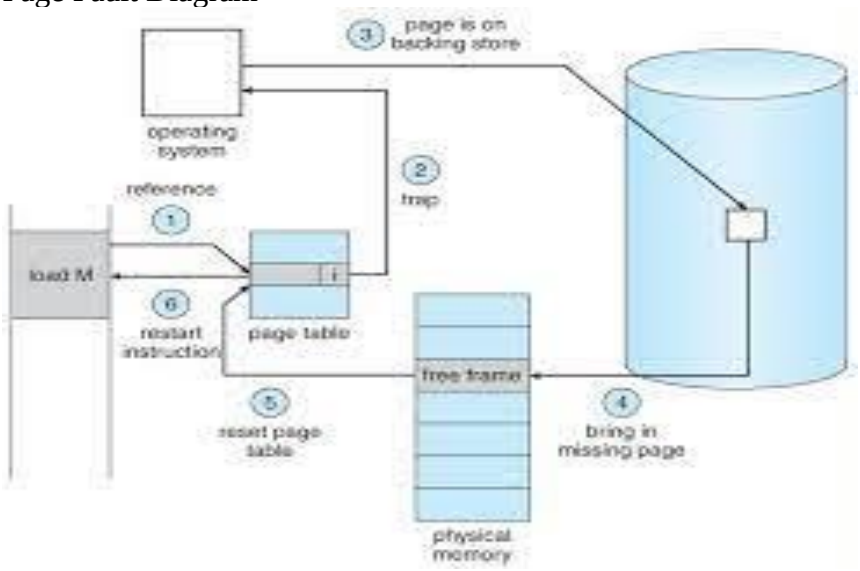
Solution

Internal Assessment Test - III – July 2021

Sub:	Operating Systems							Code:	18CS43
Date:	30/07/2021	Duration:	90mins	Max Marks:	50	Sem:	IV	Branch:	ISE

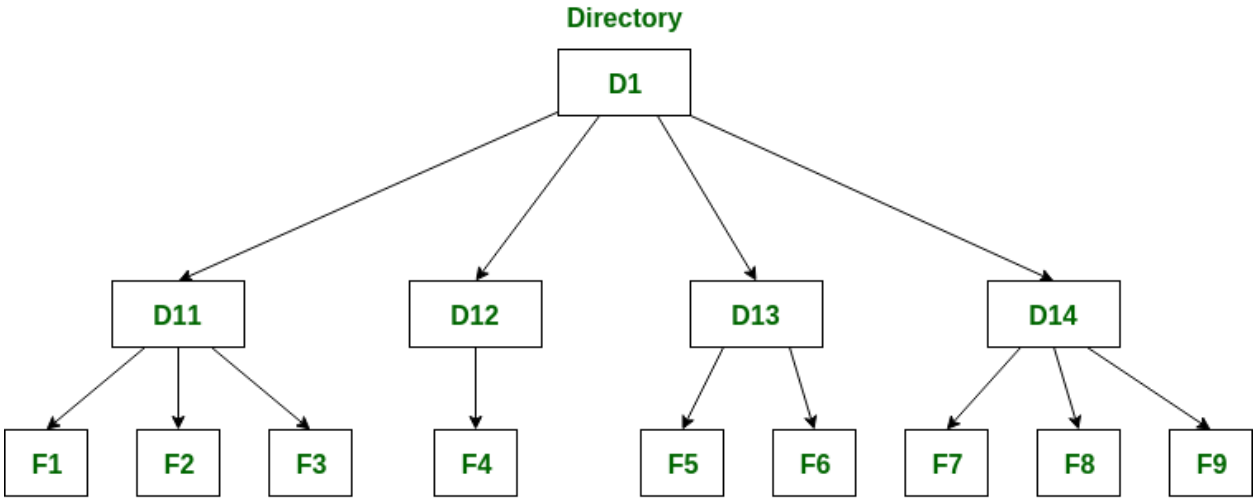
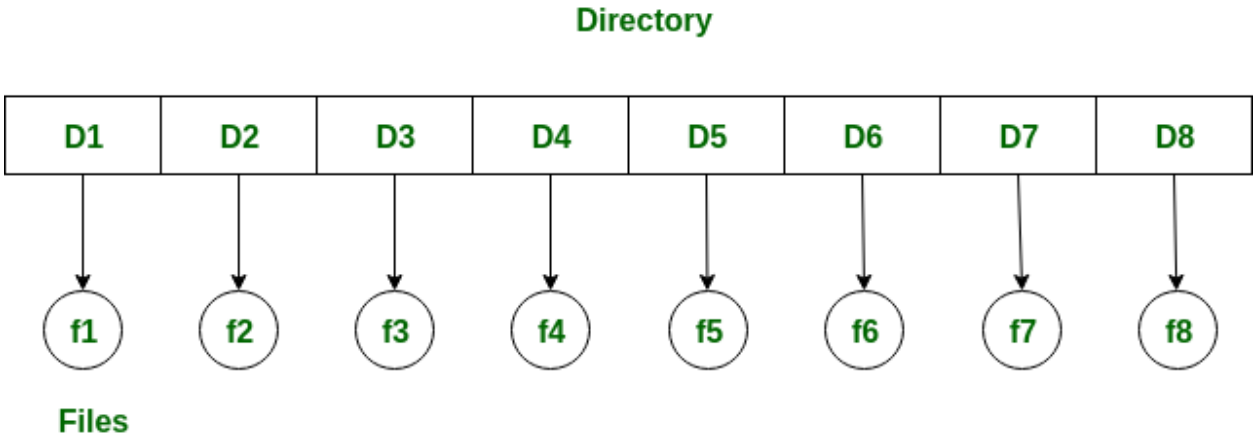
1. Definition – Page fault dominates like an error. If any program tries to access piece of memory but which is not existed into physical memory, means main memory, then page fault will be occurred. The fault specifies the O/S that it must trace the all data in to virtual memory management, and after that moves it from secondary memory like as hard disk to primary memory of system.

Page Fault Diagram

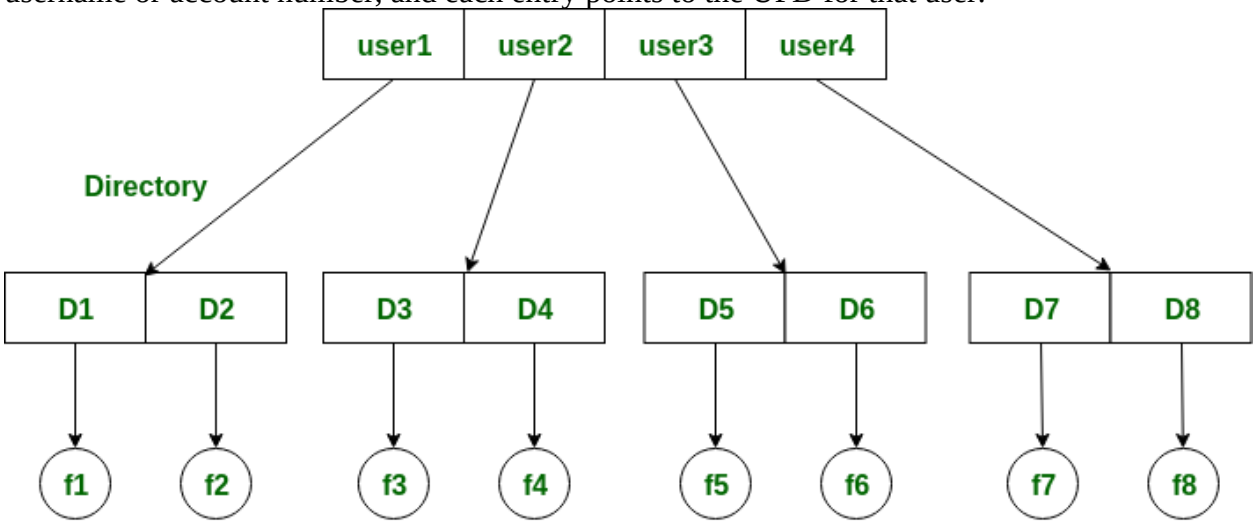


Page Fault Handling

All hardware components of computer monitor program counter and kernel that are saved in the stack, and CPU registers have to store all current running state information.
Assembly programs help to store general registers as well as volatile information.
Page fault is searched by the operating system, and trace the virtual pages which are needed.
Hardware registers also consist the all those needed information.
But, if required page is not loaded in to memory, then page fault trap is arisen.
Now these steps must be followed for handling the page fault.
Firstly, get check to main memory address requests; ensure those requests must be valid.
If reference got invalid then the process will be terminated, otherwise the page is going to paged in.
Then, Free-frame list locates the free frame.
Now disk operation will be scheduled to fetch the needed page from disk.

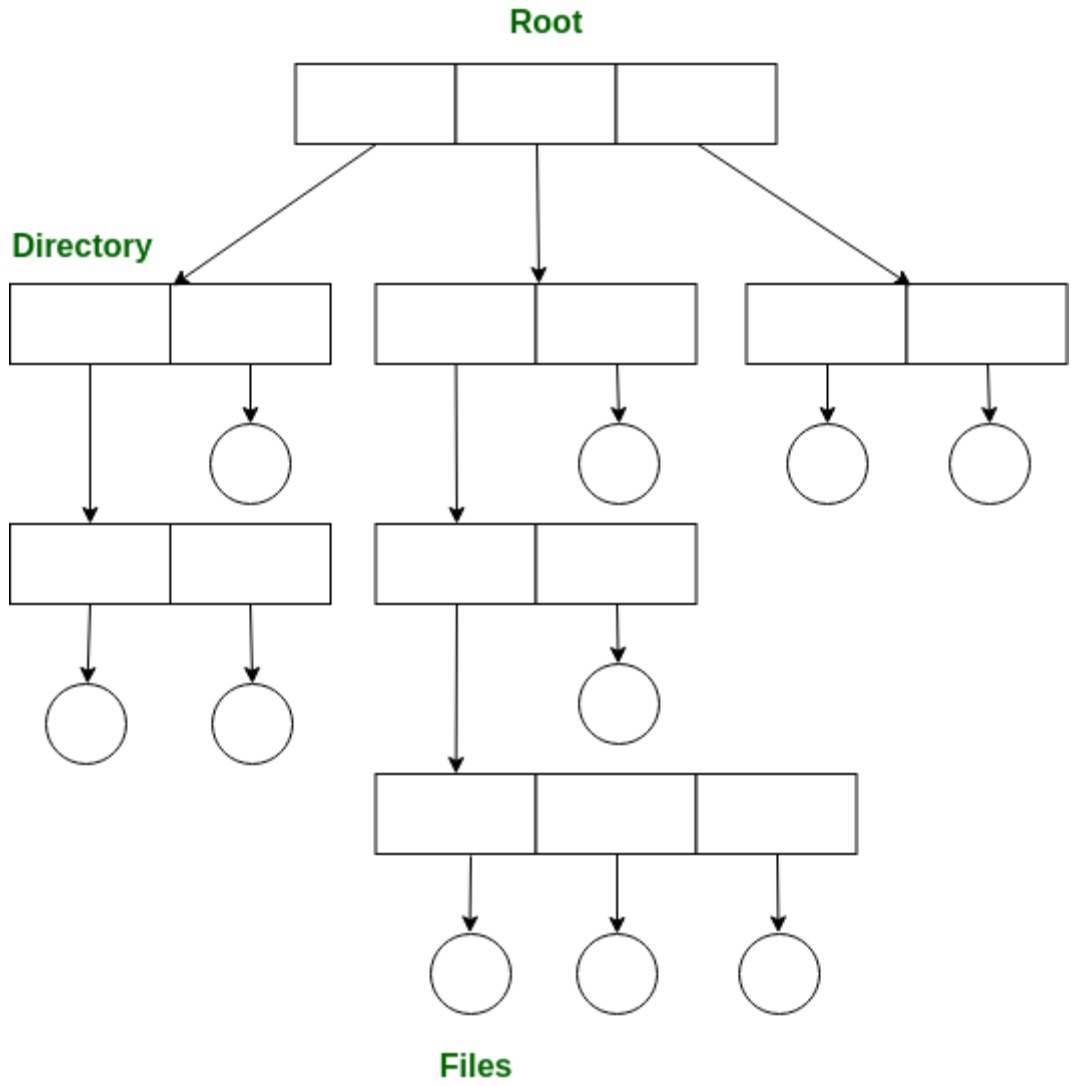
	When I/O operation is done, and new frame number will be added in the process's page table, and invalid bit is altered. Now this is valid page reference. If, further any page fault will be found then recycle these instructions from starting.
2.	A directory is a container that is used to contain folders and files. It organizes files and folders in a hierarchical manner.  Files There are several logical structures of a directory, these are given below. Single-level directory – The single-level directory is the simplest directory structure. In it, all files are contained in the same directory which makes it easy to support and understand. A single level directory has a significant limitation, however, when the number of files increases or when the system has more than one user. Since all the files are in the same directory, they must have a unique name. if two users call their dataset test, then the unique name rule violated.  Files Advantages: Since it is a single directory, so its implementation is very easy. If the files are smaller in size, searching will become faster.

The operations like file creation, searching, deletion, updating are very easy in such a directory structure.
Disadvantages:
There may chance of name collision because two files can not have the same name.
Searching will become time taking if the directory is large.
This can not group the same type of files together.
Two-level directory –
As we have seen, a single level directory often leads to confusion of files names among different users. the solution to this problem is to create a separate directory for each user.
In the two-level directory structure, each user has their own user files directory (UFD). The UFDs have similar structures, but each lists only the files of a single user. system's master file directory (MFD) is searches whenever a new user id=s logged in. The MFD is indexed by username or account number, and each entry points to the UFD for that user.



Advantages:
We can give full path like /User-name/directory-name/.
Different users can have the same directory as well as the file name.
Searching of files becomes easier due to pathname and user-grouping.
Disadvantages:
A user is not allowed to share files with other users.
Still, it not very scalable, two files of the same type cannot be grouped together in the same user.

Tree-structured directory –
Once we have seen a two-level directory as a tree of height 2, the natural generalization is to extend the directory structure to a tree of arbitrary height.
This generalization allows the user to create their own subdirectories and to organize their files accordingly.

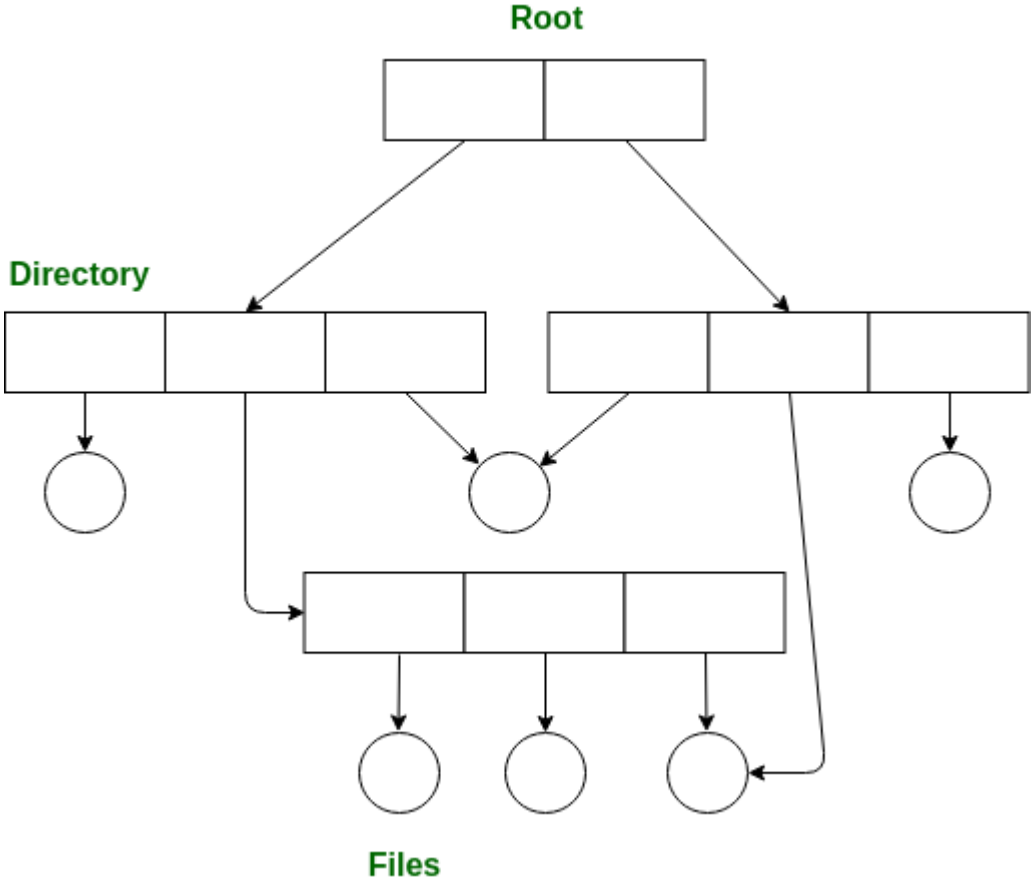


A tree structure is the most common directory structure. The tree has a root directory, and every file in the system has a unique path.
Advantages:
Very general, since full pathname can be given.
Very scalable, the probability of name collision is less.
Searching becomes very easy, we can use both absolute paths as well as relative.
Disadvantages:

Every file does not fit into the hierarchical model, files may be saved into multiple directories.
We can not share files.
It is inefficient, because accessing a file may go under multiple directories.

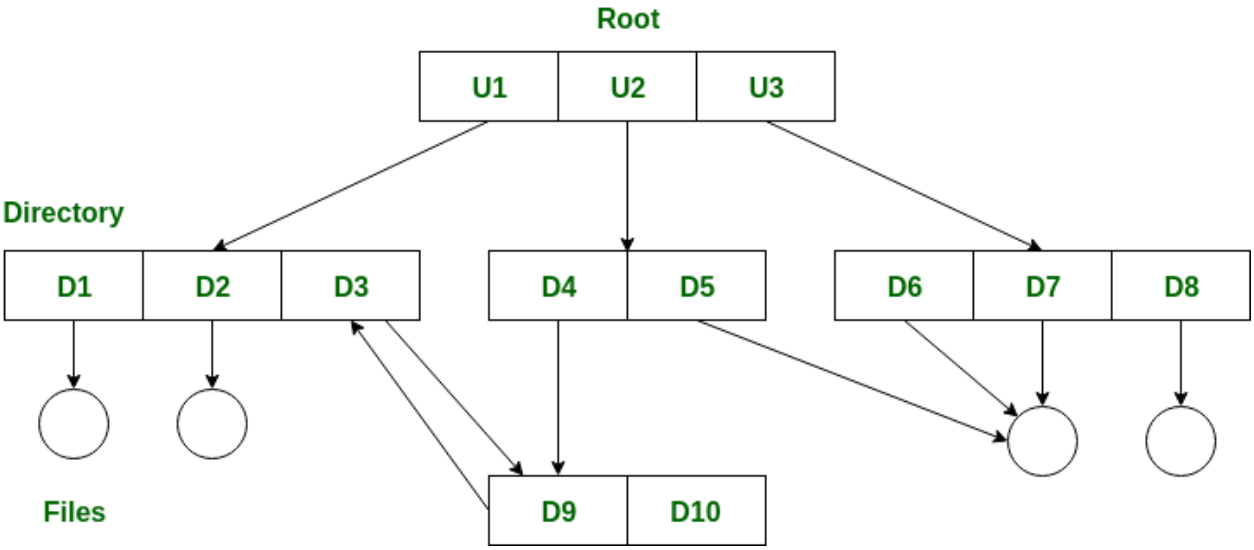
Acyclic graph directory –
An acyclic graph is a graph with no cycle and allows us to share subdirectories and files. The same file or subdirectories may be in two different directories. It is a natural generalization of the

tree-structured directory.
It is used in the situation like when two programmers are working on a joint project and they need to access files. The associated files are stored in a subdirectory, separating them from other projects and files of other programmers since they are working on a joint project so they want the subdirectories to be into their own directories. The common subdirectories should be shared. So here we use Acyclic directories.
It is the point to note that the shared file is not the same as the copy file. If any programmer makes some changes in the subdirectory it will reflect in both subdirectories.



Advantages:
We can share files.
Searching is easy due to different-different paths.
Disadvantages:
We share the files via linking, in case deleting it may create the problem,
If the link is a soft link then after deleting the file we left with a dangling pointer.
In the case of a hard link, to delete a file we have to delete all the references associated with it.

General graph directory structure –
In general graph directory structure, cycles are allowed within a directory structure where multiple directories can be derived from more than one parent directory.
The main problem with this kind of directory structure is to calculate the total size or space that has been taken by the files and directories.



Advantages:
It allows cycles.
It is more flexible than other directories structure.
Disadvantages:
It is more costly than others.
It needs garbage collection.

3.

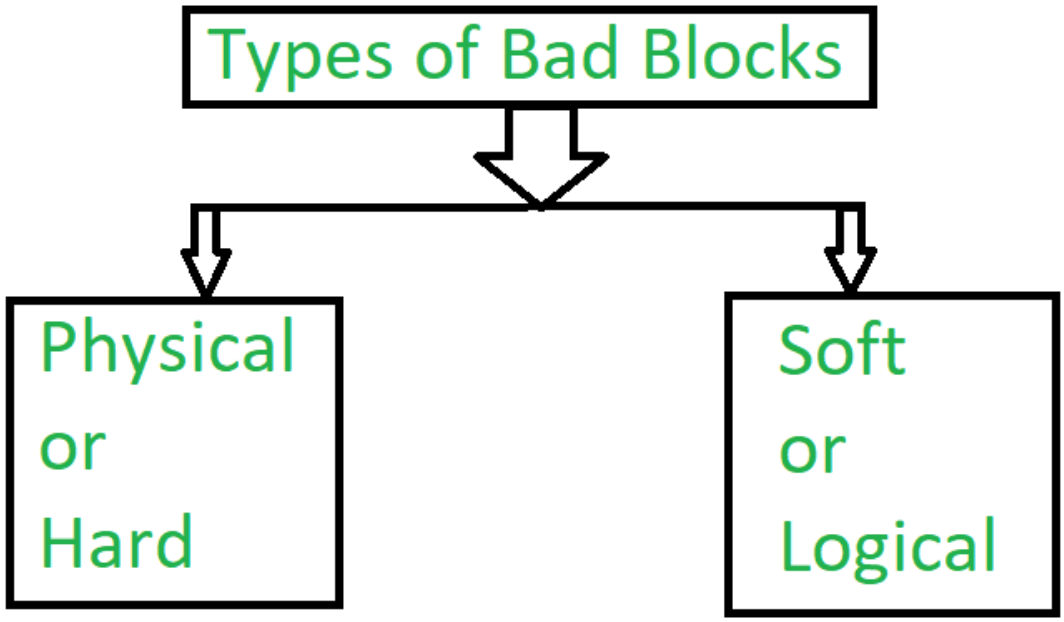
ccess Matrix is a security model of protection state in computer system. It is represented as a matrix. Access matrix is used to define the rights of each process executing in the domain with respect to each object. The rows of matrix represent domains and columns represent objects. Each cell of matrix represents set of access rights which are given to the processes of domain means each entry(i, j) defines the set of operations that a process executing in domain Di can invoke on object Oj.

	F1	F2	F3	Printer
D1	read		read	
D2				print
D3		read	execute	
D4	read write		read write	

According to the above matrix: there are four domains and four objects- three files(F1, F2, F3) and one printer. A process executing in D1 can read files F1 and F3. A process executing in domain D4 has same rights as D1 but it can also write on files. Printer can be accessed by only one process executing in domain D2. The mechanism of access matrix consists of many policies and semantic properties. Specifically, We must ensure that a process executing in domain Di can access only those objects that are specified in row i.

	Policies of access matrix concerning protection involve which rights should be included in the (i, j)th entry. We must also decide the domain in which each process executes. This policy is usually decided by the operating system. The Users decide the contents of the access-matrix entries.
4.	Access Control List (ACL): An access control list (ACL) is a table that tells a computer operating system which access rights each user has to a particular system object, such as a file directory or individual file. Each object has a security attribute that identifies its access control list. The list has an entry for each system user with access privileges. The most common privileges include the ability to read a file (or all the files in a directory), to write to the file or files, and to execute the file (if it is an executable file, or program). Microsoft Windows NT/2000, Novell's NetWare, Digital's OpenVMS, and UNIX-based systems are among the operating systems that use access control lists. Capability List: A capability is a token, ticket, or key that gives the possessor permission to access an entity or object in a computer system. A capability can be thought of as a pair (x, r) where x is the name of an object and r is a set of privileges or rights. With each subject we can store that subject's capabilities. And, the subject presents to the guard a capability in order to get access to an object. Note that a capability is completely transferable; it doesn't matter who presents the capability. This framework completely eliminates the need for authentication. However, with ACLs we were assuming that authentication was unforgettable. With capabilities, we now need a way to make capabilities unforgettable. The success of a capability-based mechanism depends on it. Comparison of Access control list and Capability list Consider the Real-Life Analogy: Bank Analogy Carla wishes to keep all of her valuables in a safe deposit box in the bank. On occasion, she would like one or more trustworthy friends to make deposits or withdrawals. There are two ways that the bank can control access to the box: i. The bank maintains a list of people authorized to access the box. ii. The bank issues Carla one or more keys to the safe deposit box. ACL Approach i. Bank's involvement : The bank must (i) store the list, (ii) verify users. ii. Forging access right: The bank must safeguard the list. The bank must authenticate. iii. Add a new person : The owner must visit the bank. iv. Delegation : A friend cannot extend his or her privilege to someone else. v. If a friend becomes untrustworthy, the owner can remove his/her name. Capability Approach i. Bank's involvement : The bank need not be involved in any transactions ii. Forging access right : The key cannot be forged iii. Adding a new person: The owner can give the key to the new person iv. Delegation : A friend can extend his or her privilege to someone else. v. Revoke : The owner can ask for the key back, but it may not be possible to know whether or not the friend has made a copy.

5.	Bad Block is an area of storing media that is no longer reliable for the storage of data because it is completely damaged or corrupted. We know disk have moving parts and have small tolerances, they are prone to failure. In case when the failure is complete, then the disk needs to be replaced and its contents restored from backup media to the new disk. More frequently, one or more sectors become defective. More disks even come from the factory named Bad blocks. This is also referred to as Bad Sector. Cause of Bad Block : Storage drives can ship from the factory with defective blocks that originated in the manufacturing process. The device with bad-blocks are marked as defective before leaving the factory. These are remapped with the available extra memory cells. A physical damage to device also makes a device as bad block because sometimes operating system does not able to access the data. Dropping a laptop will also cause damage to the platter of the HDD's. Sometimes dust also cause damage to HDD's. When the memory transistor fails it will cause damage to the solid-state drive. Storage cells can also become unreliable over time, as NAND flash substrate in a cell becomes unusable after a certain number of program-erase cycles. For the erase process on the solid-state drive it requires a huge amount of electrical charge through the flash cards. This degrades the oxide layer that separates the floating gate transistors from the flash memory silicon substrate and the bit error rates increase. The drive's controller can use error detection and correction mechanisms to fix these errors. However, at some point, the errors can outstrip the controller's ability to correct them and the cell can become unreliable. Soft bad sectors are caused by software problems. For instance, if a computer unexpectedly shuts down, due to this, hard drive also turn of in the middle of writing to a block. Due to this, the data contain in the block doesn't match with the CRC detection error code and it would marked as bad sector. Types of Bad Blocks : There are two types of bad blocks –



Physical or Hard bad block :
It comes from damage to the storage medium.
Soft or Logical bad block :
A soft, or logical, bad block occurs when the operating system (OS) is unable to read data from a sector.
Example –
A soft bad block include when the cyclic redundancy check (CRC), or error correction code (ECC), for a particular storage block does not match the data read by the disk.
How Bad blocks are handled :
These blocks are handled in a number of ways, but it depends upon the disk and controller.

On simple disks, such as some disks with IDE controller, bad blocks are handled manually. One strategy is to scan the disk to find bad blocks while disk is being formatted. Any bad block that are discovered as flagged as unusable so that file system does not allocate them. If blocks go bad during normal operation, a special program (such as the Linux badblocks command) must be run manually to search for the bad blocks and to lock them away.
More sophisticated disks are smarter about bad-block recovery. The work of controller is to maintain the list of bad blocks. The list formed by the controller is initialized during the low-level formatting at the factory and is updated over the life of the disk. Low-level formatting holds the spare sectors which are not visible to the operating system.The last task is done by controller which is to replace each bad sector logically with the spare sectors. This scheme is also known as sector sparing and forwarding.
A typical bad-sector transaction is as follows –
Suppose Operating system wants to read logical block 80.
Now, the controller is going to calculate EEC and suppose it found the block as bad. It reports to

	operating system that the requested block is bad. Whenever, next time the system is rebooted, a special command is used and it will tell the controller that this sector is to be replaced with the spare sector. In future, whenever there is a request for the block 80, the request is translated to replacement sector's address by the controller. Note – The redirection by the controller (i.e., the request translated to replacement) could invalidate any optimization by the operating system's disk-scheduling algorithm. For this reason, most disks are formatted to provide a few spare sectors in each cylinder and spare cylinder as well. Whenever the bad block is going to remap, the controller will use spare sector from the same cylinder, if possible; otherwise spare cylinder is also present. Some controllers use spare sector to replace bad block, there is also another technique to replace bad block which is sector slipping. Example of sector slipping – Suppose that logical block 16 becomes defective and the first available spare sector follows sector 200. Sector slipping then starts remapping. All the sectors from 16 to 200, moving them all down one spot. That is, sector 200 is copied into the spare, then sector 199 into 200, then 198 into 199, and so on, until sector 17 is copied into sector 18. In this way slipping the sectors frees up the space of sector 17 so that sector 16 can be mapped to it. The replacement of bad block is not totally automatic, because data in the bad block are usually lost. A process is trigger by the soft errors in which a copy of the block data is made and the block is spared or slipped. Hard error which is unrecoverable will lost all its data. Whatever file was using that block must be repaired and that requires manual intervention.
6.	This model is based on the above-stated concept of the Locality Model. The basic principle states that if we allocate enough frames to a process to accommodate its current locality, it will only fault whenever it moves to some new locality. But if the allocated frames are lesser than the size of the current locality, the process is bound to thrash. According to this model, based on a parameter A, the working set is defined as the set of pages in the most recent 'A' page references. Hence, all the actively used pages would always end up being a part of the working set. The accuracy of the working set is dependent on the value of parameter A. If A is too large, then working sets may overlap. On the other hand, for smaller values of A, the locality might not be covered entirely. If D is the total demand for frames and is the working set size for a process i, Now, if 'm' is the number of frames available in the memory, there are 2 possibilities: (i) D>m i.e. total demand exceeds the number of frames, then thrashing will occur as some processes would not get enough frames. (ii) D<=m, then there would be no thrashing.

7.

Q7.

i) LRU:

frames.	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
f_1	1	1	1	1	1	1	1	1	1	1	1	1	1	6	6	6	6	6	6	6
f_2		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
f_3			3	3	3	3	5	5	5	5	5	3	3	3	3	3	3	3	3	3
f_4				4	4	4	4	6	6	6	6	6	7	7	7	7	1	1	1	1
	x	x	x				x	x					x	x	x			x		

No. of page faults = 10

ii) FIFO:

frames.	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
f_1	1	1	1	1	1	1	5	5	5	5	5	3	3	3	3	3	1	1	1	1
f_2		2	2	2	2	2	2	6	6	6	6	6	7	7	7	7	7	3	3	
f_3			3	3	3	3	3	3	2	2	2	2	2	6	6	6	6	6	6	
f_4				4	4	4	4	4	4	1	1	1	1	1	1	2	2	2	2	2
	x	x	x	x			x	x	x	x			x	x	x			x	x	

No. of page faults. = 14

3) Optimal.

Frames.

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
F_1	1	1	1	1	1	1	1	1	1	1	1	1	7	7	7	7	1	1	1	1
F_2		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
F_3			3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
F_4				4	4	4	5	6	6	6	6	6	6	6	6	6	6	6	6	6
	x	x	x	x			x	x					x				x			

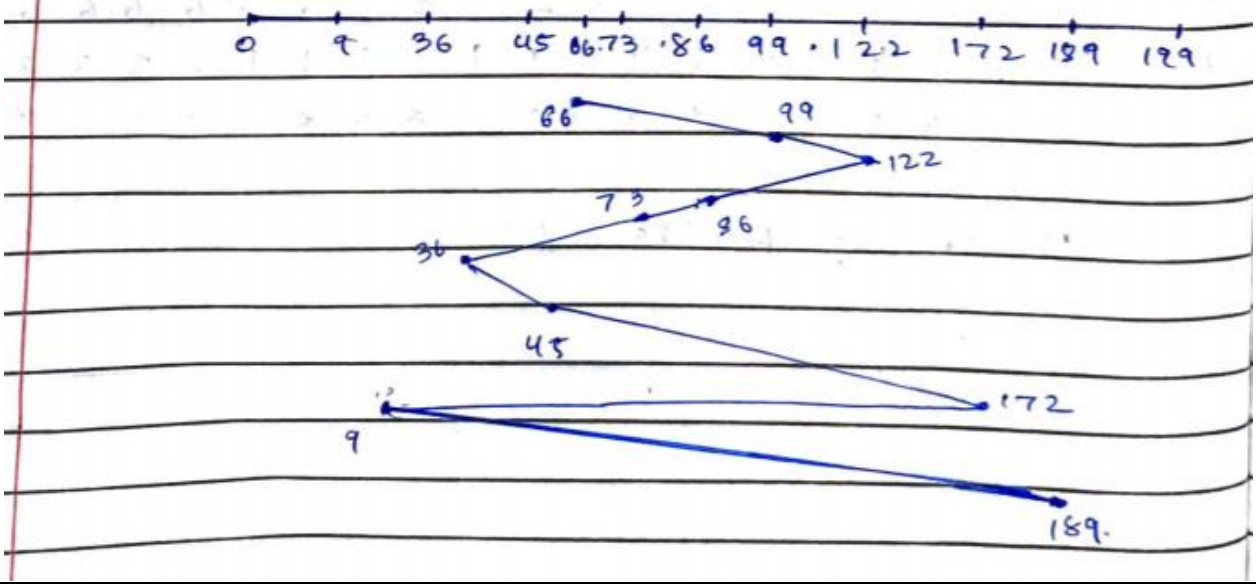
No. of page faults. = 8.

8.

Cylinder read sequence : 99, 122, 86, 73, 36, 45, 172, 9, 189.

current head position : 66.

1) FCFS.



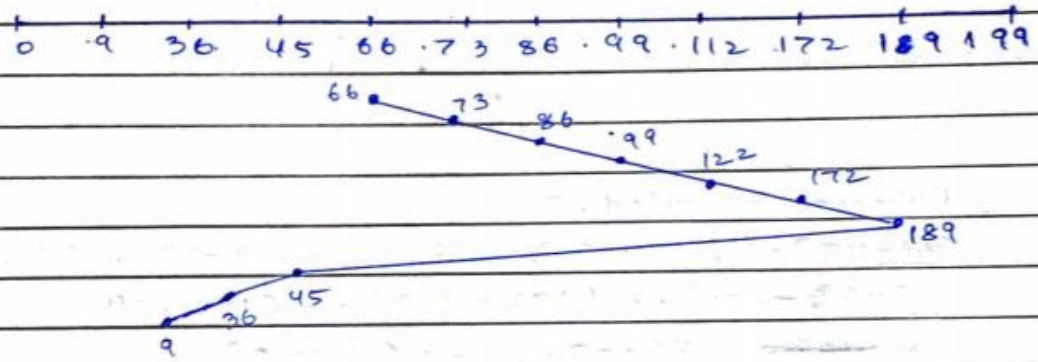
Distance travelled.

→ (99-66) + (122-99) + (122-86) + (86-73) + (73-36) + (45-36) + (172-45) + (172-9) + (189-9)

⇒ 33 + 23 + 36 + 13 + 37 + 9 + 127 + 163 + 180

= 621

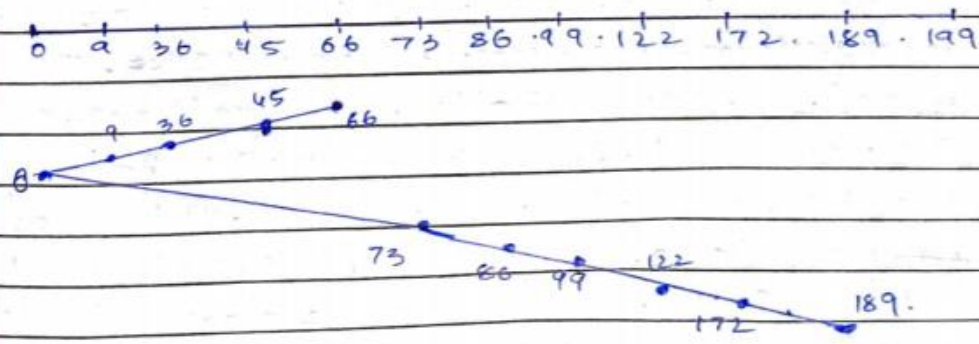
ii) SSTF.



Distance travelled.

→ (73-66) + (86-73) + (99-86) + (112-99) + (172-122) + (189-172) + (189-45) + (45-36) + (36-9)

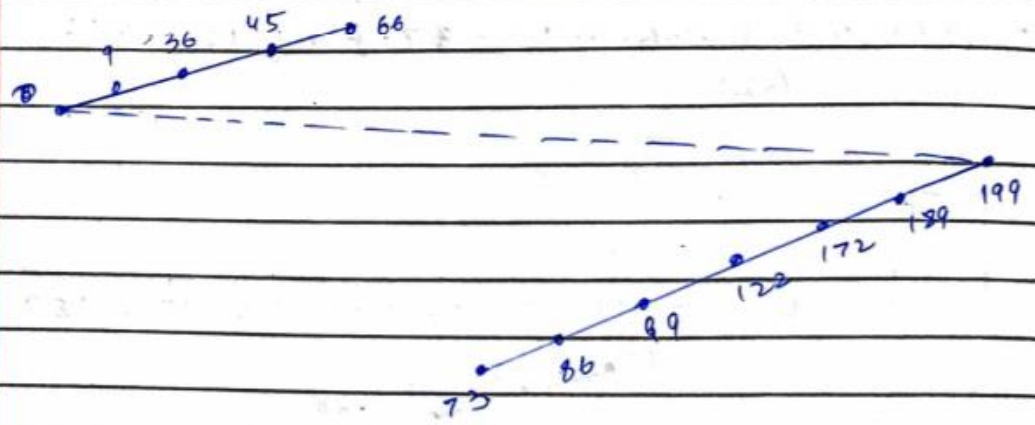
iii) SCAN



Distance travelled ⇒ (66-45) + (45-36) + (36-9) + (9-0) + (73-0) + (86-73) + (99-86) + (122-99) + (172-122) + (189-172) = 225

IV. C-SCAN.

0 · 9 · 36 · 45 · 66 · 73 · 86 · 99 · 122 · 172 · 189 · 199



Distance travelled. =

(66-45) + (45-36) + (36-9) + (9-0) + (199-189) + (189-172) + (172-122) + (122-99) + (99-86) + (86-73) + (199-0)

= 391.