. Principle of I/O Software: [3.5]

The principles of I/O Software are:

a) Device Independence:

It should be possible to write programs that can access any I/O device without having to specify the device in advance.

b) Uniform naming:

The name of device should simply be a string or an integer and do not depend on the device in any way.

c) Error Handling

Errors should be handled as close to the hardware as possible.

d) Synchronous (blocking) V/s Asynchronous (Interrupt driven) transfe

It is upto OS to make the operation that are interrupt driven look blocking to the user program.

e) Buffering

f) Dedicated V/s Shared device

2073 Magh

Q. Total no. of cylinders = 200

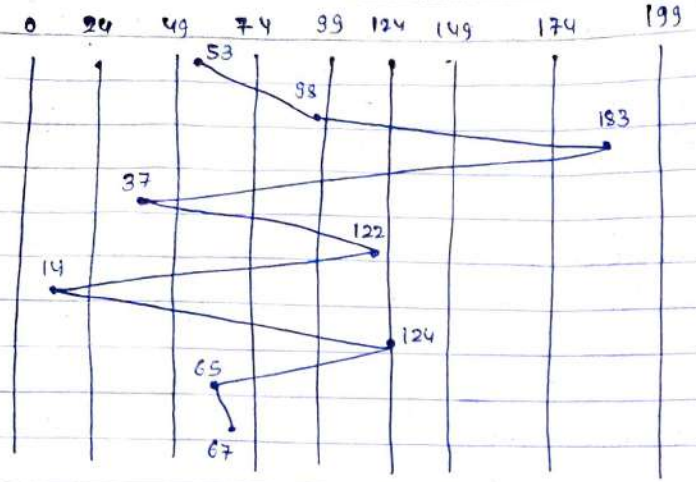
Currently at 53

Queue: 98, 183, 37, 122, 14, 124, 65, 67.

Calculate total head movement.

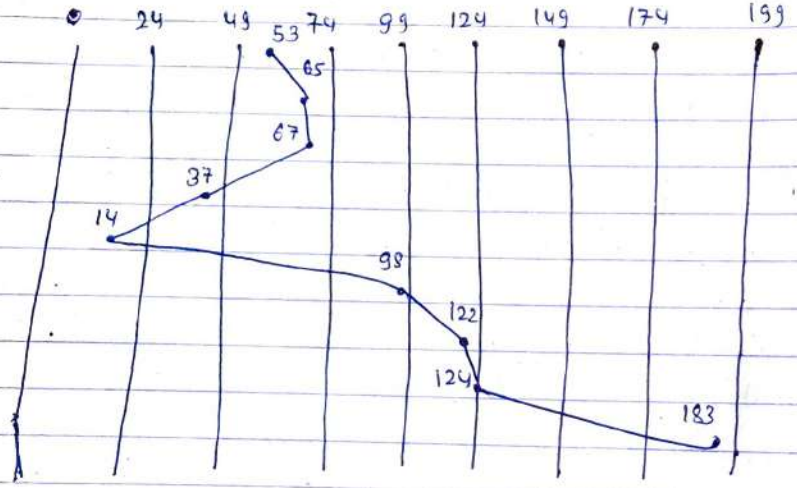
FCFS, SSF, SCAN.

Ans: i) FCFS



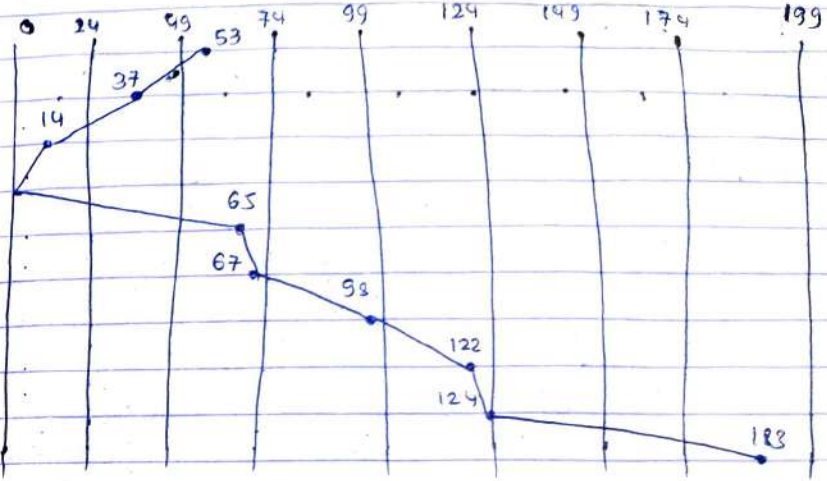
Total head movement = 640

ii) SSF



Total head movement = 236.

iii) SCAN



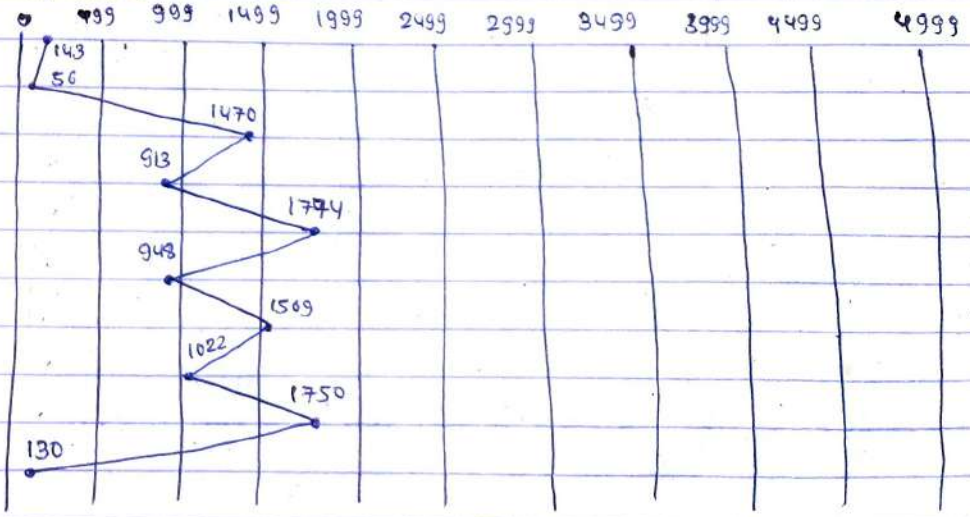
Total head movement = 236

2072 Magh

5000 cylinders, Currently at 143, previous request was 125

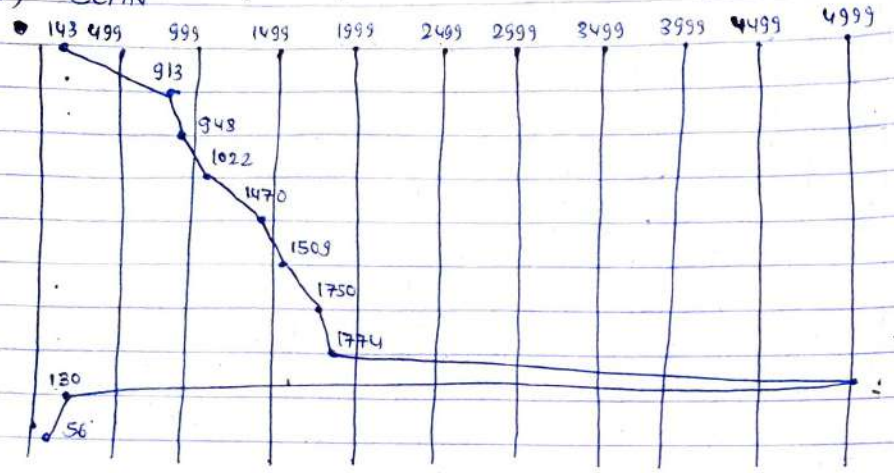
Queue: 56, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

Ans: i) FCFS



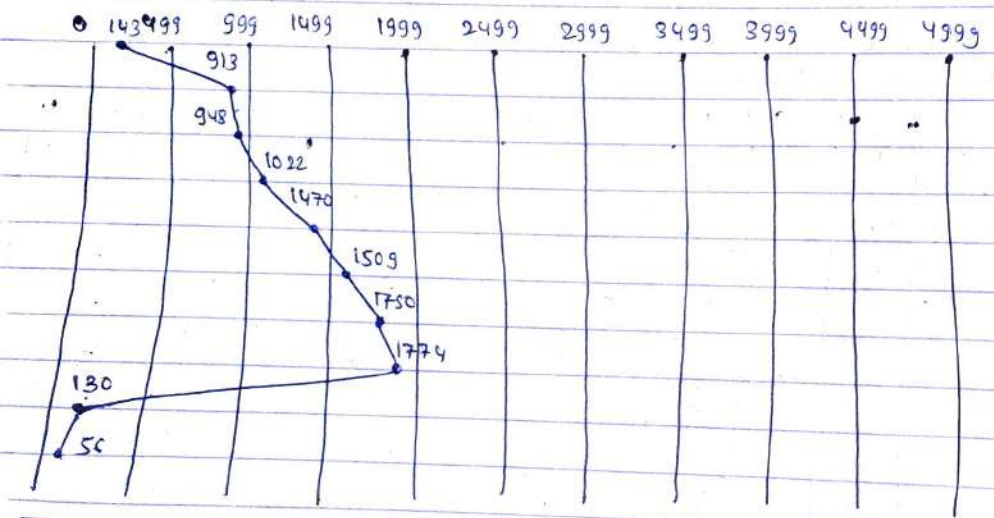
Total head movent = 7141 cylinders

ii.) SCAN



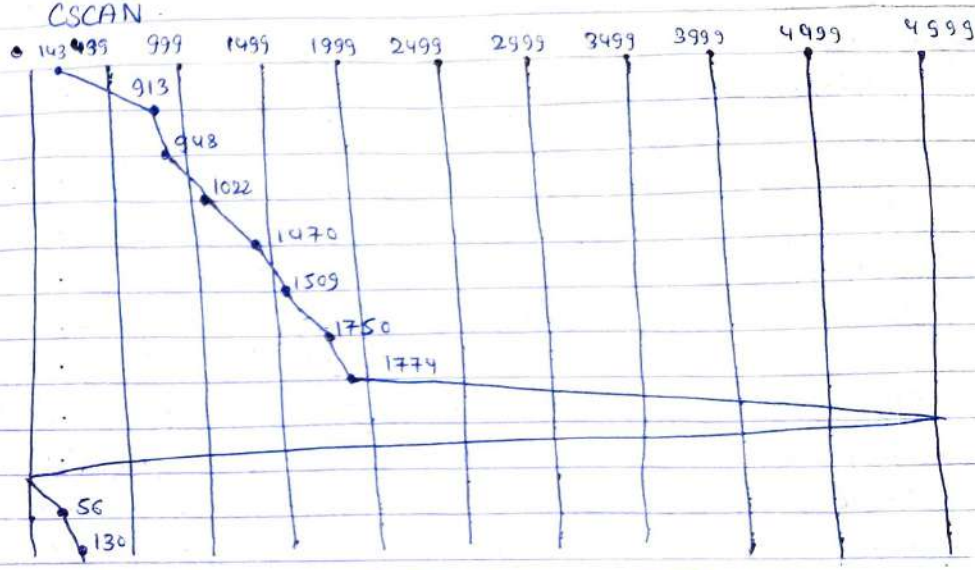
Total distance = 9779 Cylinders

iii.) LOOK.



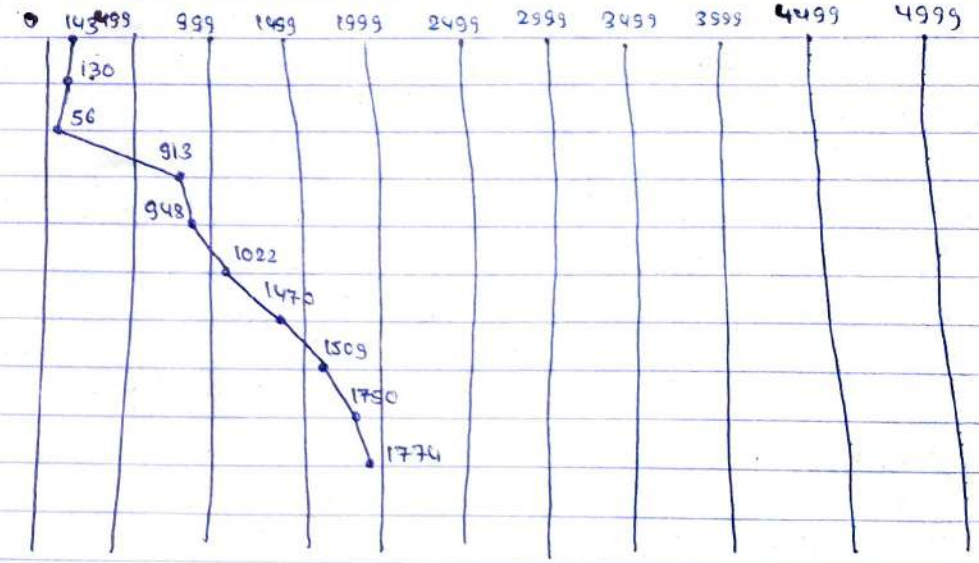
Total distance = 3349 cylinders.

iv.) CSCAN



Total distance = ~~9685 Cylinders~~ 9985 cylinders

v.) SSTF.



Total distance = 1805 cylinders

2072 Ashwin

Q. What is disc scheduling? Explain details about the device independent I/O software with example. [3+6]

Ans: Disk scheduling is done by operating system to schedule I/O requests arriving ~~from~~ for the disk. Disk scheduling is also known as I/O scheduling. First Come First Serve, Shortest Seek Time First, C-Scan, etc. are some of the disk scheduling algorithm.

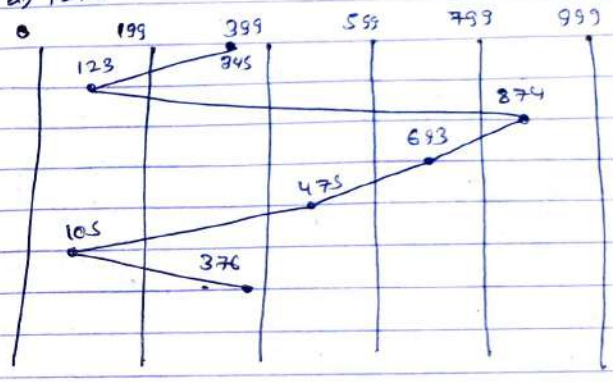
The basic function of the device-independent I/O software is to perform the I/O functions that are common to all devices and to provide a uniform interface to the user-level software. Though it is difficult to write completely device independent software but we can write some modules which are common among all devices. The functions performed by device independent I/O Software are:

- > Uniform Interfacing for device driver
- > Device naming
- > Device protection
- > Error reporting
- > Providing a device-independent block size
- > Storage allocation on block devices.
- > Allocating and releasing dedicated drivers.

2071 Bhadra

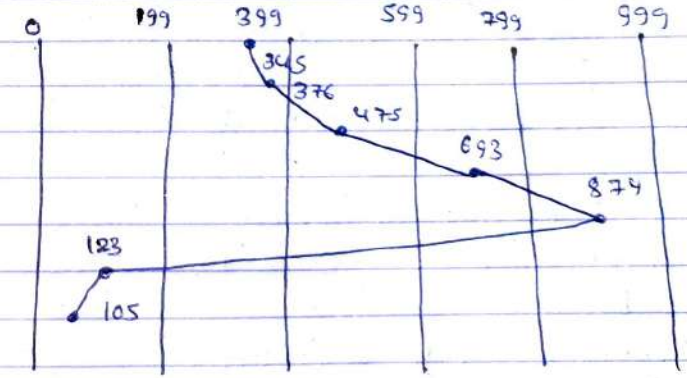
Q. Disk with 1000 cylinders
Last request at 345 and moving towards 0.
Queue: 123, 874, 693, 475, 105, 376
Calculate computation for following scheduling algorithms
a.) FIFO b.) SSTF c.) SCAN

a) FIFO



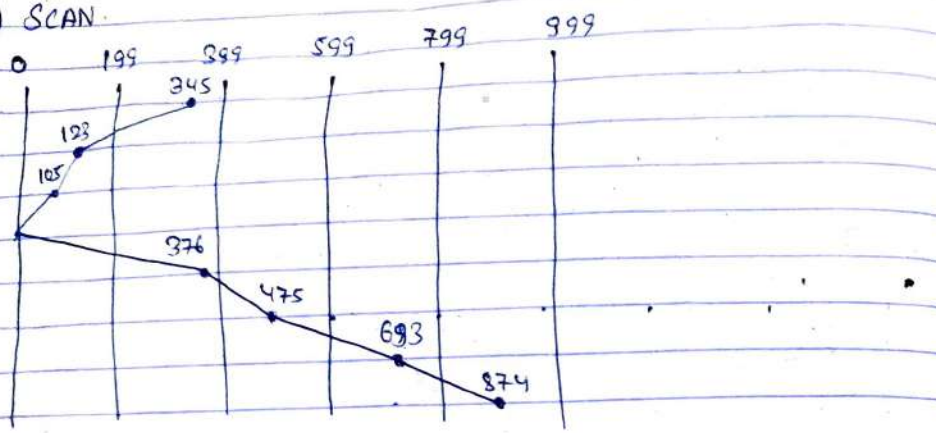
Total distance in cylinders = 2013

b) SSTF



Total distance in cylinders = 1298

c.) SCAN



Total distance in cylinders = 1219.(/)

2071 Magh

⇒ 2073 Magh.

2070 Bhadra

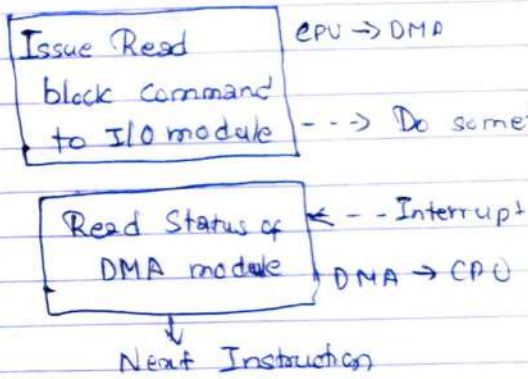
Q. What are the disadvantages of programmed I/O? Explain about DMA. What are the functions of device independent I/O Software. [2+2+4]

Ans: Device independent I/O Software [2072 Ashwin]

The disadvantages of programmed I/O are

- It is a time consuming process since it needlessly keeps the CPU busy.
- Performance of entire system is degraded.
- Processor, while waiting, must repeatedly interrogate the status of I/O module

Direct Memory access (DMA) is a method that allows an input/output device to send or receive data directly to or from the main memory, bypassing the CPU to speed up memory operations. This process is performed by DMA controller.



Direct Memory Access.

Chapter 7: Deadlock:

2070 Bhadra:

Q. What is deadlock? State the necessary conditions for deadlock to occur. Give reasons, why all conditions are necessary. [10]

Ans: Deadlock is the state where none of the processes can:

- Run
- Release resources
- Be awakened

Deadlock occurs when the process in a set are in simultaneous wait state for the release of resource held exclusively by one of the waiting process in the set.

The necessary conditions for deadlock are:

i) Mutual Exclusion:

At least one resource is held in a non-sharable mode that is only one process at a time can use the resource. If another process request that resource, the requesting process must be delayed until the resource has been released.

ii) Hold & Wait.

There must exist a process that is holding at least one resource and is waiting to acquire additional resources that are currently being held by other resources.

iii) No Preemption.

Resources cannot be preempted, that is, a resource can only be released voluntarily by the process holding it, after the process has completed the task.

Circular Wait.

There must exist a set $\{P_0, P_1, \ldots P_n\}$ of waiting process such that P_0 is waiting for a resource which is held by P_1, P_1 is waiting for a resource which is held by P_2 ... P_{n-1} is waiting for a resource which is held by P_n and P_n is waiting for a resource which is held by P_0

All the conditions are necessary for deadlock to occur. If any of the condition is not satisfied then, the resource may be free of any process i.e. the process release the resource and new ~~resource~~ process can be allocated that resource and deadlock won't occur.

2070 Magh.

Q. Consider a system consisting of m resource of the same type, being shared by n process. Resource can be requested and released by process only one at a time. Show that the system is deadlock free if the following two conditions hold:

a) The maximum need of each process is between 1 and m resources.

b) The sum of all maximum needs is less than m+n.

Ans: Let:

N = Summation of all Need$_i$; P_i = Process i

A = Summation of Allocation$_i$

M = Summation of all Max$_i$

Now,

DATE:

Given: Condition:

(a) The maximum need of each process is between 1 and m resources.

If the system is assumed to not be deadlock free and there exists a deadlock state, then $A \geq m$ because there is only one resource which can be requested/released one at a time.

(b) The sum of all maximum need is less than m+n.

In this condition, $M < m+n = N+A$

So, we get $N+m < m+n$

i.e. $N < n$.

It shows that at least one process i that $Need_i = 0$.

From condition (a), this process can release at least one resource, so there are n-1 process sharing n resources at this point, and both condition (a) and (b) still hold true. No process will wait permanently so there is no deadlock.

2071 Bhadra:

Q. Consider a system with 5 processes P_0 through P_4 and three resources types A, B, C. Resources type A has 7 instances, B has 2 and C has 6 instances. Suppose at to time we have the following state:

a. Is the given system in deadlock state?

b. Suppose P2 makes an additional request (0,0,1), what'll be the effect of this request to the system.

Process	Allocation			Request			Available		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	0	0	0			
P1	2	0	0	2	0	2	0	0	0
P2	3	0	3	0	0	0			
P3	2	1	1	1	0	0			
P4	0	0	2	0	0	2			

a

Solution,

The need matrix is :- [Need : Max.Req - Allocatin]

	A	B	C
P0	0	-1	0
P1	0	0	2
P2	-3	0	-3
P3	-1	-1	-1
P4	0	0	0

Finish [n]

(1)	(5)	(2)	(3)	(4)
P0	P1	P2	P3	P4
F	F	F	F	F
T	T	T	T	T

Work matrix is: (Available Initially)

0	0	0

1) Finish P0 = f and Need P0 < Work. So, P0 executes. When P0 completes execution, update work.

Work matrix: | 0 | 1 | 0 | [Work = Work + Allocation]

DATE:

2) For P1, Finish P1 = F but Need P1 ≤ Available is not satisfied. So, resource is not given to P1.

3) For P2, Finish P2 = F and Need P2 ≤ Available. So, P2 executes. When P2 completes execution, update work.

Work is:

3	1	3

4) For P3, Finish P3 = F and Need P3 ≤ Available. So, P3 executes. When P3 completes execution, update work.

Work is:

5	2	4

5) For P4, Finish P4 = F and Need P4 ≤ Available. So, P4 executes. When P4 completes execution, update work. Work is:

5	2	6.

6) For P1, Finish P1 = F and Need P1 ≤ Available. So, P1 executes. When P1 completes execution, update Work. Work is:

7	2	6.

(a) No, the system is not in deadlock state since all the process executes.

(b) Since, the request (0, 0, 1) is more than available (0, 0, 0). It can't be granted immediately.

2071 Magh.

Q. Explain the necessary condition of deadlock? How can a system detect deadlock and what does it do after detection?

Ans: 1st question: 2070 Bhadra.

The system can detect a deadlock as:

i) Deadlocks detection for single instance of each resource type:

For single instance of each resource type, deadlock can be detected by construction wait-for graph from resource allocation graph.

If cycle exist in wait-for-graph then deadlock exist otherwise deadlock doesn't exist.

E.g:

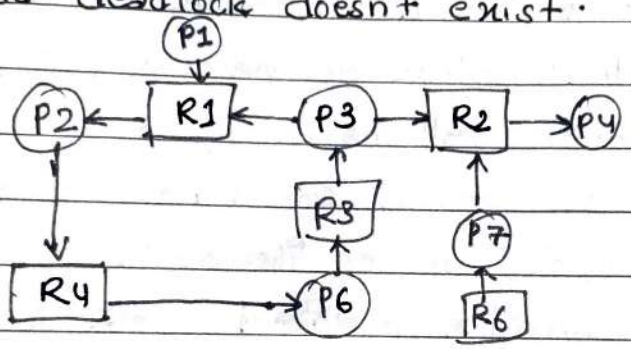


Fig: Resource allocation graph.

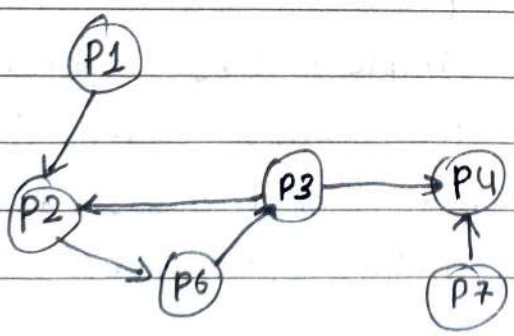


Fig: Wait-for graph.

Here, since cycle exists in wait-for graph. Deadlock exist.

ii) Deadlock detection for multiple instance of a resource type:

DATE:

If there are multiple instances of a resource type, the wait-for graph cannot detect deadlock. So, for this we use algorithm similar to Banker's algorithm to detect deadlock.

Once the deadlock has been detected in the system, different methods are applied to recover the system from deadlock and continue with processing. The methods are:

i.) Recovery through preemption:
The resource is temporarily taken away from its current owner and given to other process. It depends on the nature of resource

ii.) Recovery through roll back:
- A checkpoint is created for a process periodically.
- Creating checkpoint means its state is written to a file so that it can be ~~restarted later~~. restarted later.
- If a deadlock is found, the process is restarted

iii.) Recovery through killing process.
It is the crudest and simplest way to break deadlock. A process is killed in the deadlock state. The other process gets its resources. The process that will yield up no ill effect to entire system is chosen.

2072 Ashwin

Q. What is deadlock? Explain the essential condition of deadlock. How do you detect the deadlock. Explain with examples.

Ans: 1st + 2nd question: 2070 Bhadra.
3rd question: 2071 Magh

PAGE NO.:
DATE:

2072 Magh:

Q. What is deadlock avoidance and detection? Explain all possible deadlock prevention techniques.

Ans. Deadlock avoidance can be achieved by never allowing allocation of resources to a process if it will lead to a deadlock. It is done by being careful at the time of resource allocation. The system must be able to decide whether granting a resource is safe or not, any only make allocation when it is safe.

Deadlock detection is the process of detecting the deadlock in the system if it is not prevented or avoided.

The possible deadlock preventing techniques are:

1. Denying the mutual exclusion condition.
2. Denying the Hold and Wait condition.
   It can be done by allocating all the required resources to a process before the start of its execution this way hold and wait condition is eliminated but it will lead to low device utilization.
3. Denying the No-Preemption condition:
   Preempt resources from the process when resources are required by other high priority process.
4. Denying the circular wait condition:
   A process is entitled to only one resource at a time or at a moment. If it needs a second one it must release a second one.

PAGE NO.:
DATE:

2073 Bhadra

Q. What is the difference between deadlock and idefinite postponment? Consider a system with 5 concurrent process (P0, P1, P2, P3, P4) and 4 resource type (R0, R1, R2, R3). The no. of instances of each resource type in the system are (6, 4, 4, 2) respectively. Is the state safe? Show the execution of the process.

Allocation:

	R0	R1	R2	R3
P0	2	0	1	1
P1	1	1	0	0
P2	1	1	0	0
P3	1	0	1	0
P4	0	1	0	1

Maximum claim:

	R0	R1	R2	R3
P0	3	2	1	1
P1	1	2	0	2
P2	1	1	2	0
P3	3	2	1	0
P4	2	1	0	1

Ans:

Deadlock is a situation in which two or more competing actions are each waiting for the other to finish and thus neither ever does. If two processess are in deadlock, it is not possible for them to ever do any useful work - because they depend on one another, and neither will ever yield.

Indefinite postponement is to delay indefinitely the scheduling of a process while other process receive the system's attention. If a process is postponed indefinitely, it is at least theoretically possible for such process to continue and do some useful work at some time in the future.

Here,

Available = $(6-(2+1+1+1), 4-(1+1+1), 4-(1+1), 2-(1+1))$

= $(1, 1, 2, 0)$

The need matrix is:

	R0	R1	R2	R3
P0	1	2	0	0
P1	0	1	0	2
P2	0	0	2	0
P3	2	2	0	0
P4	2	0	0	0

Finish [n]

(4)	(5)	(1)	(2)	(3)
P0	P1	P2	P3	P4
F	F	F	F	F
T	T	T	T	T

Initially Work : Available.

1	1	2	0

1.) For P0, Finish P0 = F but Need ≤ Work is not satisfied. So, resource is not given to P0.

2.) For P1, Finish P1 = F but Need P1 ≤ Work is not satisfi So, resource is not given to P1.

3.) For P2, Finish P2 = F, and Need P2 ≤ Work. So, P2 executes and work is updated.

Work:

2	2	2	0

4.) For P3, Finish P3 = F, and Need P3 ≤ Work. So, P3 is executed and work is updated.

Work:

3	2	3	0

5.) For P4, Finish P4 = F and Need P4 ≤ Work. So, P4 is

executed and work is updated.

Work

5	2	3	0

3	3	3	1

6.) For P0, Finish P0 = F and Need P0 ≤ ~~Allocation~~ Work. So, P0 executes and work is updated

Work:

6	4	3	0

5	3	4	2

7) For P1, Finish P1 = F and Need P1 ≤ Work. So, P1 executes and work is updated.

Work:

6	4	4	2

∴ The system is in safe state because all the process executes.

Sequence of execution:

P2, P3, P4, P0, P1. //

~~2073 Magh~~:

Q. 2074 Bhadra

Q. A system has 2 process and 3 resources. Each process need maximum of two resources. Is, deadlock possible? Explain.

Ans. For deadlock to occur, the following four conditions must be satisfied.

i) Mutual Exclusion.

ii) Hold & Wait

iii) No preemption

iv) Circular wait

DATE :

Suppose, two resources are provided to a process and one to another.

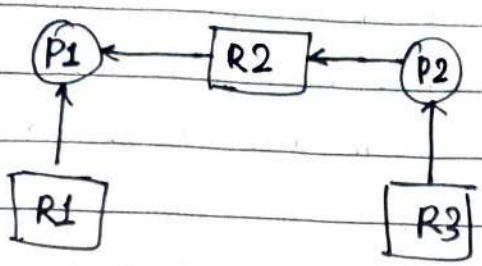


Fig: Resource Allocation Graph.

Now, since P1 has 2 resource as required, there is no hold and wait and also there is no circular wait. So, P1 can preempt resource R2 and P2 can use the resource. So, there is no deadlock possible.

2075 Bhadra

Q. Similar to 2073 Bhadra.

chapter8 Security.

2075 Bhadra.

6. Explain private and public key used in assymetric cryptography. What is the use of ACL?

Ans: Assymetric encryption use a mathematically related key pair for encryption and decryption; one is the public key and other is the private key.

If the public key is used for encryption; the related private key is used for decryption and vice versa. Only the user or computer that generates the key pair has the private key. The public key can be distributed to anyone who wants to send encrypted data to the holder of the private key. The two participants in the assymetric encryption workflow are the sender and the receiver. First, the sender obtains the receiver's public key. Then the plaintext is encrypted with the assymetric encryption algorithm using the recipient's public key, creating the ciphertext. The ciphertext is then sent to the receiver, who decrypts the ciphertext with this private key so that we can assess the sender's plaintext.

Use of ACL

1. ACL are filters that enable us to control which routing updates or packets are permitted or denied in or out of the network
2. They are specially used by network administrators

to filter traffic and to provide extra security for the network.

3. ACLs provide a powerful way to control traffic into and out of our network.

2074 Bhadra

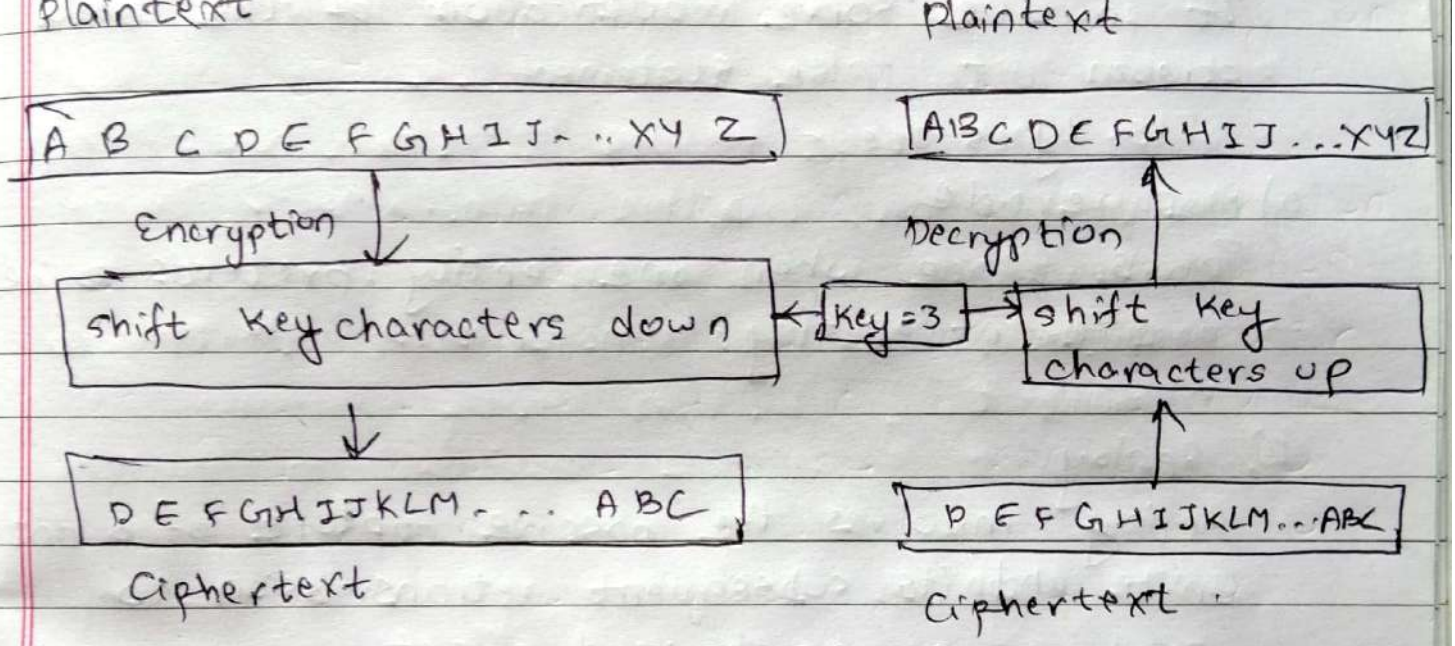
8. How authentication is an essential mechanism for maintaining security. Explain.

Ans:- Authentication is the process of determining whether someone or something is, in fact, who or what it declares itself to be. Authentication technology provides ~~technology~~ access control for systems by checking to see if a user's credentials match the credentials in a database of authorized users or in a data authentication server. The most widely used form of authentication is to require the user to type a login name and password. Password is easy to understand and implement. The second method for authenticating users is to check for some physical object they have rather than something they know. The third authentication method measures physical characteristics of the user that are hard to forge. These are called biometrics.

Authentication is important because it enables organizations to keep their networks secure by permitting only autheticated users to access its protected areas.

9a) Caesar Cipher

The caesar cipher is one of the earliest known and simplest cipherest. It is a type of substitution cipher in which each letter in 'plaintext' is shifted a certain number of places down the alphabet



Encryption

$C = (p + K) \bmod 26$

p = plaintext

C = ciphertext

K = shift

Decryption

$p = (C - K) \bmod 26$

2073 Bhadra

(ii) Types of security attack

1) Active attacks

An active attacks attempts to alter system resources or effect their operations. It involve some modification of the data stream of false statement

Its types are:-

a) Masquerade

Takes place when one entity pretends to be a different enitity.

b) Replay

Replay involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect

3. Modification of meassage

means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect.

4. Man in the middle attack

an attacker sits in the data flow of a communication, masquerading as the sender to the receiver, and vice versa.

5. Denial of service (DOS)

This violation involves preventing legitimate

use of the system. Disable network or overload it with messages.

2. Passive attack.
a) Obtaining Message content
b) Traffic analysis

2073 Magh

Q i) Protection Domain

→ A computer system contains many "objects" that need to be protected.

→ These objects can be hardware (eg. CPUs, memory segments, disk drives or printers) or they can be software (eg, processes, files, databases or semaphores)

→ Each object has a unique name by which it is referenced and a finite set of operations that processes are allowed to carry out on it.

→ a way is needed to prohibit processes from accessing objects that they are not authorized to access.

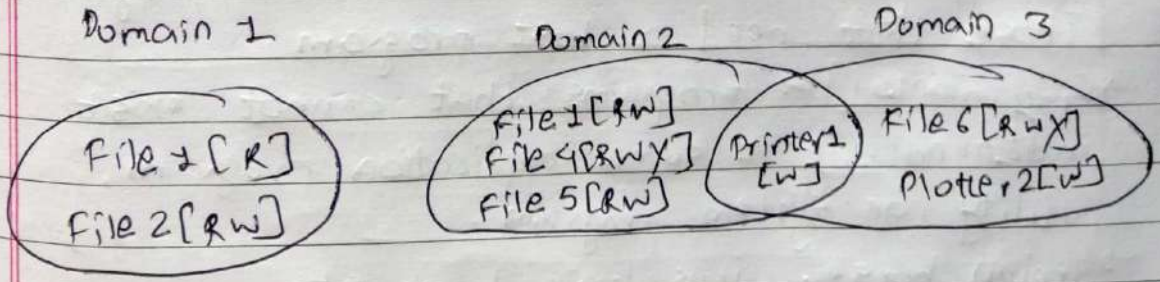


Fig: Three protection domains.

ii) Cryptography
→ It is the science and art of transforming message to make them secure and immune to attack
→ The purpose of cryptography is to take a message or file, called the plaintext, and encrypt it into ciphertext in such a way that only authorized people know how to convert it back to plaintext.
→ Original message before transformation => Plaintext
→ An encryption algorithm transforms => Plaintext to ciphertext
→ Decryption algorithm transforms => Ciphertext to Plaintext.

2072 Magh.

8. The use of internet is possible cause of a security breach. Describe the major threats by which a system connected to the internet is always prone to attack. Explain.

Ans.- The major threats ~~connected~~ are:-
Category 1: Based on need of Host program. Those that need a host program
→ Fragments of programs that cannot exist independently of some application program, utility, or system program
→ Trojan horse, virus, logic bomb, etc

Independent
→ Self contained programs that can be scheduled

and run by the operating system
→ Zombie, worms, etc.

Category 2: Based on replicative behavior
Replicative
→ Virus, worm, Zombies

Non-replicative
→ Trap droors, Trojan horse

2071 Magh

8c) Security Policy.

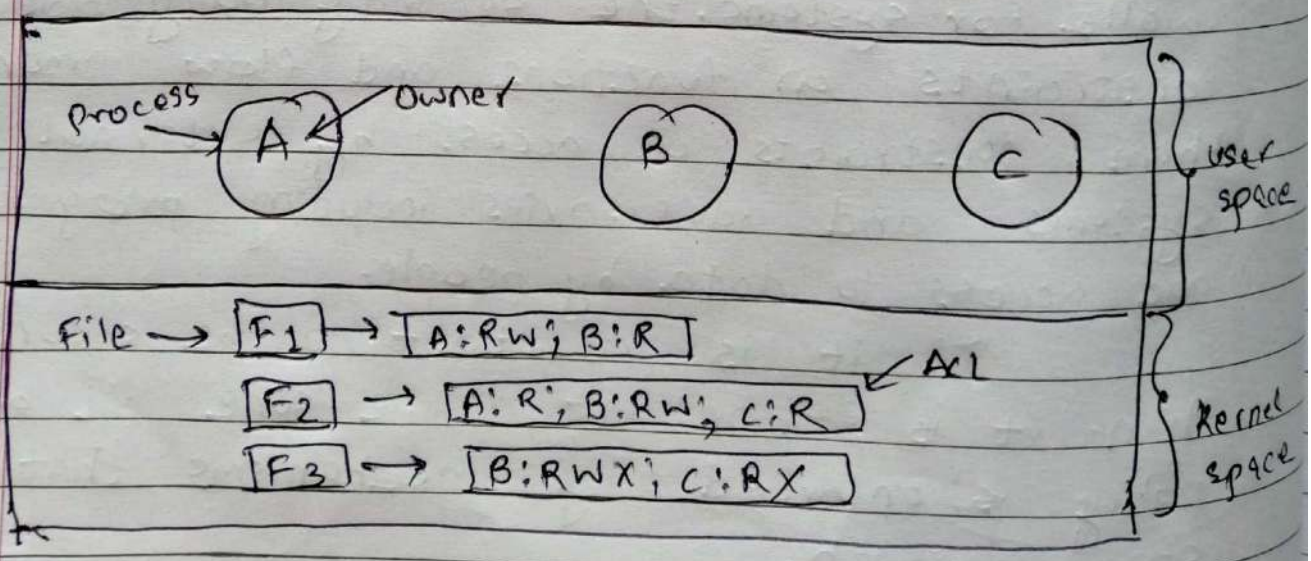
Security policy is a definition of what it means to be secure for a system, organization or other entity. For an organization, it addresses the constraints on behavior of its members as well as constraints imposed on adversaries by mechanisms such as doors, locks, keys and walls. For systems, the security pdicy addresses constraints on functions and flow among them, constraints on access by external systems and adversaries including programs and access to data by people.

If it is important to be secure, then it is important to be sure all of the policy is enforced by mechanisms that are strong enough.

2070 Bhadra.

8. Explain the ACL. How its mechanisms are implemented for security?

Ans: → In practice, actually storing the matrix of slide 34 is rarely done because it is large and sparse
→ Most domain have no access at all to most objects, so storing a very large, mostly empty, matrix is a waste of disk space.
→ More practical approach would be to store the matrix by row or column and then storing only the non empty elements.
→ The first technique consists of associating with each object an (ordered) list containing all the domains that may access the object, and how.
→ The list is called ACL.



2070 Magh

9a) Information security Model.

In formation security modes are models used to ~~avat~~ authenticate security policies as they are intended to provide a precise set of rules that a computer can follow to implement the fundamental security concepts, processess and procedures contained in a security policy. These models can be abstract or intuitive.

Some of the information security Models are,

1) State Machine Model
2) Bell-La Padula Model
3) Biba Model
4) Clark-Wilson Model
5) Non interference Model, etc

Chapter 9: System Administration

2075 Bhadra

Q. What is the significance of system administration? Describe the role & responsibilities of system administrator to keep the system updated & efficient. Explain with an example.

Ans: System administration refers to the management of one or more hardware & software systems. The task is performed by a system administrator who monitors system health, monitors and allocates system resources like disk space, performs backups, provides user access, manage user accounts, monitors system security and performs many other functions.

The roles & responsibilities of a system administrator to keep the system updated and efficient are:

1. User administration (setup and maintaining account)
2. Maintaining system.
3. Verify that peripherals are ~~coorkl~~ working properly.
4. Quickly arrange repair for hardware in occasion of hardware failure.
5. Monitor system performance.
6. Create file system
7. Install software.
8. Create a backup and recover policy.
9. Monitor network communication.
10. Update system as soon as new version of OS and applicatio software comes out.
11. Password and identity management.

2074 Bhadra.

Q. Write short notes on:
Administration tasks:

⇒ Roles and responsibilites of system administrator:
[2075 Bhadra]

2073 Bhadra.

Q. Short notes on:
Duities and responsibilites of system administrator.

⇒ [2075 Bhadra]

2073 Magh.

Q. Short notes on:
System administration

⇒ [2075 Bhadra]

2072 Ashwin

Q. What is system administration? How is a special user different from a general user? Explain.

Ans. General user is designed to provide basic permission for completing common daily tasks. It allows user to launch applications, create new documents, and modify basic system configurations. These operations affect only the user who is logged in on. They do not include system wide changes such as installation of new software.

On other hand special user has the capability of performing any operation or task on the system. This includes all of the permission that are granted to a standard user account plus the ability to make major OS changes, install new software, and create and modify other user accounts. It also has the permission for to set permission for other users on the system.

2072 Magh, 2070 Bhadra

Short notes on:
Role of system administrator

⇒ 2075 Bhadra.

2071 Bhadra

Duities & Responsibilites of system administrator.

⇒ 2075 Bhadra.

2071 Magh., 2070 Bhadra

Q. Write short notes on:
Shell scripts.

Ans. A shell script is a computer program designed to run by the Unix/Linux shell which could be one of the following:

- The Bourne Shell
- The C Shell
- The Korn Shell
- The GNU Bourne-Again Shell

A shell is a command-line interpreter and typical operations performed by shell scripts include file manipulation, program execution and printing text. Shell accept human readable commands from user and convert them into something which kernel can understand. The shell gets started when the user logs in or start the terminal.

PAGE NO.:
DATE:

The reason to write shell scripts are:

- To avoid repetitive work and automation
- Systems monitoring
- Adding new functionality to the shell
- For routine backups

Advantages:

- Quick start
- Writing shell scripts are much quicker.
- Interactive debugging.
- Command and syntax are exactly the same as those entered in the command line.

Disadvantages:

- Prone to costly errors
- Slow execution speed.
- Not well suited for large and complex task.
- Provide minimal data structure.