16SCCCS8 /16SCCCA6/16SCCCIT7- OPERATING SYSTEMS
UNIT – IV& V
Part – A (2 Marks)

1. Define Device management and mention its types.

The process of implementation, operation and maintenance of a device by operating system is called device management. Operating system uses utility software called device driver as interface to the device.

Types: Dedicated, Shared and Virtual devices

2. Write the Difference between Sequential and Direct Access Storage.

S.No	Sequential Access Storage	Direct Access Storage
01.	Sequential access must begin at the beginning and access each element in order, one after the other	Direct access allows the access of any element directly by locating it by its index number or address.
02.	Magnetic tape has only sequential access, but CDs had direct access.	Arrays allow direct access.
03.	It is Slower than Direct Access Storage.	Direct access is faster than sequential access, but it requires some external mechanism (array index, file byte number, etc).

3. What is Interrecord Gap?

The tape needs time and space to stop, so a gap is inserted between each record , that gap is called **Interrecord gap (IRG)** which is about 1⁄2 inch long regardless of the sizes of the records it separates. Therefore, if 10 records are stored individually, there will be nine 1⁄2-inch IRGs between each record.

4. How to calculate a Transfer rate .?

The transfer rate, which is the density of the tape (measured in bpi), multiplied by the tape drive speed, called transport speed, which is measured in inches per second (ips):

Transfer rate (ips) = density * transport speed

5. What are the 3 Factors that contribute for time required to access a file?

Depending on whether a disk has fixed or movable heads, there can be as many as three factors that contribute to the time required to access a file:

- seek time,
- search time and
- transfer time.

6. Define RAID.

“Redundant Arrays of Independent Disks” is a technique which makes use of a combination of multiple disks instead of using a single disk for increased performance, data redundancy or both. The term was coined by David Patterson, Garth A. Gibson, and Randy Katz at the University of California, Berkeley in 1987.

7. Differentiate RAID 0 and RAID 1.

S.NO	RAID 0	RAID 1
1.	RAID 0 stands for Redundant Array of Independent Disk level 0.	While RAID 1 stands for Redundant Array of Independent Disk level 1.
2.	In RAID 0 technology, Disk stripping is used.	While in RAID 1 technology, Disk mirroring is used.
3.	The cost of RAID 0 technology is low.	While RAID 1 is costly or expensive.
4.	In RAID 0, There is no write penalty.	While in RAID 1, There is write penalty.
5.	The Relative storage efficiency of RAID 0 is 100%.	While the relative storage efficiency of RAID 1 is 50%.

8. Define File Management in OS

The File Manager (also called the file management system) is the software responsible for creating, deleting, modifying, and controlling access to files—as well as for managing the resources used by the files. The File Manager provides support for libraries of programs and data to online users, for spooling operations, and for interactive computing. These functions are performed in collaboration with the Device Manager.

9. What is called Relative Filename?

The relative filename is the name that differentiates it from other files in the same directory.

10. Write differences between fixed and variable length record.

S. No	Fixed Length Record	Variable Length Record
1.	A file where all the records are of the same length is said to have fixed length records.	One or more of the fields can be of differing lengths in each record, called variable length records
2.	**Advantage :** Access is fast because the computer knows where each record starts.	**Advantage:** The records will be smaller and will need less storage space the records will load faster
3.	**Disadvantage :** Using Fixed length records, the records are usually larger and therefore need more storage space and are slower to transfer (load or save).	**Disadvantages:** The computer will be unable to determine where each record starts so processing the records will be slower.

11. Define Access methods and write its types.

Access methods are dictated by a file's organization; the most flexibility is allowed with indexed sequential files and the least with sequential. A file that has been organized in sequential fashion can support only sequential access to its records, and these records can be of either fixed or variable length. The File Manager uses the address of the last byte read to access the next sequential record. Therefore, the current byte address (CBA) must be updated every time a record is accessed, such as when the READ command is executed.

Types : (i) Sequential Access **(ii)** Direct access.

12. Write a formula to find Block Number and Offset value of record in File System

$$\text{block number} = \text{integers}\left[\frac{\text{byte address}}{\text{physical block size}}\right] + \text{address of the first physical record}$$

$$\text{offset} = \text{remainder}\left[\frac{\text{byte address}}{\text{physical block size}}\right]$$

13. What are the file management functions?

- Keeps track of information, location, uses, status etc.
- Decides who gets the resources.
- Allocates the resources.

14. Define Access matrix

The access control matrix is intuitively appealing and easy to implement, but because of its size it only works well for systems with a few files and a few users. In the matrix, each column identifies a user and each row identifies a file

15. Write difference between Contiguous and Non Contiguous Memory allocation.

Basis The Comparison	Contiguous Memory Allocation	Noncontiguous Memory Allocation
Basic	Allocates consecutive blocks of memory to a process.	Allocates separate blocks of memory to a process.
Overheads	Contiguous memory allocation does not have the overhead of address translation while execution of a process.	Non contiguous memory allocation has overhead of address translation while execution of a process.
Execution rate	A process executes faster in contiguous memory allocation	A process executes quite slower comparatively in non contiguous memory allocation.

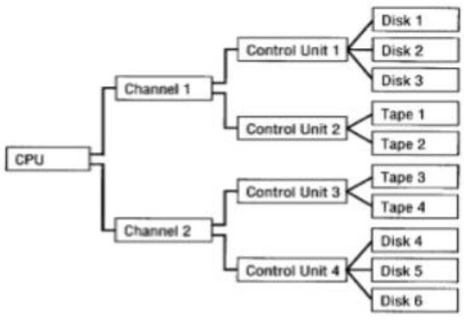
Part – B (5 Marks)

1. Describe about Components of I/O Subsystem.

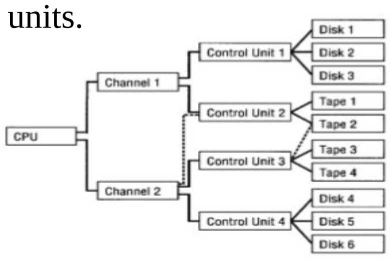
The I/O subsystems components perform similar functions like Taxi Cab Company. The channel plays the part of the dispatcher. Its job is to keep up with the I/O requests from the CPU and pass them down the line to the appropriate control unit. The control units play the part of the mechanics. The I/O devices play the part of the vehicles. I/O channels are programmable units placed between the CPU and the control units. Their job is to synchronize the fast speed of the CPU with the slow speed of the I/O device, and they make it possible to overlap I/O operations with processor operations so the CPU and I/O can process concurrently. Channels use I/O channel programs, which can range in size from one to many instructions. Each channel program specifies the action to be performed by the devices and controls the transmission of data between main memory and the control units. The channel sends one signal for each function, and the I/O control unit interprets the signal, which might say "go to the top of the page" if the device is a printer or "rewind" if the device is a tape drive. The operating system normally deals with the controller, not the device.

At the start of an I/O command, the information passed from the CPU to the channel is this:

- I/O command (READ, WRITE, REWIND, etc.)
- Channel number
- Address of the physical record to be transferred (from or to secondary storage)
- Starting address of a memory buffer from which or into which the record is to be transferred



A typical configuration might have one channel and up to eight control units, each of which communicates with up to eight I/O devices. Channels are often shared because they're the most expensive items in the entire I/O subsystem. The system shown in above Figure requires that the entire path be available when an I/O command is initiated. However, there's some flexibility built into the system because each unit can end independently of the others, as will be explained in the next section. This figure also shows the hierarchical nature of the interconnection and the one-to-one correspondence between each device and its transmission path. Additional flexibility can be built into the system by connecting more than one channel to a control unit or by connecting more than one control unit to a single device. That's the same as if the mechanics of the Flynn Taxicab Company could also make repairs for the ABC Taxicab Company, or if its vehicles could be used by ABC drivers (or if the drivers in the company could share vehicles). These multiple paths increase the reliability of the I/O subsystem by keeping communication lines open even if a component malfunctions. The second Figure shows the same system presented in first Figure, but with one control unit connected to two channels and one device connected to two control units.



2. Explain in detail about management of I/O requests.

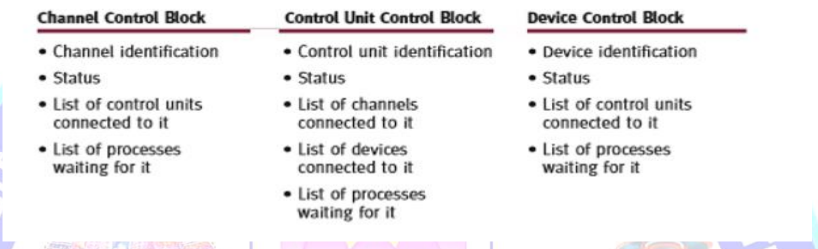
The Device Manager divides the task into three parts with each one handled by a specific software component of the I/O subsystem. The I/O traffic controller watches the status of all devices, control units, and channels. The I/O scheduler implements the policies that determine the allocation of, and access to, the devices, control units, and channels. The I/O device handler performs the actual transfer of data and processes the device interrupts.

The I/O traffic controller monitors the status of every device, control unit, and channel. It's a job that becomes more complex as the number of units in the I/O subsystem increases and as the number of paths between these units increases.

The traffic controller has three main tasks:

- It must determine if there's at least one path available;
- If there's more than one path available, it must determine which to select; and
- If the paths are all busy, it must determine when one will become available.

To do all this, the traffic controller maintains a database containing the status and connections for each unit in the I/O subsystem, grouped into Channel Control Blocks, Control Unit Blocks, and Device Control Blocks, as shown in following Table.



Channel Control Block	Control Unit Control Block	Device Control Block
• Channel identification	• Control unit identification	• Device identification
• Status	• Status	• Status
• List of control units connected to it	• List of channels connected to it	• List of control units connected to it
• List of processes waiting for it	• List of devices connected to it	• List of processes waiting for it
	• List of processes waiting for it	

The I/O scheduler performs the same job as the Process Scheduler—that is, it allocates the devices, control units, and channels. Under heavy loads, when the number of requests is greater than the number of available paths, the I/O scheduler must decide which request to satisfy first. In many systems, the major difference between I/O scheduling and process scheduling is that I/O requests are not pre-empted.

Once the channel program has started, it's allowed to continue to completion even though I/O requests with higher priorities may have entered the queue. This is feasible because channel programs are relatively short, 50 to 100 ms. Other systems subdivide an I/O request into several stages and allow preemption of the I/O request at any one of these stages. Some systems allow the I/O scheduler to give preferential treatment to I/O requests from high-priority programs.

In that case, if a process has high priority, then its I/O requests would also have high priority and would be satisfied before other I/O requests with lower priorities. The I/O scheduler must synchronize its work with the traffic controller to make sure that a path is available to satisfy the selected I/O requests. The I/O device handler processes the I/O interrupts, handles error conditions, and provides detailed scheduling algorithms, which are extremely device dependent. Each type of I/O device has its own device handler algorithm.

3. Describe about Communication among devices.

The Device Manager relies on several auxiliary features to keep running efficiently under the demanding conditions of a busy computer system, and there are three problems that must be resolved:

- It needs to know which components are busy and which are free.
- It must be able to accommodate the requests that come in during heavy I/O traffic.
- It must accommodate the disparity of speeds between the CPU and the I/O devices.

The first is solved by structuring the interaction between units. The last two problems are handled by buffering records and queuing requests. For example, after a device has begun writing a record, and before it has completed the task, the connection between the device and its control unit can be cut off so the control unit can initiate another I/O task with another device.

Meanwhile, at the other end of the system, the CPU is free to process data while I/O is being performed, which allows for concurrent processing and I/O. The success of the operation depends on the system's ability to know when a device has completed an operation. This is done with a hardware flag that must be tested by the CPU. This flag is made up of three bits and resides in the **Channel Status Word (CSW)**, which is in a predefined location in main memory and contains information indicating the status of the channel. Each bit represents one of the components of the I/O subsystem, one each for the channel, control unit, and device. Each bit is changed from 0 to 1 to indicate that the unit has changed from free to busy.

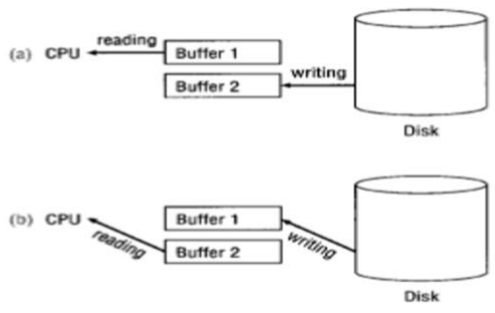
Each component has access to the flag, which can be tested before proceeding with the next I/O operation to ensure that the entire path is free and vice versa. There are two common ways to perform this test—polling and using interrupts. Polling uses a special machine instruction to test the flag. The interrupt handler's job is to determine the best course of action based on the current situation because it's not unusual for more than one unit to have caused the I/O interrupt.

Direct memory access (DMA) is an I/O technique that allows a control unit to directly access main memory. This means that once reading or writing has begun, the remainder of the data can be transferred to and from memory without CPU intervention. However, it is possible that the DMA control unit and the CPU compete for the system bus if they happen to need it at the same time. To activate this process, the CPU sends enough information—such as the type of operation (read or write), the unit number of the I/O device needed, the location in memory where data is to be read from or written to, and the amount of data (bytes or words) to be transferred—to the DMA control unit to initiate the transfer of data; the CPU then can go on to another task while the control unit completes the transfer independently.

The DMA controller sends an interrupt to the CPU to indicate that the operation is completed. This mode of data transfer is used for high-speed devices such as disks. Without DMA, the CPU is responsible for the physical movement of data between main memory and the device a time-consuming task that results in significant overhead and decreased CPU utilization.

Buffers are used extensively to better synchronize the movement of data between the relatively slow I/O devices and the very fast CPU. Buffers are temporary storage areas residing in three convenient locations throughout the system: main memory, channels, and control units. They're used to store data read from an input device before it's needed by the processor and to store data that will be written to an output device. A typical use of buffers occurs when blocked records are either read from, or written to, an I/O device. In this case, one physical record contains several logical records and must reside in memory while the processing of each individual record takes place.

To minimize the idle time for devices and, even more important, to maximize their throughput, the technique of double buffering is used, as shown in following Figure . In this system, two buffers are present in main memory, channels, and control units. The objective is to have a record ready to be transferred to or from memory at any time to avoid any possible delay that might be caused by waiting for a buffer to fill up with data. Thus, while one record is being processed by the CPU, another can be read or written by the channel.



When using blocked records, upon receipt of the command to “READ last logical record” the channel can start reading the next physical record, which results in overlapped I/O and processing. When the first READ command is received, two records are transferred from the device to immediately fill both buffers. Then, as the data from one buffer has been processed, the second buffer is ready. As the second is being read, the first buffer is being filled with data from a third record, and so on.

4. Explain about File System in OS.

A file is a collection of related information that is recorded on secondary storage. Or file is a collection of logically related entities. From user’s perspective a file is the smallest allotment of logical secondary storage.

Attribute	Type	Operation
Name	Doc	Create
Type	Exe	Open
Size	Jpg	Read
Creation Data	Xis	Write
Author	C	Append
Last Modified	Java	Truncate
protection	class	Delete

File Type	Usual Extension	Function
Executable	exe, com, bin	Read to run machine language program
Object	obj, o	Compiled, machine language not linked
Source Code	C, java, pas, asm, a	Source code in various languages
Batch	bat, sh	Commands to the command interpreter
Text	txt, doc	Textual data, documents
Word Processor	wp, tex, rrf, doc	Various word processor formats
Archive	arc, zip, tar	Related files grouped into one compressed file

FILE DIRECTORIES:

Collection of files is a file directory. The directory contains information about the files, including attributes, location and ownership. Much of this information especially that is concerned with storage is managed by the operating system. The directory is itself a file, accessible by various file management routines. Information contained in a device directory are

- Name
- Type
- Address
- Current length
- Maximum length
- Date last accessed
- Date last updated
- Owner id
- Protection information

Advantages of maintaining directories are

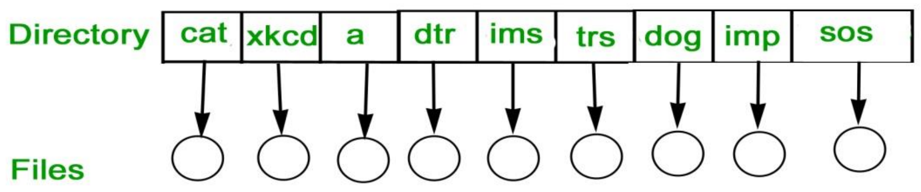
i. Efficiency: A file can be located more quickly.
ii. Naming: It becomes convenient for users as two users can have same name for different files or may have different name for same file.
iii. Grouping: Logical grouping of files can be done by properties e.g. all java programs, all games etc.

1. SINGLE-LEVEL DIRECTORY

In this a single directory is maintained for all the users.

Naming problem : Users cannot have same name for two files.

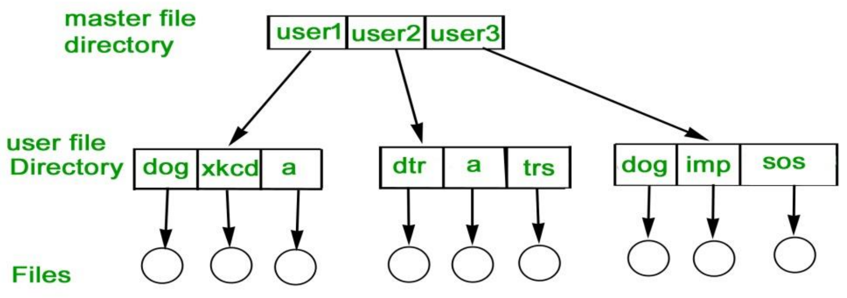
Grouping problem : Users cannot group files according to their need.



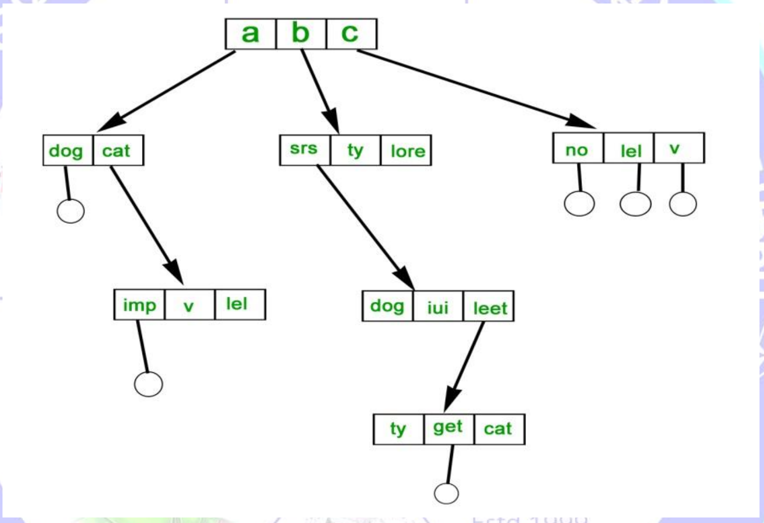
2. TWO-LEVEL DIRECTORY

In this separate directories for each user is maintained.

- Path name:Due to two levels there is a path name for every file to locate that file.
- Now,we can have same file name for different user.
- Searching is efficient in this method.



3. **TREE-STRUCTURED DIRECTORY :**
   Directory is maintained in the form of a tree. Searching is efficient and also there is grouping capability. We have absolute or relative path name for a file.



5. Explain about Access methods in File Management.

When a file is used, information is read and accessed into computer memory and there are several ways to access this information of the file. Some systems provide only one access method for files. Other systems, such as those of IBM, support many access methods, and choosing the right one for a particular application is a major design problem.

There are three ways to access a file into a computer system:

- Sequential-Access,
- Direct Access,
- Index sequential Method.

1. **Sequential Access**

It is the simplest access method. Information in the file is processed in order, one record after the other. This mode of access is by far the most common; for example, editor and compiler usually access the file in this fashion.

Read and write make up the bulk of the operation on a file. A read operation -read next- read the next position of the file and automatically advances a file pointer, which keeps track I/O location. Similarly, for the write next append to the end of the file and advance to the newly written material.

Key points:

- ✓ Data is accessed one record right after another record in an order.
- ✓ When we use read command, it move ahead pointer by one
- ✓ When we use write command, it will allocate memory and move the pointer to the end of the file
- ✓ Such a method is reasonable for tape\

.

2. **Direct Access**

Another method is direct access method also known as relative access method. A filed-length logical record that allows the program to read and write record rapidly. in no particular order. The direct access is based on the disk model of a file since disk allows random access to any file block. For direct access, the file is viewed as a numbered sequence of block or record. Thus, we may read block 14 then block 59 and then we can write block 17. There is no restriction on the order of reading and writing for a direct access file.

A block number provided by the user to the operating system is normally a relative block number; the first relative block of the file is 0 and then 1 and so on.

3. **Index sequential method**

It is the other method of accessing a file which is built on the top of the direct access method. These methods construct an index for the file. The index, like an index in the back of a book, contains the pointer to the various blocks. To find a record in the file, we first search the index and then by the help of pointer we access the file directly.

Key points:

- ✓ It is built on top of Sequential access.
- ✓ It controls the pointer by using index.

PART-C (10 Marks)

1. Explain in detail about Device Management with its types.

Device management involves four basic functions:

- Monitoring the status of each device, such as storage drives, printers, and other peripheral devices
- Enforcing preset policies to determine which process will get a device and for how long
- Allocating the devices
- Deallocating them at two levels—at the process (or task) level when an I/O command has been executed and the device is temporarily released, and then at the job level when the job is finished and the device is permanently released

Types of Devices :

The system's peripheral devices generally fall into one of three categories:

- Dedicated,
- Shared and
- Virtual.

The differences are a function of the characteristics of the devices, as well as how they're managed by the Device Manager.

Dedicated devices are assigned to only one job at a time; they serve that job for the entire time it's active or until it releases them. Some devices, such as tape drives, printers, and plotters, demand this kind of allocation scheme, because it would be awkward to let several users share them. A shared plotter might produce half of one user's graph and half of another.

The disadvantage of dedicated devices is that they must be allocated to a single user for the duration of a job's execution, which can be quite inefficient, especially when the device isn't used 100 percent of the time.

And some devices can be shared or virtual. Shared devices can be assigned to several processes. For instance, a disk, or any other direct access storage device (often shortened to DASD), can be shared by several processes at the same time by interleaving their requests, but this interleaving must be carefully controlled by the Device Manager.

All conflicts—such as when Process A and Process B each need to read from the same disk—must be resolved based on predetermined policies to decide which request will be handled first. Virtual devices are a combination of the first two: They're dedicated devices that have been transformed into shared devices. For example, printers (which are dedicated devices) are converted into sharable devices through a spooling program that reroutes all print requests to a disk. Only when all of a job's output is complete, and the printer is ready to print out the entire document, is the output sent to the printer for printing

1. Sequential Access Storage Media:

Magnetic tape was developed for routine secondary storage in early computer systems and features records that are stored serially, one after the other. The length of these records is usually determined by the application program and each record can be identified by its position on the tape. Therefore, to access a single record, the tape must be mounted and fast-forwarded from its beginning until the desired position is located. This can be a time-consuming process. To appreciate just how long it takes, let's consider a hypothetical computer system that uses a reel of tape that is 2400 feet long (see Figure 1.1). Data is recorded on eight of the nine parallel tracks that run the length of the tape. (The ninth track, shown at the top of the figure, holds a parity bit that is used for routine error checking.)

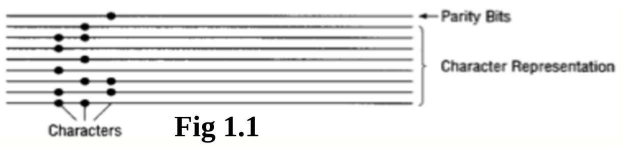


Fig 1.1

The number of characters that can be recorded per inch is determined by the density of the tape, such as 1600 bytes per inch (bpi). For example, if you had records of 160 characters each, and were storing them on a tape with a density of 1600 bpi, then theoretically you could store 10 records on one inch of tape. However, in actual practice, it would depend on how you decided to store the records: individually or grouped into blocks. If the records are stored individually, each record would need to be separated by a space to indicate its starting and ending places. If the records are stored in blocks, then the entire block is preceded by a space and followed by a space, but the individual records are stored sequentially within the block.

Magnetic tape moves under the read/write head only when there's a need to access a record; at all other times it's standing still. So the tape moves in jerks as it stops, reads, and moves on at high speed, or stops, writes, and starts again, and so on. Records would be written in the same way. The tape needs time and space to stop, so a gap is inserted between each record. This interrecord gap (IRG) is about 1⁄2 inch long regardless of the sizes of the records it separates.

An alternative is to group the records into blocks before recording them on tape. This is called blocking and it's performed when the file is created. (Later, when you retrieve them, you must be sure to unblock them accurately.) The number of records in a block is usually determined by the application program, and it's often set to take advantage of the transfer rate, which is the density of the tape (measured in bpi), multiplied by the tape drive speed, called transport speed, which is measured in inches per second (ips): **transfer rate (ips) = density * transport speed**

Blocking has two distinct advantages:

• Fewer I/O operations are needed because a single READ command can move an entire block, the physical record that includes several logical records, into main memory.

• Less tape is wasted because the size of the physical record exceeds the size of the gap.
The two disadvantages of blocking are

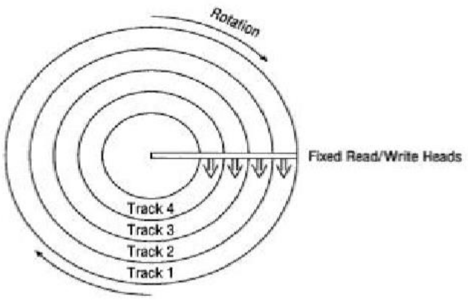
• Overhead and software routines are needed for blocking, deblocking, and recordkeeping.
• Buffer space may be wasted if you need only one logical record but must read an entire block to get it.

2. **Direct Access Storage Devices :**

Direct access storage devices (DASDs) are devices that can directly read or write to a specific place. DASDs can be grouped into three categories: magnetic disks, optical discs, and flash memory. Although the variance in DASD access times isn't as wide as with magnetic tape, the location of the specific record still has a direct effect on the amount of time required to access it.

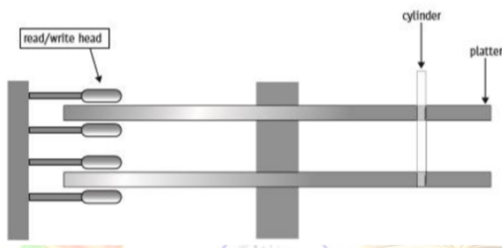
2.1 Fixed-Head Magnetic Disk Storage:

A fixed-head magnetic disk looks like a large CD or DVD covered with magnetic film that has been formatted, usually on both sides, into concentric circles. Each circle is a track. Data is recorded serially on each track by the fixed read/write head positioned over it. A fixed-head disk, shown in below, is also very fast—faster than the movable-head disks we'll talk about in the next section. However, its major disadvantages are its high cost and its reduced storage space compared to a movable-head disk (because the tracks must be positioned farther apart to accommodate the width of the read/write heads). These devices have been used when speed is of the utmost importance, such as space flight or aircraft applications.



2.2 Movable-Head Magnetic Disk Storage:

Movable-head magnetic disks, such as computer hard drives, have one read/write head that floats over each surface of each disk. Disks can be a single platter, or part of a disk pack, which is a stack of magnetic platters. Figure shows a typical disk pack—several platters stacked on a common central spindle, separated by enough space to allow the read/write heads to move between each pair of disks. As shown in Figure, each platter (except those at the top and bottom of the stack) has two surfaces for recording, and each surface is formatted with a specific number of concentric tracks where the data is recorded. The number of tracks varies from manufacturer to manufacturer, but typically there are a thousand or more on a high capacity hard disk. Each track on each surface is numbered: Track 0 identifies the outermost concentric circle on each surface; the highest-numbered track is in the centre.



2.3 Optical Disc Storage:

The advent of optical disc storage was made possible by developments in laser technology. Among the many differences between an optical disc and a magnetic disk is the design of the disc track and sectors. A magnetic disk, which consists of concentric tracks of sectors, spins at a constant speed—this is called constant angular velocity (CAV). Because the sectors at the outside of the disk spin faster past the read/write head than the inner sectors, outside sectors are much larger than sectors located near the center of the disk. This format wastes storage space but maximizes the speed with which data can be retrieved.

2.4 CD and DVD Technology:

In the CD or DVD player, data is read back by focusing a low powered red laser on it, which shines through the protective layer of the disc onto the CD track (or DVD tracks) where data is recorded. Light striking a land is reflected into a photo detector while light striking a pit is scattered and absorbed. The photo detector then converts the intensity of the light into a digital signal of 1s and 0s. Recordable CD and DVD disc drives require more expensive disc controllers than the read-only disc players because they need to incorporate write mechanisms specific to each medium. For example, a CD consists of several layers, including a gold reflective layer and a dye layer, which is used to record the data. The write head uses a high-powered laser beam to record data. A permanent mark is made on the dye when the energy from the laser beam is absorbed into it and it cannot be erased after it is recorded. When it is read, the existence of a mark on the dye will cause the laser beam to scatter and light is not returned back to the read head. However, when there are no marks on the dye, the gold layer reflects the light right back to the read head. This is similar to the process of reading pits and lands. The software used to create a recordable CD (CD-R) uses a standard format, such as ISO 9096, which automatically checks for errors and creates a table of contents, used to keep track of each file's location.

2. Describe about RAID and its Levels in detail.

RAID, or "Redundant Arrays of Independent Disks" is a technique which makes use of a combination of multiple disks instead of using a single disk for increased performance, data redundancy or both. The term was coined by David Patterson, Garth A. Gibson, and Randy Katz at the University of California, Berkeley in 1987.

Data Redundancy:

Data redundancy, although taking up extra space, adds to disk reliability. This means, in case of disk failure, if the same data is also backed up onto another disk, we can retrieve the data and go on with the operation. On the other hand, if the data is spread across just multiple disks without the RAID technique, the loss of a single disk can affect the entire data.

Key evaluation points for a RAID System

- **Reliability:** How many disk faults can the system tolerate?
- **Availability:** What fraction of the total session time is a system in uptime mode, i.e. how available is the system for actual use?
- **Performance:** How good is the response time? How high is the throughput (rate of processing work)? Note that performance contains a lot of parameters and not just the two.
- **Capacity:** Given a set of N disks each with B blocks, how much useful capacity is available to the user?

RAID is very transparent to the underlying system. This means, to the host system, it appears as a single big disk presenting itself as a linear array of blocks. This allows older technologies to be replaced by RAID without making too many changes in the existing code.

Different RAID levels

RAID-0 (Striping)

- Blocks are "striped" across disks.

Disk 0	Disk 1	Disk 2	Disk 3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

In the figure, blocks “0,1,2,3” form a stripe.

- Instead of placing just one block into a disk at a time, we can work with two (or more) blocks placed into a disk before moving on to the next one.

Disk 0	Disk 1	Disk 2	Disk 3
0	3	4	6
1	3	5	7
8	10	12	14
9	11	13	15

Evaluation:

- Reliability: 0
  There is no duplication of data. Hence, a block once lost cannot be recovered.
- Capacity: N*B
  The entire space is being used to store data. Since there is no duplication, N disks each having B blocks are fully utilized.

RAID-1 (Mirroring)

More than one copy of each block is stored in a separate disk. Thus, every block has two (or more) copies, lying on different disks.

Disk 0	Disk 1	Disk 2	Disk 3
0	0	1	1
2	2	3	3
4	4	5	5
6	6	7	7

The above figure shows a RAID-1 system with mirroring level 2.

- RAID 0 was unable to tolerate any disk failure. But RAID 1 is capable of reliability.

Evaluation:

Assume a RAID system with mirroring level 2.

- Reliability: 1 to N/2
  1 disk failure can be handled for certain, because blocks of that disk would have duplicates on some other disk. If we are lucky enough and disks 0 and 2 fail, then again

this can be handled as the blocks of these disks have duplicates on disks 1 and 3. So, in the best case, N/2 disk failures can be handled.

- Capacity: N*B/2

  Only half the space is being used to store data. The other half is just a mirror to the already stored data.

RAID-4 (Block-Level Striping with Dedicated Parity)

- Instead of duplicating data, this adopts a parity-based approach.

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
4	5	6	7	P1
8	9	10	11	P2
12	13	14	15	P3

In the figure, we can observe one column (disk) dedicated to parity.

- Parity is calculated using a simple XOR function. If the data bits are 0,0,0,1 the parity bit is XOR(0,0,0,1) = 1. If the data bits are 0,1,1,0 the parity bit is XOR(0,1,1,0) = 0. A simple approach is that even number of ones results in parity 0, and an odd number of ones results in parity 1.

C1	C2	C3	C4	Parity
0	0	0	1	1
0	1	1	0	0

Assume that in the above figure, C3 is lost due to some disk failure. Then, we can recompute the data bit stored in C3 by looking at the values of all the other columns and the parity bit. This allows us to recover lost data.

Evaluation:

- Reliability: 1

  RAID-4 allows recovery of at most 1 disk failure (because of the way parity works). If more than one disk fails, there is no way to recover the data.
- Capacity: (N-1)*B

  One disk in the system is reserved for storing the parity. Hence, (N-1) disks are made available for data storage, each disk having B blocks.

RAID-5 (Block-Level Striping with Distributed Parity)

- This is a slight modification of the RAID-4 system where the only difference is that the parity rotates among the drives.

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
5	6	7	P1	4
10	11	P2	8	9
15	P3	12	13	14
P4	16	17	18	19

In the figure, we can notice how the parity bit "rotates".

- This was introduced to make the random write performance better.

Evaluation:

- Reliability: 1

  RAID-5 allows recovery of at most 1 disk failure (because of the way parity works). If more than one disk fails, there is no way to recover the data. This is identical to RAID-4.
- Capacity: (N-1)*B

  Overall, space equivalent to one disk is utilized in storing the parity. Hence, (N-1) disks are made available for data storage, each disk having B blocks.

Other RAID levels:

- RAID-2 consists of bit-level striping using a Hamming Code parity. RAID-3 consists of byte-level striping with a dedicated parity. These two are less commonly used.
- RAID-6 is a recent advancement which contains a distributed double parity, which involves block-level striping with 2 parity bits instead of just 1 distributed across all the disks. There are also hybrids RAIDs, which make use of more than one RAID levels nested one after the other, to fulfil specific requirements.

3. Describe about Physical Storage allocation in File System .

The allocation methods define how the files are stored in the disk blocks. There are three main disk space or file allocation methods.

- Contiguous Allocation
- Linked Allocation
- Indexed Allocation

The main idea behind these methods is to provide:

- Efficient disk space utilization.
- Fast access to the file blocks.

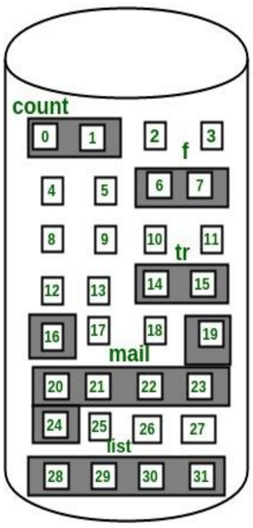
All the three methods have their own advantages and disadvantages as discussed below:

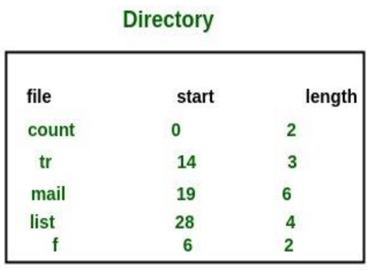
1. Contiguous Allocation

In this scheme, each file occupies a contiguous set of blocks on the disk. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: *b, b+1, b+2,……b+n-1.* This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file. The directory entry for a file with contiguous allocation contains

- Address of starting block
- Length of the allocated portion.

The *file 'mail'* in the following figure starts from the block 19 with length = 6 blocks. Therefore, it occupies *19, 20, 21, 22, 23, 24* blocks.




Directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Advantages:

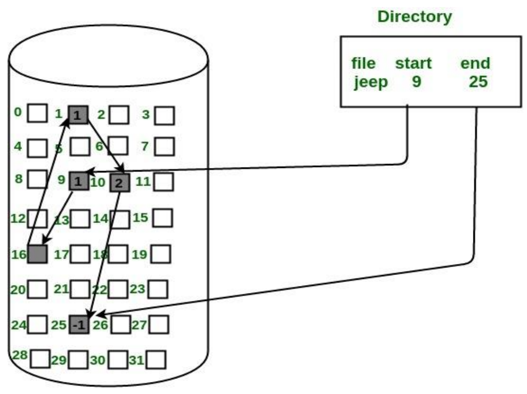
- Both the Sequential and Direct Accesses are supported by this. For direct access, the address of the kth block of the file which starts at block b can easily be obtained as (b+k).
- This is extremely fast since the number of seeks are minimal because of contiguous allocation of file blocks.

Disadvantages:

- This method suffers from both internal and external fragmentation. This makes it inefficient in terms of memory utilization.
- Increasing file size is difficult because it depends on the availability of contiguous memory at a particular instance.

2. Linked List Allocation

In this scheme, each file is a linked list of disk blocks which need not be contiguous. The disk blocks can be scattered anywhere on the disk.The directory entry contains a pointer to the starting and the ending file block. Each block contains a pointer to the next block occupied by the file.The file 'jeep' in following image shows how the blocks are randomly distributed. The last block (25) contains -1 indicating a null pointer and does not point to any other block.



Advantages:

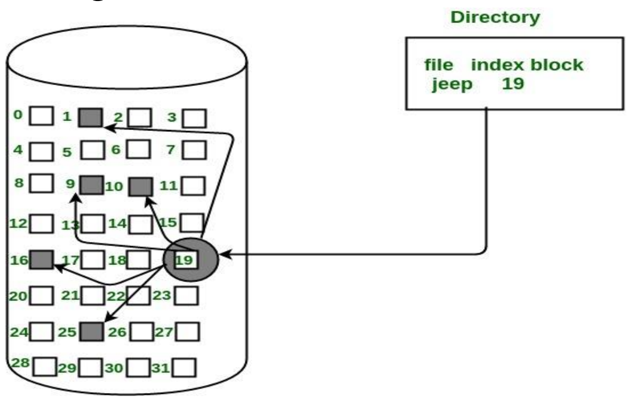
- This is very flexible in terms of file size. File size can be increased easily since the system does not have to look for a contiguous chunk of memory.
- This method does not suffer from external fragmentation. This makes it relatively better in terms of memory utilization.

Disadvantages:

- Because the file blocks are distributed randomly on the disk, a large number of seeks are needed to access every block individually. This makes linked allocation slower.
- It does not support random or direct access. We can not directly access the blocks of a file. A block k of a file can be accessed by traversing k blocks sequentially (sequential access) from the starting block of the file via block pointers.
- Pointers required in the linked allocation incur some extra overhead.

3. Indexed Allocation

In this scheme, a special block known as the **Index block** contains the pointers to all the blocks occupied by a file. Each file has its own index block. The ith entry in the index block contains the disk address of the ith file block. The directory entry contains the address of the index block as shown in the image:



Advantages:

- This supports direct access to the blocks occupied by the file and therefore provides fast access to the file blocks.
- It overcomes the problem of external fragmentation.

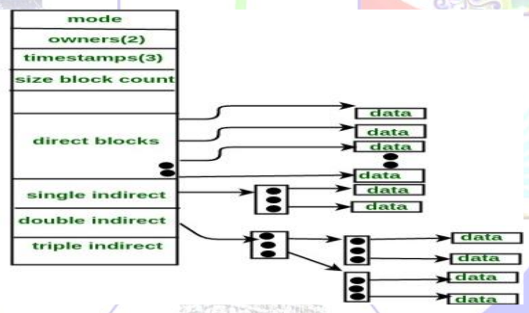
Disadvantages:

- The pointer overhead for indexed allocation is greater than linked allocation.
- For very small files, say files that expand only 2-3 blocks, the indexed allocation would keep one entire block (index block) for the pointers which is inefficient in terms of

memory utilization. However, in linked allocation we lose the space of only 1 pointer per block.

For files that are very large, single index block may not be able to hold all the pointers. Following mechanisms can be used to resolve this:

1. **Linked scheme:** This scheme links two or more index blocks together for holding the pointers. Every index block would then contain a pointer or the address to the next index block.
2. **Multilevel index:** In this policy, a first level index block is used to point to the second level index blocks which inturn points to the disk blocks occupied by the file. This can be extended to 3 or more levels depending on the maximum file size.
3. **Combined Scheme:** In this scheme, a special block called the **Inode (information Node)** contains all the information about the file such as the name, size, authority, etc and the remaining space of Inode is used to store the Disk Block addresses which contain the actual file *as shown in the image below*. The first few of these pointers in Inode point to the **direct blocks** i.e the pointers contain the addresses of the disk blocks that contain data of the file. The next few pointers point to indirect blocks. Indirect blocks may be single indirect, double indirect or triple indirect. **Single Indirect block** is the disk block that does not contain the file data but the disk address of the blocks that contain the file data. Similarly, double indirect blocks do not contain the file data but the disk address of the blocks that contain the address of the blocks containing the file data.



4. Describe about Access control verification Module.

The first operating systems couldn't support file sharing among users. For instance, early systems needed 10 copies of a compiler to serve 10 users. Today's systems require only a single copy to serve everyone, regardless of the number of active programs in the system. In fact, any file can be shared—from data files and user-owned program files to system files. The advantages of file sharing are numerous. In addition to saving space, it allows for synchronization of data updates, as when two applications are updating the same data file. It also improves the efficiency of the system's resources because if files are shared in main memory, then there's a reduction of I/O operations. However, as often happens, progress brings problems. The disadvantage of file sharing is that the integrity of each file must be safeguarded; that calls for control over who is allowed to access the file and what type of access is permitted. There are five possible actions that can be performed on a file—the ability to READ only, WRITE only, EXECUTE only, DELETE only, or some combination of the four. Each file management system has its own method to control file access.

Access Control Matrix

The access control matrix is intuitively appealing and easy to implement, but because of its size it only works well for systems with a few files and a few users. In the matrix, each column identifies a user and each row identifies a file. The intersection of the row and column contains the access rights for that user to that file, as below Table illustrates

(table 8.2)

The access control matrix showing access rights for each user for each file. User 1 is allowed unlimited access to File 1 but is allowed only to read and execute File 4 and is denied access to the three other files. R = Read Access, W = Write Access, E = Execute Access, D = Delete Access, and a dash (-) = Access Not Allowed.

	User 1	User 2	User 3	User 4	User 5
File 1	RWED	R-E-	----	RWE-	--E-
File 2	----	R-E-	R-E-	--E-	----
File 3	----	RWED	----	--E-	----
File 4	R-E-	----	----	----	RWED
File 5	----	----	----	----	RWED

In the actual implementation, the letters RWED are represented by bits 1 and 0: a 1 indicates that access is allowed, and a 0 indicates access is denied. Therefore, as shown in Table below , the code for User 2 for File 1 would read “1010” and not “RE”.

(table 8.3)

The five access codes for User 2 from Table 8.2. The resulting code for each file is created by assigning a 1 for each checkmark, and a 0 for each blank space.

Access	R	W	E	D	Resulting Code
R-E-	✓		✓		1010
R-E-	✓		✓		1010
RWED	✓	✓	✓	✓	1111
----					0000
----					0000

As you can see, the access control matrix is a simple method; but as the numbers of files and users increase, the matrix becomes extremely large—sometimes too large to store in main memory. Another disadvantage is that a lot of space is wasted because many of the entries are all null, such as in Table-1, where User 3 isn’t allowed into most of the files, and File 5 is restricted to all but one user. A scheme that conserved space would have only one entry for User 3 or one for File 5, but that’s incompatible with the matrix format.

Access Control Lists

The access control list is a modification of the access control matrix. Each file is entered in the list and contains the names of the users who are allowed to access it and the type of access each is permitted. To shorten the list, only those who may use the file are named; those denied any access are grouped under a global

(table 8.4)

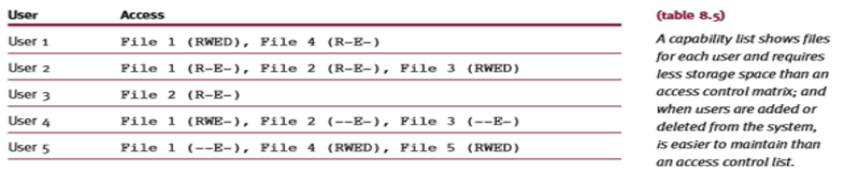
An access control list showing which users are allowed to access each file. This method requires less storage space than an access control matrix.

File	Access
File 1	USER1 (RWED), USER2 (R-E-), USER4 (RWE-), USER5 (--E-), WORLD (----)
File 2	USER2 (R-E-), USER3 (R-E-), USER4 (--E-), WORLD (----)
File 3	USER2 (RWED), USER4 (--E-), WORLD (----)
File 4	USER1 (R-E-), USER5 (RWED), WORLD (----)
File 5	USER5 (RWED), WORLD (----)

Some systems shorten the access control list even more by putting every user into a category: system, owner, group, and world. SYSTEM or ADMIN is designated for system personnel who have unlimited access to all files in the system. The OWNER has absolute control over all files created in the owner’s account. An owner may create a GROUP file so that all users belonging to the appropriate group have access to it. WORLD is composed of all other users in the system; that is, those who don't fall into any of the other three categories. In this system, the File Manager designates default types of access to all files at creation time, and it's the owner's responsibility to change them as needed.

Capability Lists:

A capability list shows the access control information from a different perspective. It lists every user and the files to which each has access, as shown in Table..



User	Access
User 1	File 1 (RWED), File 4 (R-E-)
User 2	File 1 (R-E-), File 2 (R-E-), File 3 (RWED)
User 3	File 2 (R-E-)
User 4	File 1 (RWE-), File 2 (--E-), File 3 (--E-)
User 5	File 1 (--E-), File 4 (RWED), File 5 (RWED)

(table 8.5)

A capability list shows files for each user and requires less storage space than an access control matrix; and when users are added or deleted from the system, is easier to maintain than an access control list.

Of the three schemes described so far, the most commonly used is the access control list. However, capability lists are gaining in popularity because in operating systems such as Linux or UNIX, they control access to devices as well as to files. Although both methods seem to be the same, there are some subtle differences best explained with an analogy. A capability list may be equated to specific concert tickets that are made available to only individuals whose names appear on the list. On the other hand, an access control list can be equated to the reservation list in a restaurant that has limited seating, with each seat assigned to a certain individual.