AKTU

B. Tech II-Year

Gateway Classes

Semester -IV

CS IT & Allied Branches

BCS401-Operating system

ONE SHOT UNIT-5

I/O Management & Disk Scheduling

Gateway Series for Engineering

- Topic Wise Entire Syllabus
- Long - Short Questions Covered
- AKTU PYQs Covered
- DPP
- Result Oriented Content

Download App

For Full Courses including Video Lectures

AKTU

B. Tech II-Year

Gateway Classes

BCS401-Operating system

ONE SHOT Unit-5

Introduction to I/O Management & Disk Scheduling

Syallbus

I/O Management and Disk Scheduling: I/O devices, and I/O subsystems, I/O buffering, Disk storage and disk scheduling, RAID. File System: File concept, File organization and access mechanism, File directories, and File sharing, File system implementation issues, File system protection and security

Download App

For Full Courses including Video Lectures

Operating system

GATEWAY CLASSES

ONE SHOT

Unit-5

I/O Management and Disk Scheduling

CS / IT / CS Allied / MCA

By Dr. Kapil Kumar Sir

OPERATING SYSTEM

Syllabus Unit - 5

I/O Management and Disk Scheduling: I/O devices, and I/O subsystems, I/O buffering, Disk storage and disk scheduling, RAID.

File System: File concept, File organization and access mechanism, File directories, and File sharing, File system implementation issues, File system protection and security.

Input and Output Devices

- **Input Devices:**
  - **Keyboard** - Used for typing text and commands.
  - **Mouse** - A pointing device used to interact with graphical user interfaces.
  - **Scanner** - Converts physical documents and images into digital form.
  - **Microphone** - Captures audio input.
  - **Camera** - Captures still images or video.
- **Output Devices:**
  - **Monitor** - Displays visual output from the computer.
  - **Printer** - Produces a physical copy of digital documents.
  - **Speakers** - Output sound from the computer.
  - **Projector** - Projects visual output onto a larger screen or surface.
  - **Headphones** - Output audio in a personal and portable manner.

Input - output Devices

- Some devices can both input data to and output data from a computer system.
- Examples include:
- Touchscreen - Acts as both an input device (when touched) and an output device (by displaying visual information).
- External Hard Drives / USB Drives - Can be used to both read data from and write data to the computer.
- Network Cards - Facilitate data transmission to and from other computers over a network.
- Modems - Convert digital data to analog for transmission over phone lines and vice versa.

Input/output Subsystems

- I/O subsystems manages the communication between the CPU and its peripheral devices.
- These subsystems ensure efficient data transfer and handling between different parts of the computer system, such as input devices, output devices, storage devices, and network interfaces.
- **Key Components of I/O Subsystems:**
- **Device Controllers:**
  - **Hardware interfaces** between the CPU and the peripheral devices.
  - Manage the specific details of device operations.
- **Device Drivers:**
  - **Software programs** that allow the operating system to communicate with hardware devices.
  - Translate high-level commands from the Operating System into device-specific operations.

❑ **I/O Ports:**

- Interfaces through which the CPU communicates with peripheral devices.
- Can be parallel or serial, depending on the data transfer method.

❑ **DMA (Direct Memory Access) Controllers:**

- Allow peripheral devices to directly transfer data to/from memory without involving the CPU.
- Improve system efficiency by freeing up the CPU for other tasks during data transfer.

❑ **Buses:**

- Communication pathways that connect the CPU, memory, and peripheral devices.
- Examples include the system bus, PCI (Peripheral Component Interconnect), and USB (Universal Serial Bus).

GW GATEWAY CLASSES

Concepts Related to I/O Subsystems

Interrupts:

- Signals that alert the CPU to an event that requires immediate attention.
- Can be hardware interrupts (from a device) or software interrupts (from a program).

Buffering:

- Temporary storage areas (buffers) that hold data being transferred between the CPU and peripheral devices.
- Helps accommodate differences in data transfer rates between devices.

Spooling:

- A technique used to manage data by placing it in a temporary storage area (the spool) before sending it to a device.
- Commonly used in printing and batch processing systems.

Caching:

- Temporary storage of frequently accessed data to improve access speed.
- Used in both hardware (e.g., disk caches) and software (e.g., web browser caches).

I/O Buffering

- I/O buffering is a technique used in computer systems to temporarily store data while it is being transferred between two devices or between a device and an application.
- This temporary storage area is known as a buffer.
- Buffers help manage the speed differences between devices, smooth out data flow, and improve overall system performance and efficiency.
- **Purposes of I/O Buffering**

1. Handling Speed Mismatches:

- Different devices often operate at different speeds.
- For example, a hard drive may transfer data slower than the CPU can process it.
- Buffers help manage these speed mismatches by storing data temporarily until the slower device is ready to process it.

2. Data Flow Smoothing:

- Buffers smooth out the data flow, preventing interruptions or delays that might occur if the system had to wait for each individual piece of data to be transferred immediately.

3. Reducing Latency:

- By allowing data to be read or written in larger chunks, buffers reduce the latency associated with frequent small data transfers.

4. Improving Throughput:

- Buffers can improve overall system throughput by allowing the CPU and I/O devices to work more efficiently without being constantly interrupted.

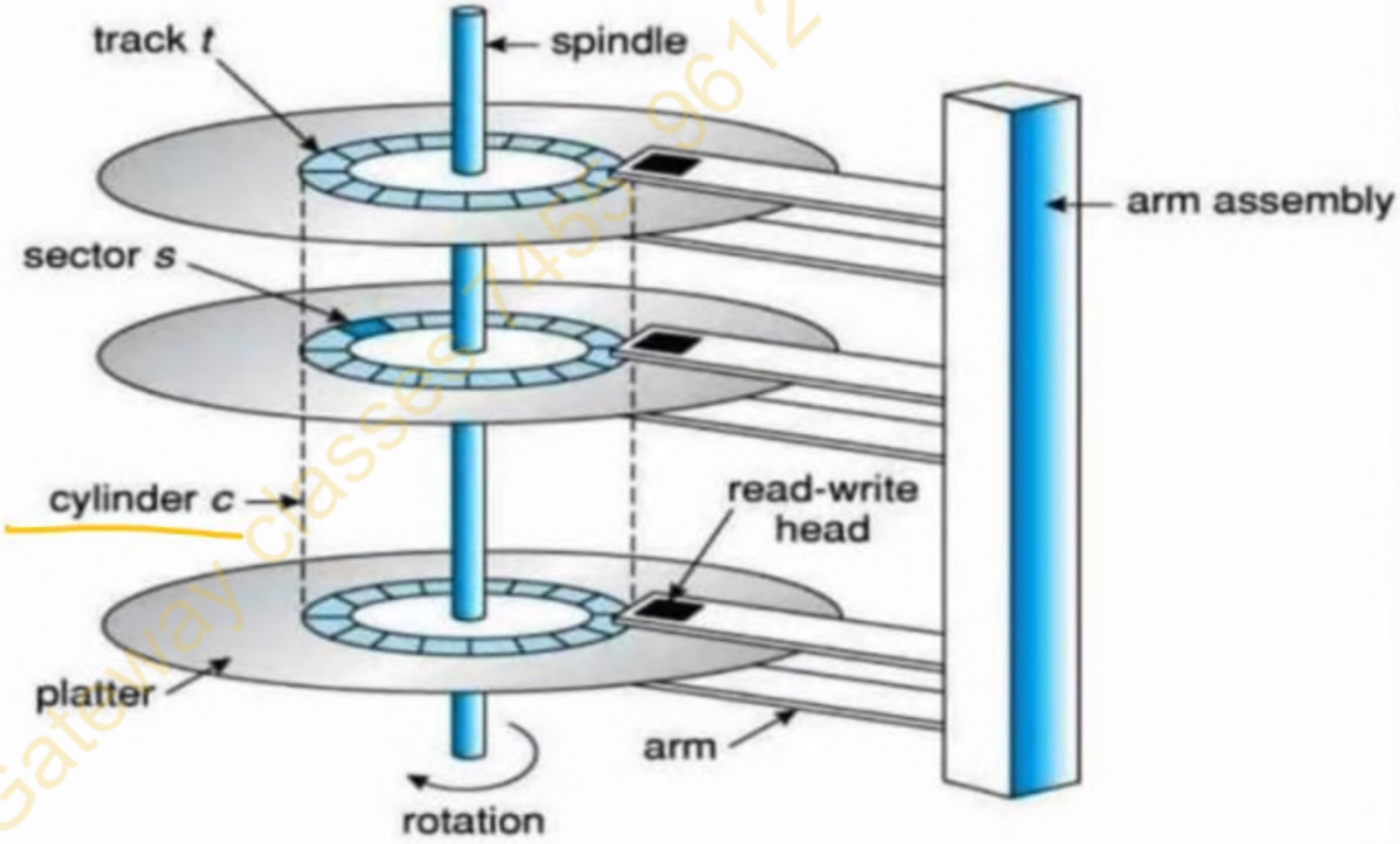
Advantages of I/O Buffering

- **Efficiency:** Reduces the number of times the CPU is interrupted, allowing it to perform more tasks concurrently.
- **Speed:** Manages speed differences between devices, allowing faster devices to operate without being held up by slower ones.
- **Resource Utilization:** Optimizes the use of system resources by smoothing out data transfer peaks and troughs.
- **Reliability:** Helps in error handling and recovery by providing a buffer zone that can be used to retry data transfer in case of errors.

Disk and Disk Scheduling

- A disk is a type of storage device used to store data permanently.
- Disks can be hard disk drives (HDDs) or solid-state drives (SSDs).
- Disk scheduling is done by operating systems to schedule I/o requests arriving for the disk.
- **Disk scheduling is important because:**
  - Multiple I/o requests may arrive by different processes and only one I/o request can be served at a time by the disk controller.
  - Thus other I/o requests need to wait in the waiting queue and need to be scheduled.
  - Hard drives are one of the slowest parts of the computer system and thus need to be accursed in an efficient manner.

Disk



- There are many disk scheduling algorithms but first lets discuss the following important terms:

- **Seek Time**

- It is the time taken by the disk arm (read – write head) to locate the desired track.
- Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or write.
- So the disk scheduling algorithm that gives minimum average seek time is better.

- **Rotational Latency**

- Rotational Latency is the time taken to reach to desired sector.
- So the disk scheduling algorithm that gives minimum rotational latency is better.

- **Cylinder**

- Collection of tracks of same radius from all surfaces.
- Content in disk stored cylinder wise so that seek time can be saved.

Disk Scheduling

- Disk scheduling in operating systems refers to the process of deciding the order in which I/O requests are serviced by a disk drive.
- When multiple processes or tasks request disk access simultaneously, the operating system's disk scheduler determines the most efficient way to handle these requests to minimize access time and optimize overall system performance.
- Disk scheduling algorithms aim to reduce seek time, latency, and maximize disk throughput by organizing and prioritizing I/O operations effectively.
- Some common disk scheduling algorithms include FCFS (First-Come, First-Served), SSTF (Shortest Seek Time First), SCAN, C-SCAN, LOOK, and C-LOOK, each with its own approach to prioritizing and ordering I/O requests.

FCFS - Disk Scheduling

- In FCFS, the requests are addressed in the order they arrive in the disk queue.

Example 1: Consider a disk queue with requests for I/O to blocks on cylinder –

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

200 cylinders

Head starts at 53. Find the total head movement. (ans=640)

Seek time

Solution:-

QUEUE - 98, 183, 37, 122, 14, 124, 65, 67

FIFO = ?

Head = 53

Total Head Movement (or total seek time) –

98 – 53 = 45

183 – 98 = 85

183 – 37 = 146

122 – 37 = 85

122 – 14 = 108

124 – 14 = 110

124 – 65 = 59

67 – 65 = 2

Total head movement = 640

Advantages -

- Every request gets a fair chance.
- No indefinite postponement.

Disadvantages -

- Do not try to optimize seek time.
- May not provide the best possible services.

GW
GATEWAY CLASSES

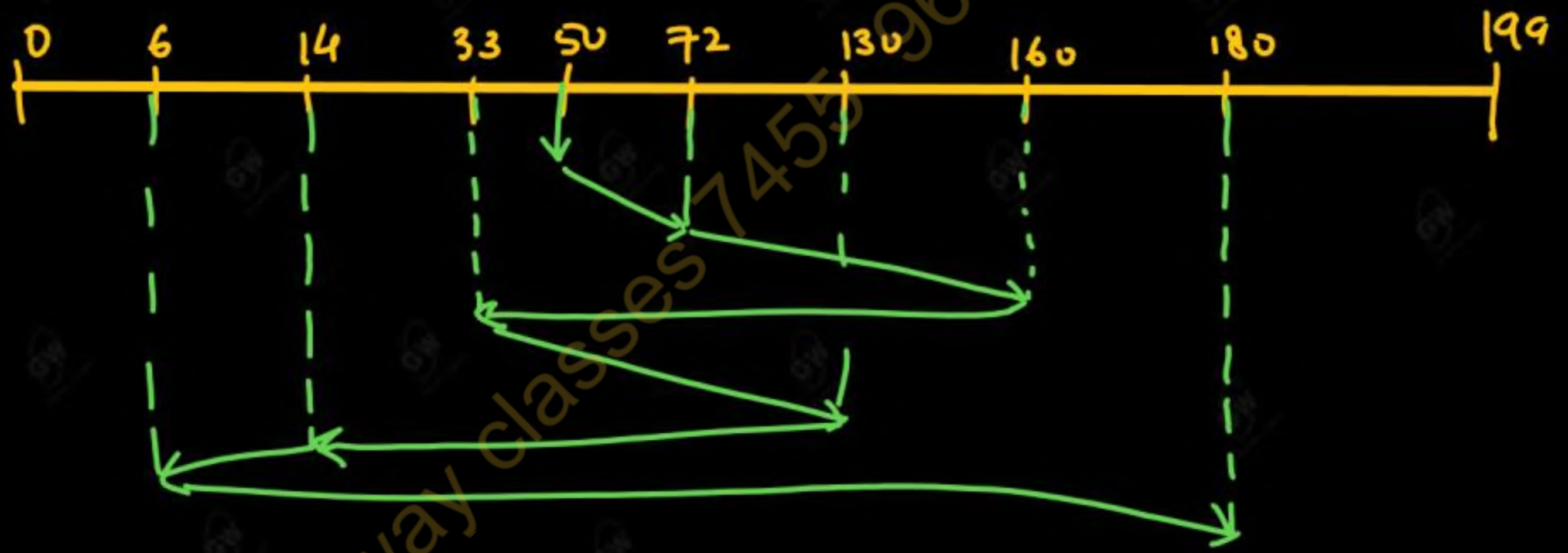
Example 2: Consider a disk queue with requests for I/O to blocks on cylinder –

Queue : 72, 160, 33, 130, 14, 6 ,180

Total number of cylinders are 200. The current head position is 50. Find the total head movement.

(ans=632)

Solution:-



Total Head Movement (or total seek time) –

72 - 50 = 22 ✓

160 – 72 = 88 ✓

160 – 33 = 127 ✓

130 – 33 = 97 ✓

130 - 14 = 116 ✓

14 ~~130~~ – 6 = ~~124~~ 8 ✓

180 – 6 = 174 ✓

Total head movement = 632

Ans

QUEUE – 72, 160, 33, 130, 14, 6, 180

CURRENT HEAD POSITION = 50

No of Cylinders = 200

FIFO = ?

AKTU PYQs

Q.1. A hard disk having 2000 cylinders, numbered from 0 to 1999. The drive is currently serving the request at cylinder 143, and the previous request was at cylinder 125. The status of the queue is as follows :

86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

What is the total distance (in cylinders) that the disk arm moves to satisfy the entire pending request for each of the following disk-scheduling algorithms?

(i) SSTF (ii) FCFS

2021-22, 2018-19, 10 Marks

Q.2. Write short notes on :

(i) I/O Buffering

(ii) Disk storage and scheduling

2017-18, 7 Marks

Q.3. Suppose the moving head disk with 200 tracks is currently serving a request for track 143 and has just finished a request for track 125. If the queue of request is kept in FIFO order

86, 147, 91, 177, 94, 150.

What is total head movement for the following scheduling:

(i) FCFS (ii) SSTF (iii) C-SCAN 2015-16, 10 Marks

Q.4. Write short notes on:

(i) I/O Buffering (ii) Sequential File (iii) Indexed File 2015-16, 10 Marks

Q.5. Discuss disk scheduling algorithms with example. 2014-15, 10 Marks

SSTF (Shortest Seek Time First) Disk Scheduling Algorithm

- The SSTF algorithm selects the request with the least seek time from the current head position.
- So, the seek time of every request is calculated in advance in the queue and they are scheduled according to their calculated seek time.
- As a result, the request near the disk arm will get executed first.
- SSTF is certainly an improvement over FCFS as it decreases the average response time and increases the throughput of system.

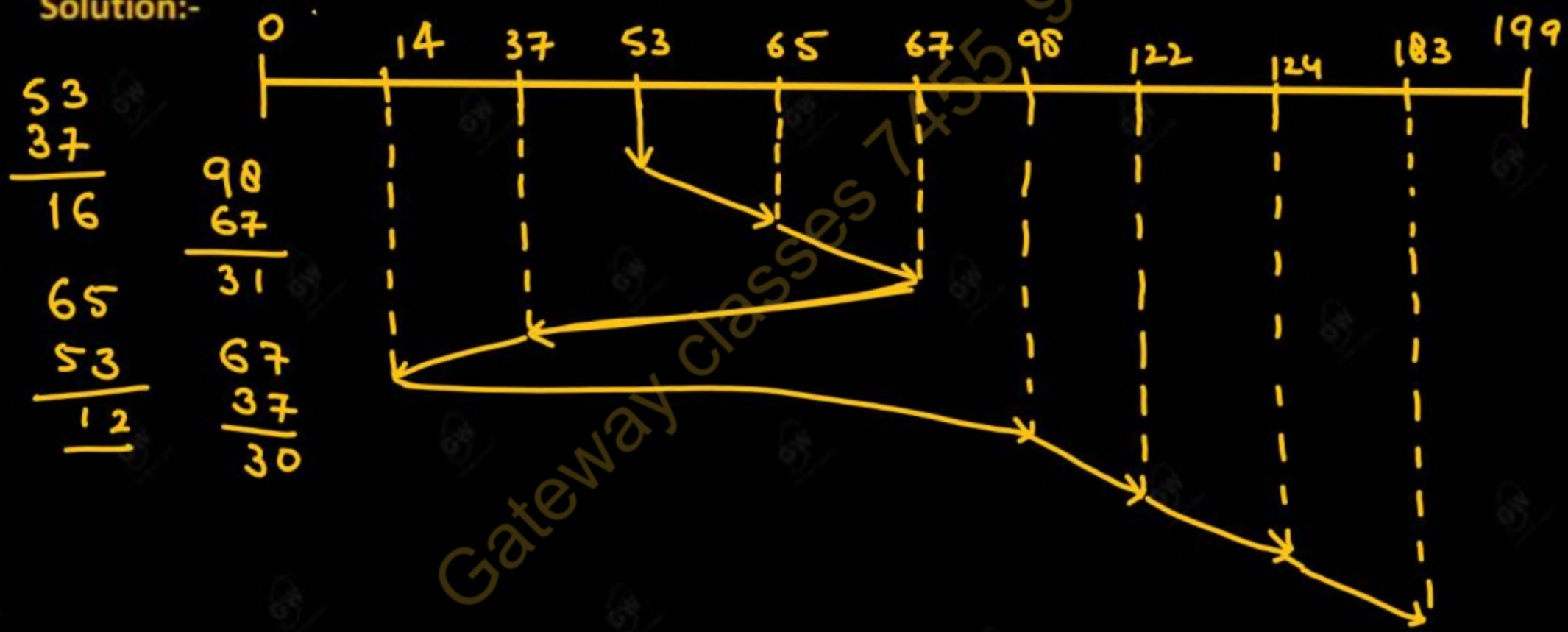
Example 1: Consider a disk queue with requests for I/O to blocks on cylinder –

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

The current position of the r/w head is 53. Find the total head movement using SSTF disk scheduling algorithm.

Assume no of cylinders are 200

Solution:-



- Total R/W Head Movements are -

67 – 53 = 14

67 – 14 = 53

183 – 14 = 169

Total Head Movements = 236 Ans

Advantages of SSTF -

- Average response time decreases.

Disadvantages of SSTF -

- Overhead to calculate seek time in advance.
- Can cause starvation for a request it if has higher seek time as compared to incoming requests.

SSTF QUEUE = 98, 183, 37, 122, 14, 124, 65, 67

Current Head Position = 53

Example 2: Consider a disk queue with requests for I/O to blocks on cylinder –

Queue : 72, 160, 33, 130, 14, 6, 180

The number of cylinders are 200. The read/write had current position is 50. Find the total head movements using SSTF disk scheduling algorithm.

Solution:-

72
50
22

50
33
17

Amp

Total Head movements = (50 – 6) + (180 – 6) = 218 (44 + 174)

SSTF QUEUE – 72, 160, 33, 130, 14, 6, 180

No. of cylinders = 200, Current Head Position = 50

SCAN Disk Scheduling (Elevator)

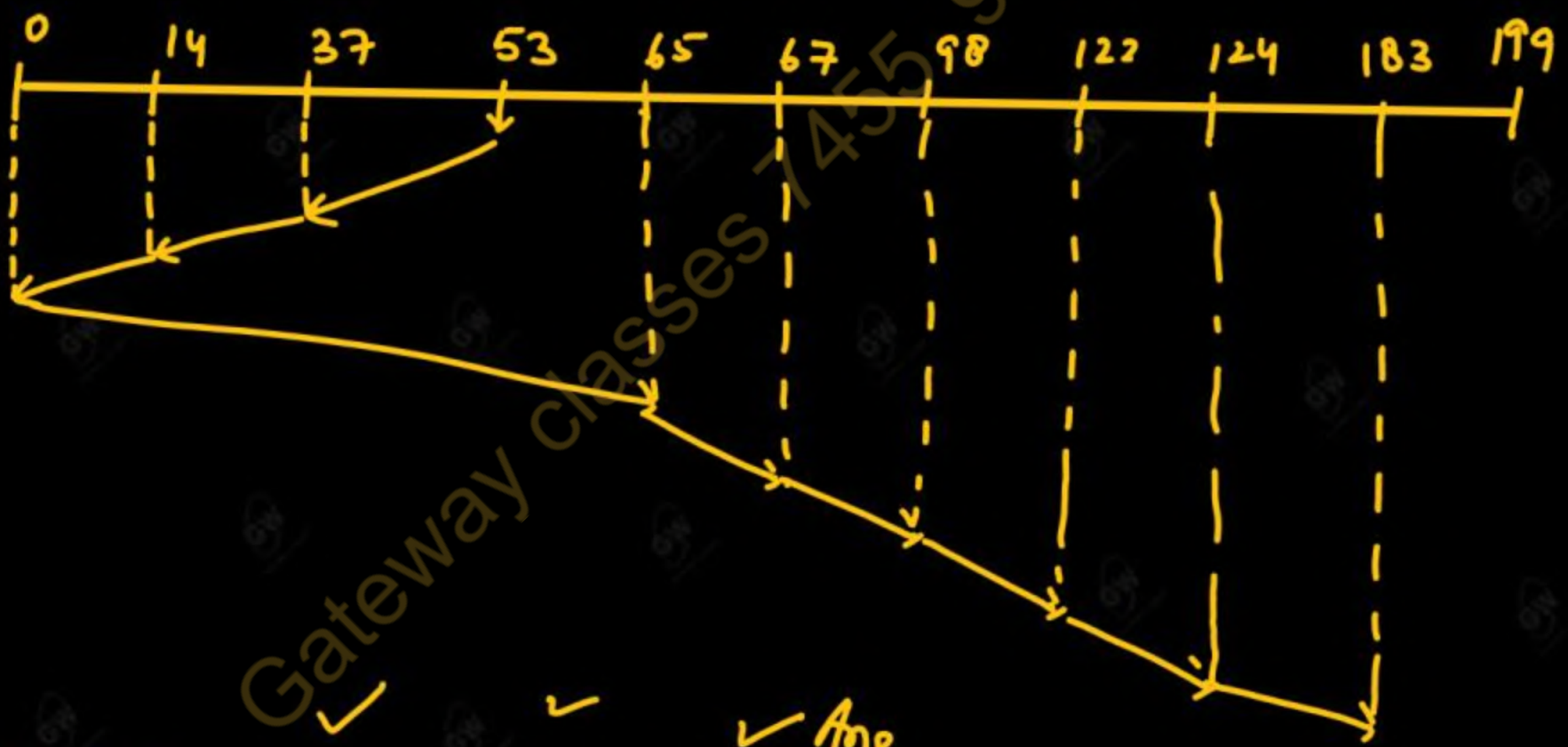
- In this algorithm, the disk arm starts at one end of the disk and moves toward the other end, servicing requests as it reaches each cylinder, until it gets to the other end of the disk.
- At the other end, the direction of the head movement is reversed, and servicing continues.
- In SCAN algorithm the disk arm moves to a particular direction and services the requests coming in its path and after reaching the end of the disk, it reverses its direction and again services the request arriving in its path.
- As a result, the requests at the midrange are serviced more and those arriving behind the disk arm will have to wait.

Example 3: Consider a disk queue with requests for I/O to blocks on cylinder –

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

Assuming that the disk arm is moving toward 0 and the initial head position is 53. Find the total head movements using SCAN disk scheduling algorithm. Elevator

Solution:-



Total Head movement = (53 – 0) + (183 – 0) = 236

SCAN

QUEUE - 98, 183, 37, 122, 14, 124, 65, 67

disk arm moving towards 0, Head Position = 53

Example 4: Consider a disk queue with requests for I/O to blocks on cylinder –

Queue : 72, 160, 33, 130, 14, 6, 180

Assuming that the disk arm is moving toward the larger value and the initial read/write head position is 50. Find the total head movements using SCAN disk scheduling algorithm. The number of cylinders are 200.

Solution:-

Total Head movements = (199 – 50) + (199 – 6) = 342

SCAN, QUEUE - 72, 160, 33, 130, 14, 6, 180

Towards larger value, Current Head Position = 50

C – SCAN Disk Scheduling Algorithm

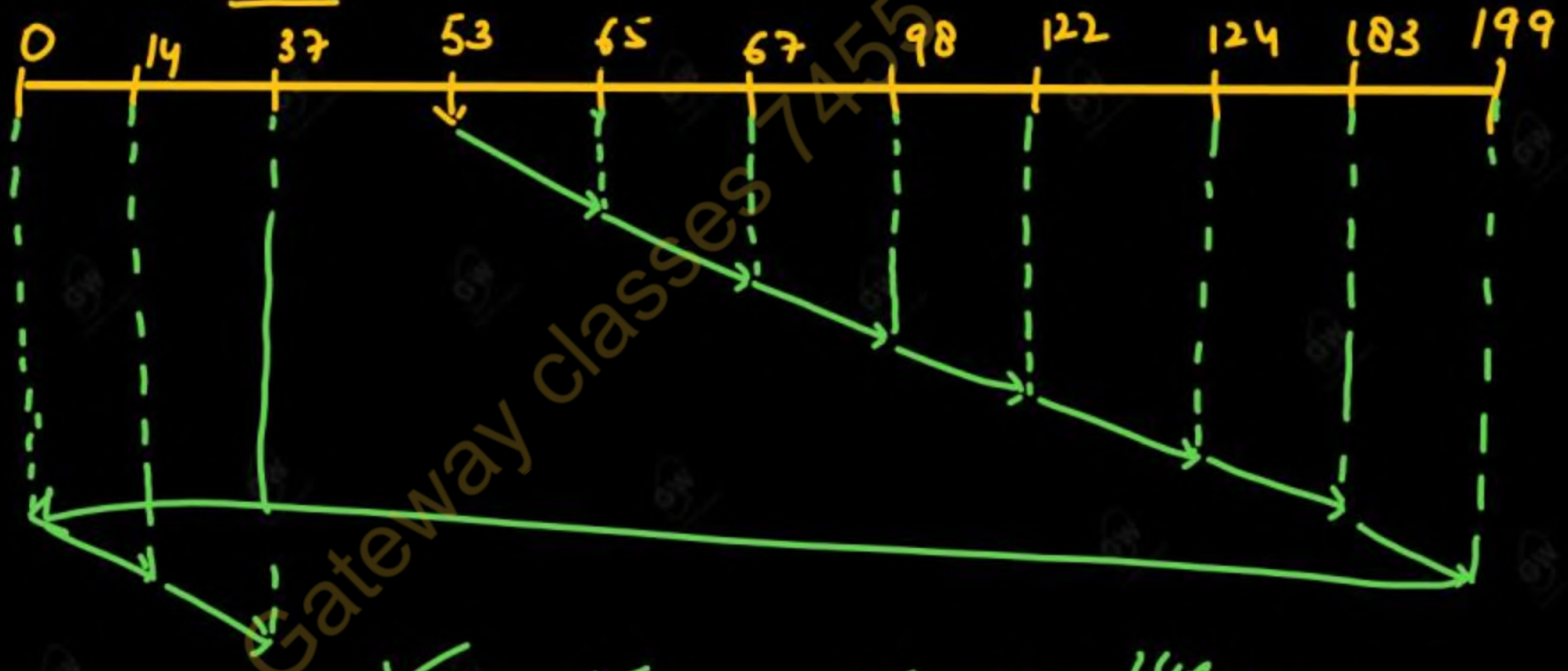
- Like SCAN, C-SCAN moves the head from one end of the disk to the other, servicing requests along the way.
- When the head reaches the other end, however, it immediately returns to the beginning of the disk without servicing any requests on the return trip.
- The C-SCAN scheduling algorithm essentially treats the cylinders as a circular list that wraps around from the final cylinder to the first one.
- In the SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction.
- So, it may be possible that too many requests are waiting at the other end or there may be zero or few requests pending in the scanned area. These situations are avoided in the C-SCAN algorithm in which the disk arm instead of reversing its direction goes to the other end of the disk and starts servicing the request from there.
- So, the disk arm moves in a circular fashion and this algorithm is also similar to the SCAN algorithm hence it is known as C-SCAN.

Example 5: Consider a disk queue with requests for I/O to blocks on cylinder –

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

Assuming that the direction of disk arm is toward the larger value and the initial read/write head position is 53. Find the total head movements using C-SCAN disk scheduling algorithm. The number of cylinders are 200.

Solution:-



Total Head movements = (199 – 53) + (199 – 0) + (37 – 0) = 382

146 + 199 + 37

C SCAN

QUEUE - 90, 103, 37, 122, 14, 124, 65, 67

Towards the larger value, Current Head = 53

Example 6: Consider a disk queue with requests for I/O to blocks on cylinder –

Queue : - 72, 160, 33, 130, 14, 6, 180

Assuming that the direction of disk arm is toward the larger value and the initial read/write head position is 50. Find the total head movements using C-SCAN disk scheduling algorithm. The number of cylinders are 200.

Solution:-

Total Head movement = (199 – 50) + (199) + (33) = 381

CSCAN :- QUEUE - 72, 160, 33, 130, 14, 6, 180

Towards larger value, Current Head Position = 50

LOOK Disk Scheduling

- Both SCAN and C-SCAN move the disk arm to access the full width of the disk.
- In practice, neither algorithm is often implemented this way.
- More commonly the arm goes only for the final request in each direction.
- Then, it reverses direction immediately without going all the way to the end of the disk.
- Versions of SCAN and C-SCAN that follow this pattern are called LOOK and C-LOOK scheduling because they look for a request before continuing to move in a given direction.
- It is similar to SCAN except for the difference that the disk arm in spite of going to the end of the disk goes only to the last request to be serviced in front of the head and then reverses its direction from there only.
- Thus it prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

Example 7: A disk contains 200 tracks (0 – 199). The request queue contains track numbers -

82, 170, 43, 140, 24, 16, 190

Current position of R/W head is 50.

(i) Calculate total number of R/W head monument using LOOK scheduling algorithm.

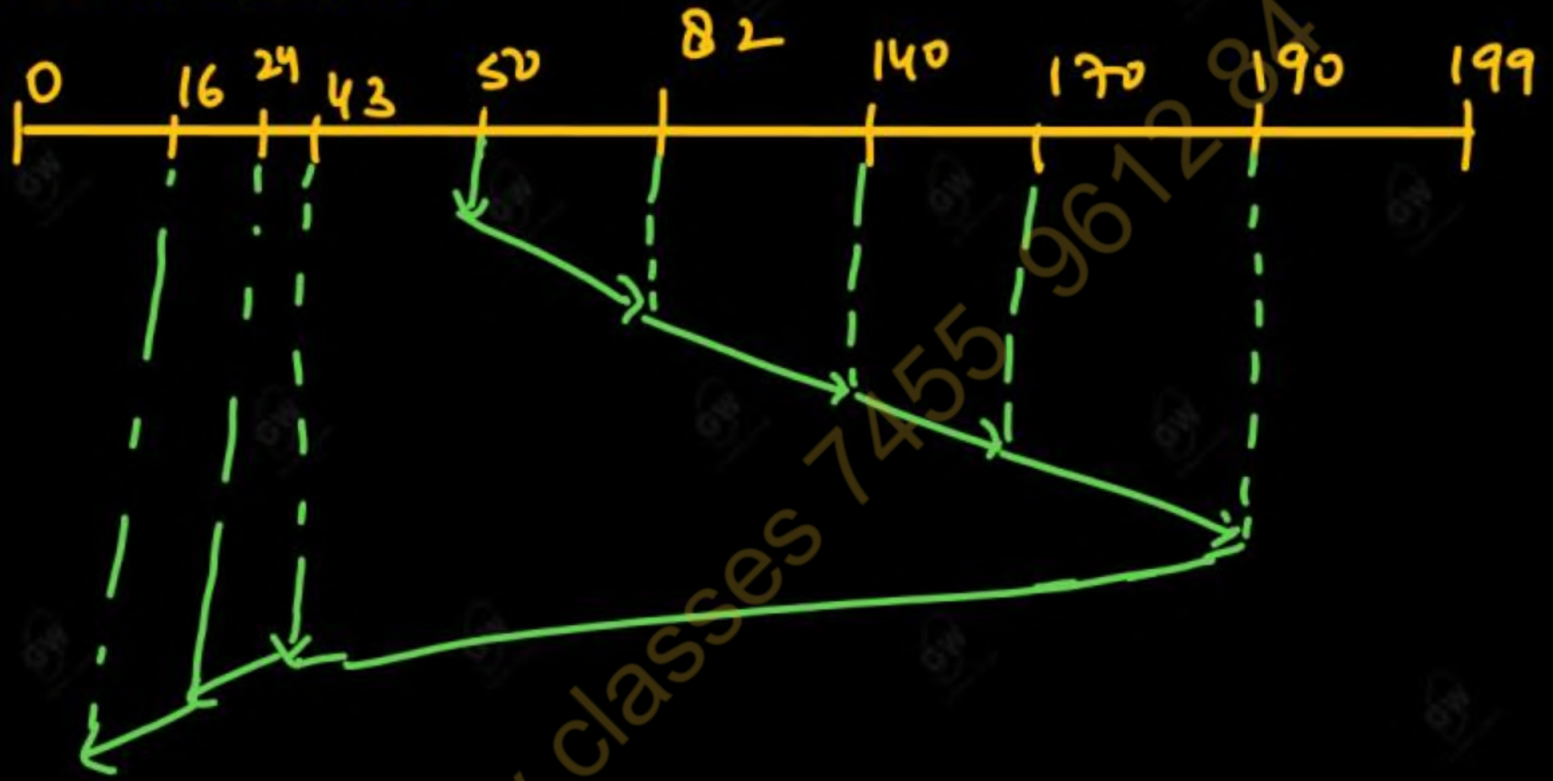
(ii) If R/W head takes 1 ns to move from one track to another then calculate the total time taken.

(iii) Given that direction is towards large value.

Solution:-

Note:- In SCAN we move in which direction, we go till the last (till 199) but in LOOK it is similar to SCAN except that will not go till the last. (Extra move is not required)

82, 170, 43, 140, 24, 16, 190



140 174 Ans

Total Head Movement = (190 – 50) + (190 – 16) = 314 (lesser then SCAN)

82, 170, 43, 140, 24, 16, 190

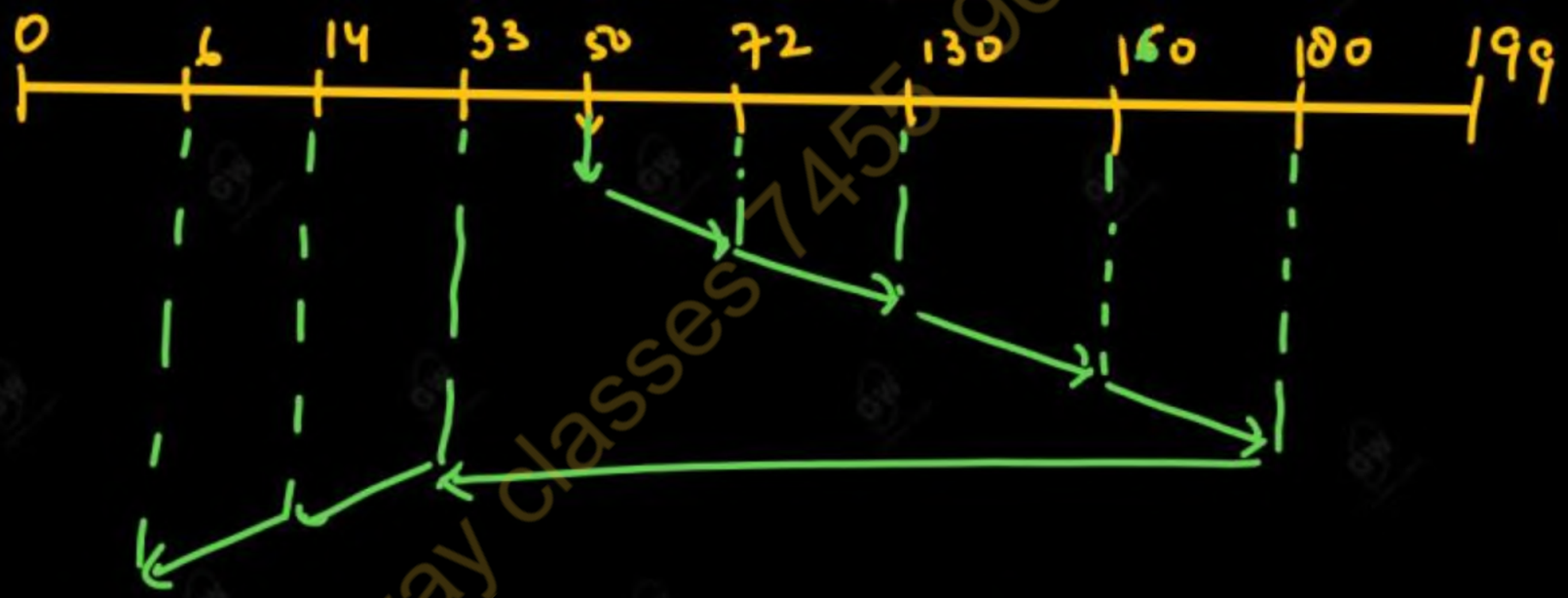
Towards larger value, Current head position = 50

Example 8: A disk contains 200 tracks (0-199). The request queue contains track numbers -

72, 160, 33, 130, 14, 6, 180

Current position of R/W head is 50. Calculate total number of R/W head monument using LOOK scheduling algorithm. Given that direction is towards large value.

Solution:-



Total Head Movements = (180 – 50) + (180 – 6) = 304

LOOK - QUEUE - 72, 160, 33, 130, 14, 6, 180

Towards larger value, Current head position = 50

CLOOK Disk Scheduling Algorithm

- As LOOK is similar to the SCAN algorithm in a similar way CLOOK is CSCAN.
- In CLOOK, the disk arm in spite of going to the end goes only to the last request to be services in front of the head and then from there goes to the other end's last request.
- Thus, it also prevents the extra delay which occurred due to unnecessary traversal to end of the disk.

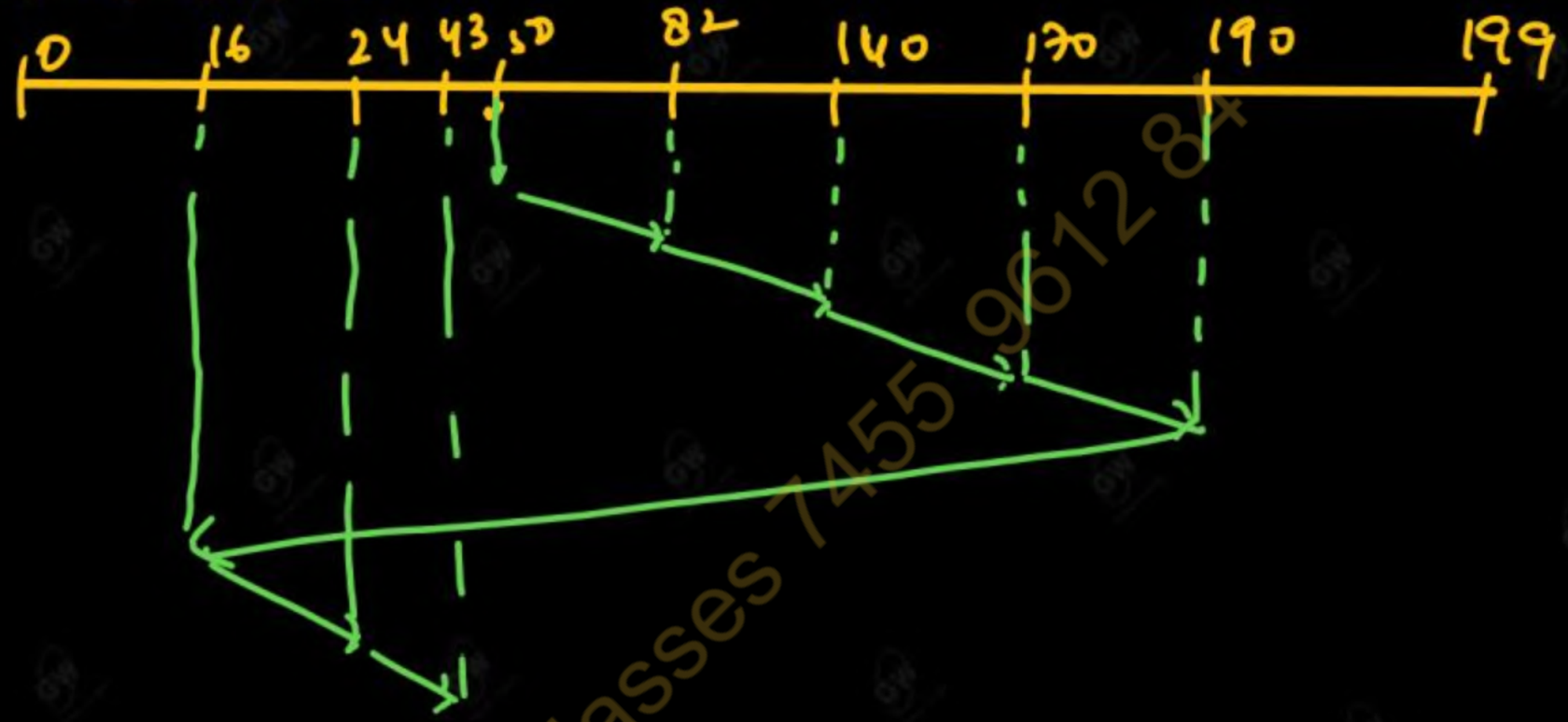
Example 9: A disk contains 200 tracks (0-199). The request queue contains track numbers -

82, 170, 43, 140, 24, 16, 190

Current position of R/W head is 50. Calculate total number of R/W head monument using CLOOK scheduling algorithm. Given that direction is towards large value.

Solution:-

- 82, 170, 43, 140, 24, 16, 190



190-16 43-16

140
174
27
341

- Total Head Movement:- (190-50) + (~~180-6~~) + (~~33-6~~) = ~~331~~

CLOOK: 82, 170, 43, 140, 24, 16, 190

Current Head position = 50, towards larger value

Example 10:- Explain the SSTF and SCAN disk scheduling policies. Obtain the total number of head movements needed to satiety the following sequence of track requests for each of the two policies : 27, 129, 110, 186, 147, 41, 10, 64, 120

Assume that the disk head is initially positioned over track 100 and is moving in the direction of decreasing track number.

Solution:-

SSTF

- **Head Movement** = (186 - 100) + (186 - 10) = 86 + 176 = 262

❑ SCAN Disk Scheduling:

❑ Total Head Movements = (100 - 0) + (186 - 0) = 100 + 186 = 286

Ans

Summary

- **FCFS:-** It is first come first serve.
- **SSTF:-** Calculate the shortest distance.
- **SCAN:-** Direction should be given otherwise towards greater numbers, in scanning direction, scan till last, like if 200 cylinders from 0-199, then if scanning is in forward direction, scan till 199 & if scanning is in backward direction then go till 0.
- **CSCAN:-** In the scanning direction reach till the last (199) and come back to 0 and not provide service in between. Then again start providing the service.
- **LOOK:-** It is like SCAN but do not go till last.
- **CLOOK:-** It is like CSCAN but do not go till last.

AKTU PYQs

Q.1. Suppose the following disk request sequence (track numbers) for a disk with 100 tracks is given: 45, 20, 90, 10, 50, 60, 80, 25, 70. Assume that the initial position of the R/W head is on track 49. Calculate the net head movement using:

(i) SSTF (ii) SCAN (iii) CSCAN (iv) LOOK **2022-23, 10 Marks**

Q.2. A hard disk having 2000 cylinders, numbered from 0 to 1999. The drive is currently serving the request at cylinder 143, and the previous request was at cylinder 125. The status of the queue is as follows- 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

What is the total distance (in cylinders) that the disk arm moves to satisfy the entire pending request for each of the following disk-scheduling algorithms?

(i) SSTF (ii) FCFS **2021-22, 10 Marks, 2018-19, 7 Marks**

Q.3. Define SCAN and C-SCAN scheduling algorithms. **2018-19, 2 Marks**

Q.4. The head of a moving head disk with 200 tracks numbered 0 to 199 is currently serving request at track 143 and the last served request was 135. Consider an ordered disk queue with request involving track number- 86, 147, 91, 177, 94, 150, 102, 175, 130

Find the total head movement for the following -

(i) SSTF (ii) C-SCAN 2017-18, 10 Marks

Q.5. Explain FCFS, SCAN & CSCAN scheduling with example. 2016-17, 15 Marks

Q.6. Suppose the moving head disk with 200 tracks is currently serving a request for track 143 and has just finished a request for track 125. If the queue of request is kept in FIFO order 86, 147, 91, 177, 94, 150. What is total head movement for the following scheduling:

(i) FCFS (ii) SSTF (iii) C-SCAN 2015-16, 10 Marks

Q.7. Discuss disk scheduling algorithms with example. 2014-15, 10 Marks

RAID (Redundant Array of Independent or Inexpensive Disks)

- Redundant means multiple copies. Here redundancy is of disk.
- The disk is being used for storing the data.
- The disk is independent and an array of disks means multiple independent disks.
- Independent means if one disk will not work another will work i.e. one disk will not affect the working of the other disk. There is no dependency among disks.
- Inexpensive in consideration with memory hierarchy in which registers are most expensive (at higher level) and disk are less expensive (at lower level in hierarchy).
- The cost of disk is gradually decreasing day by day. Nowadays size is in TB.
- RAID is a technique that makes use of a combination of multiple disks for storing the data instead of using a single disk for increased performance, data redundancy, or to protect data in the case of a drive failure.
- The term was defined by David Patterson, Garth A. Gibson, and Randy Katz at the University of California, Berkeley in 1987.

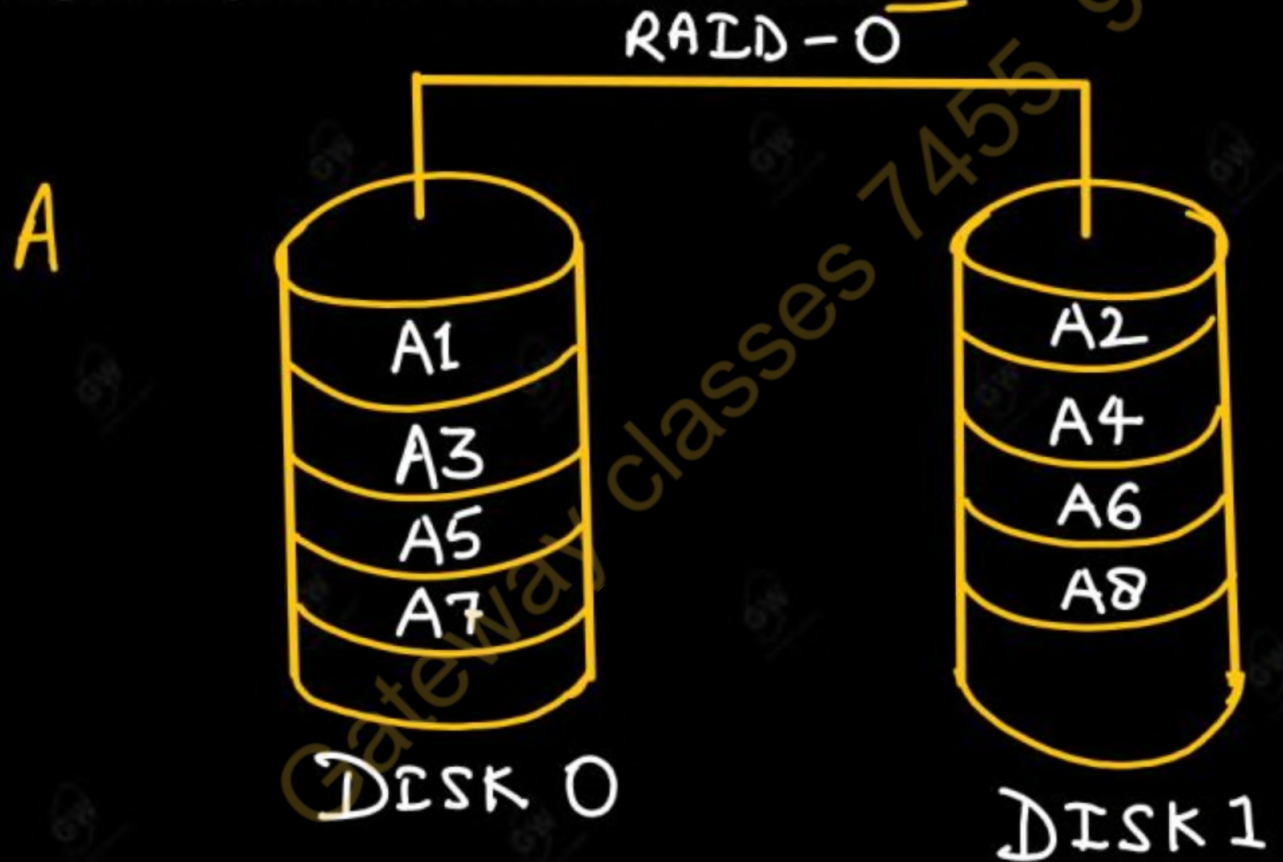
What is RAID?

- RAID (Redundant Array of Independent Disks) is like having backup copies of your important files stored in different places on several hard drives or solid-state drives (SSDs).
- If one drive stops working, your data is still safe because you have other copies stored on the other drives.
- It's like having a safety net to protect your files from being lost if one of your drives breaks down.
- RAID in a Database Management System (DBMS) is a technology that combines multiple physical disk drives into a single logical unit for data storage.
- The main purpose of RAID is to improve data reliability, availability, and performance.
- There are different levels of RAID, each offering a balance of these benefits.

- What is the need for RAID?
- PERFORMANCE:-
- We want to perform the R & W operations very fastly to improve the overall performance.
- SECURITY:- (Availability)
- Data must be available all the time. (24 hours).

RAID - 0 (Stripping)

- Data Stripping is breaking data into different parts and keeping it in different disks.
- Like in the next diagram, the original data was 'A' is broken into A1, A2, A3, A4, A5, and so on, and keeping these pieces of data in different disks.



- In this manner, it improves the performance because if you want to read 'A' then you can read the data of A in parallel as the data is available on multiple disks in pieces.
- In the same manner, if you want to perform the operation of "write" on A, it can be written in different pieces in parallel.
- Throughput is high.
- **Reliability:** There is no duplication of data. Hence, a block once lost cannot be recovered.

I1 : a = b + c

I2 : X = a + Z

Advantages -

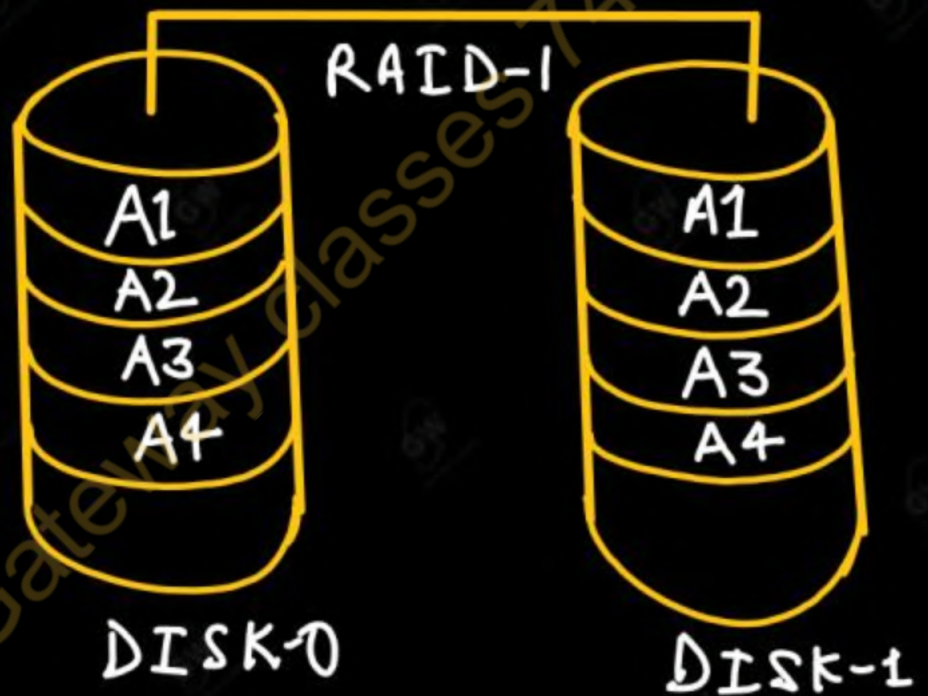
- It is easy to implement.
- It utilizes the storage capacity in a better way.

Disadvantages -

- A single drive loss can result in the complete failure of the system.
- It's not a good choice for a critical system.

RAID 1 (Mirroring)

- At this level, we keep the same copy of data on another disk (mirror image).
- It may increase the cost but in comparison to other memories, it is inexpensive disk.
- It does not consider the performance it considers the security or data availability.
- Lets say if one disk is failed still we can access the data because it is available with another disks.



- Nowadays companies keep a number of mirror images of the same data, in the distributed mirror at different locations geographically.

Advantages -

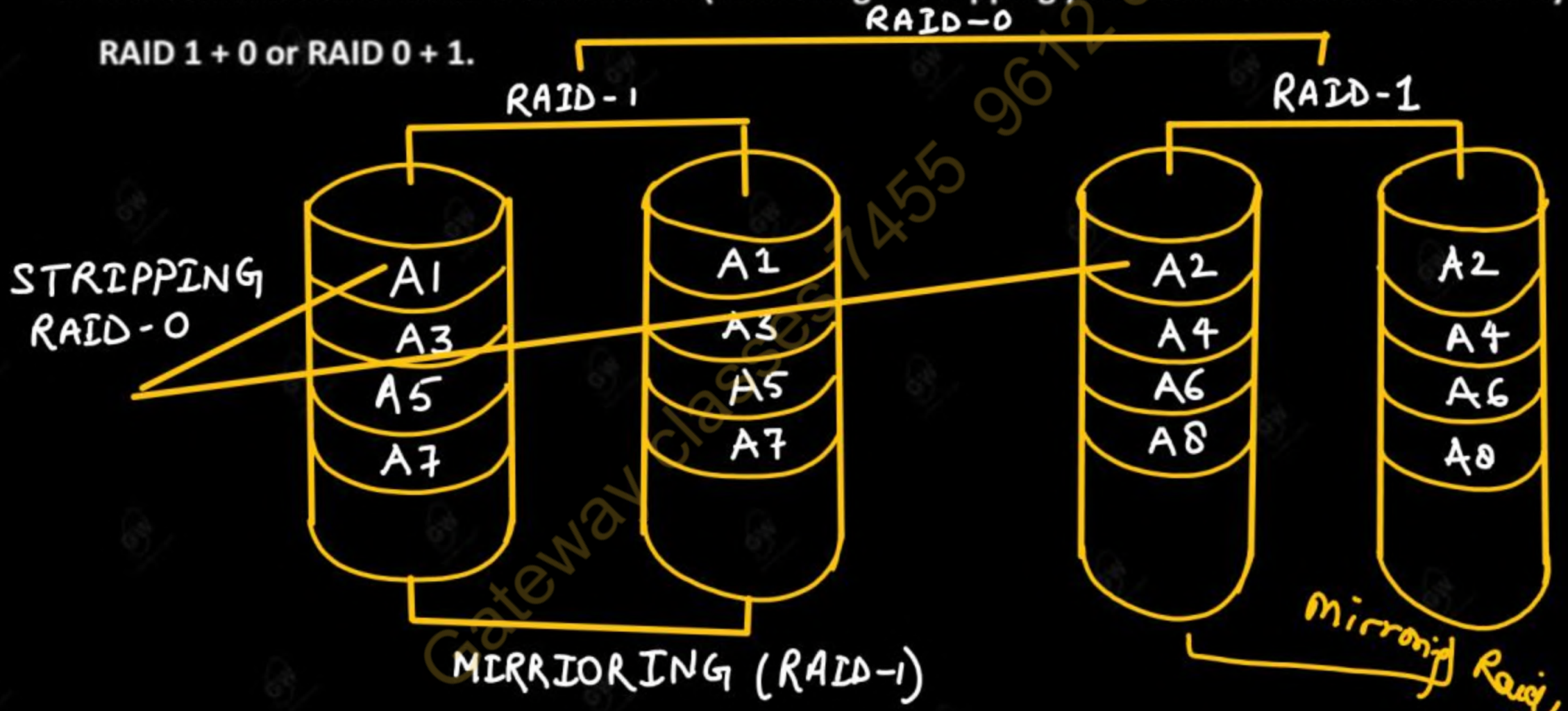
- It covers complete redundancy.
- It can increase data security and speed.

Disadvantages -

- It is highly expensive.
- Storage capacity is less.

RAID 1 + 0 or RAID 10 or RAID 01

- It is combination of RAID 0 and RAID 1 (Mirroring + Stripping) and can be written as either by RAID 1 + 0 or RAID 0 + 1.



- The actual data say 'A' is broken in A1, A2, A3, ... and so on, and maintained a copy of all pieces too.
- It is also known as Nested RAID.
- This technology is very useful.
- Nowadays such as in email servers webservers etc., it is being used to improve performance and to increase the data availability (security).
- The RAID 0 gives high performance and RAID 1 provides data security.
- RAID 1+0 (RAID 10) is an effective solution for applications requiring high performance and high availability, such as databases, file servers, and other critical systems.

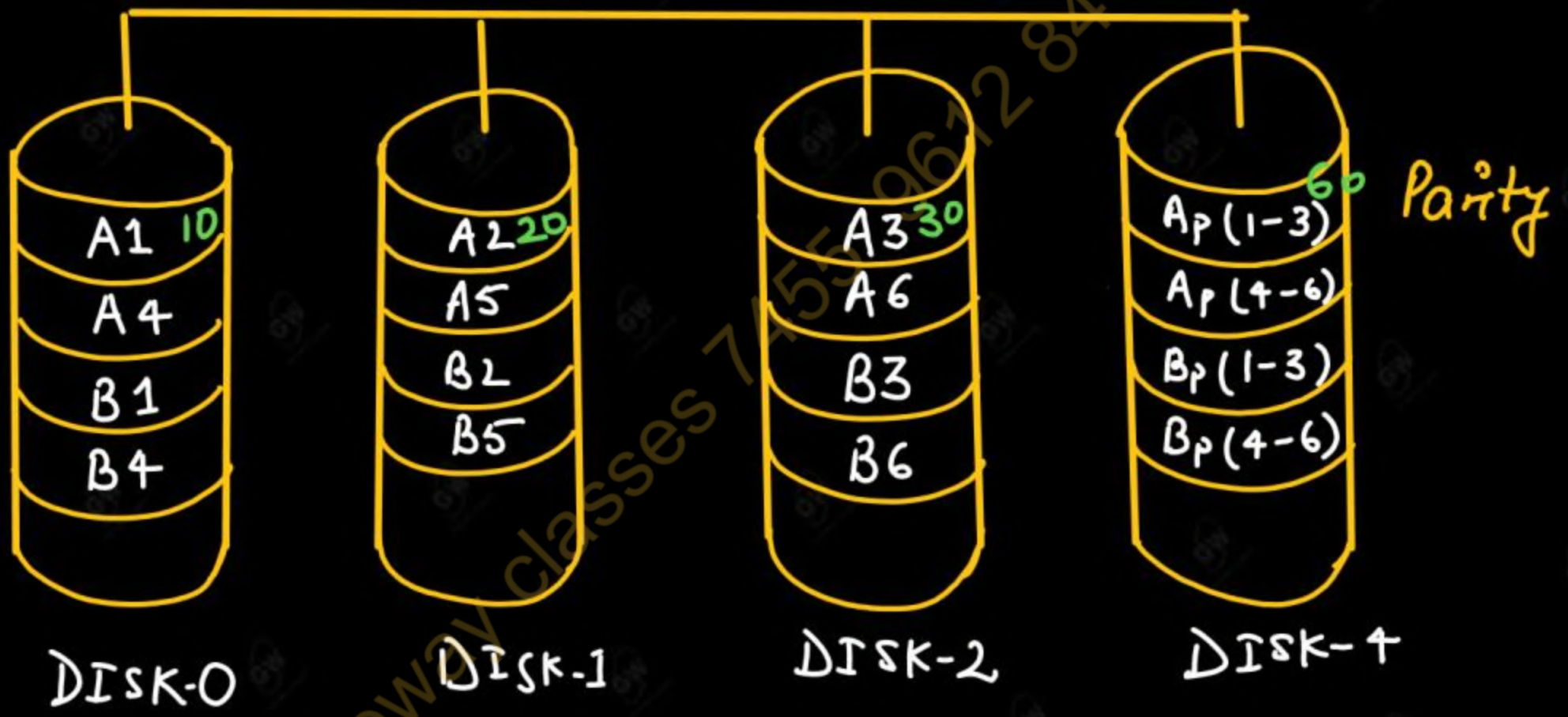
Advantages:

- Redundancy: If a drive fails, data is still accessible from the mirrored copy.
- Performance: Improved read/write speeds due to striping.

RAID 3 and RAID 4 (Striping with Dedicated Parity)

- RAID 3 is a type of RAID (Redundant Array of Independent Disks) configuration that uses striping with dedicated parity.
- Data is divided into blocks and these different blocks of original data is kept on different disk.
- Find the parity of data and store that on different disk.
- Data is striped at the byte level across multiple disks, with one disk dedicated to storing parity information.
- Parity: Parity is a form of error checking that helps reconstruct data in case one of the disks fails.
- The parity information is calculated using an XOR operation and is stored on the dedicated parity disk.

RAID - 3



- If one disk is failed the data can be recovered from the parity. (If more than one disk will be failed then parity will not work).
- All parities are stored on one disk (problem).
- Bottleneck (write operation in any disk will require modification on disk containing the parity, more pressure on a single disk).

Performance:

- Read Performance: Generally good because data can be read from multiple disks simultaneously.
- Write Performance: This can be a bottleneck because every write operation requires updating the parity disk, which can slow down the process due to the need to read existing data and parity, then calculate and write new parity.

RAID -3 – Byte level.
4 – Block level

Advantages

- **Data Protection:** Provides fault tolerance by allowing data to be reconstructed in case of a single disk failure.
- **Improved Read Performance:** Multiple disks are read simultaneously, enhancing read speeds.

Disadvantages

- **Parity Disk Bottleneck:** The dedicated parity disk can become a performance bottleneck during write operations because it needs to be updated with every write.
- **Limited Scalability:** The performance benefit is reduced as more disks are added because the single parity disk must handle all parity calculations.
- **Byte-Level Striping:** This fine-grained striping is less common and can be less efficient than block-level striping used in other RAID levels.

RAID 5 (Striping with Distributed Parity)

- The problem of RAID 3 is solved by RAID 5.
- Different ways of storing the parity.
- Distribution of parity: not on a single disk i.e. no overhead on a single disk
- Single disks will not be overloaded.

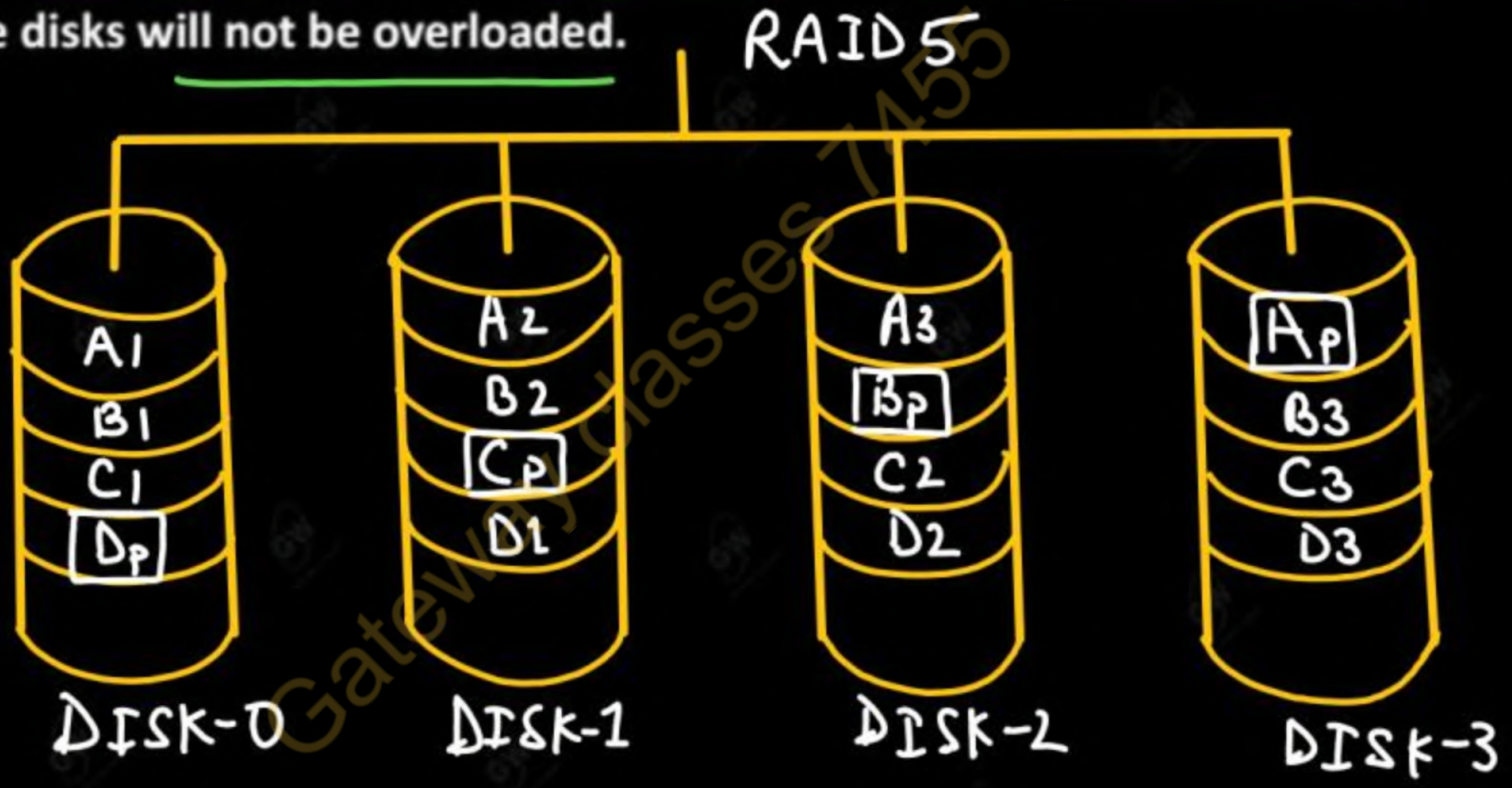


FIG:- DISTRIBUTION OF PARITIES AMONG THE DISK IN RAID 5

Advantages -

- **Data Protection:** Can tolerate a single disk failure without data loss.
- **Improved Read Performance:** Multiple disks can be read simultaneously, enhancing read speeds.
- **Efficient Use of Disk Space:** Provides a good balance between redundancy and usable storage capacity. The storage capacity of the array is (N-1) times the size of the smallest disk, where N is the number of disks.

Disadvantages -

- **Rebuild Time:** If a disk fails, rebuilding the array (i.e., reconstructing the lost data on a new disk) can be time-consuming and impact performance.
- **Write Performance:** While better than RAID 4, write performance can still be slower than RAID 0 or RAID 1 due to the need to calculate and write parity information.

RAID 6 (Striping with Double Distributed Parity)

- Two parities on different disk.
- If two disks fails, even the data can be recovered from the parity.

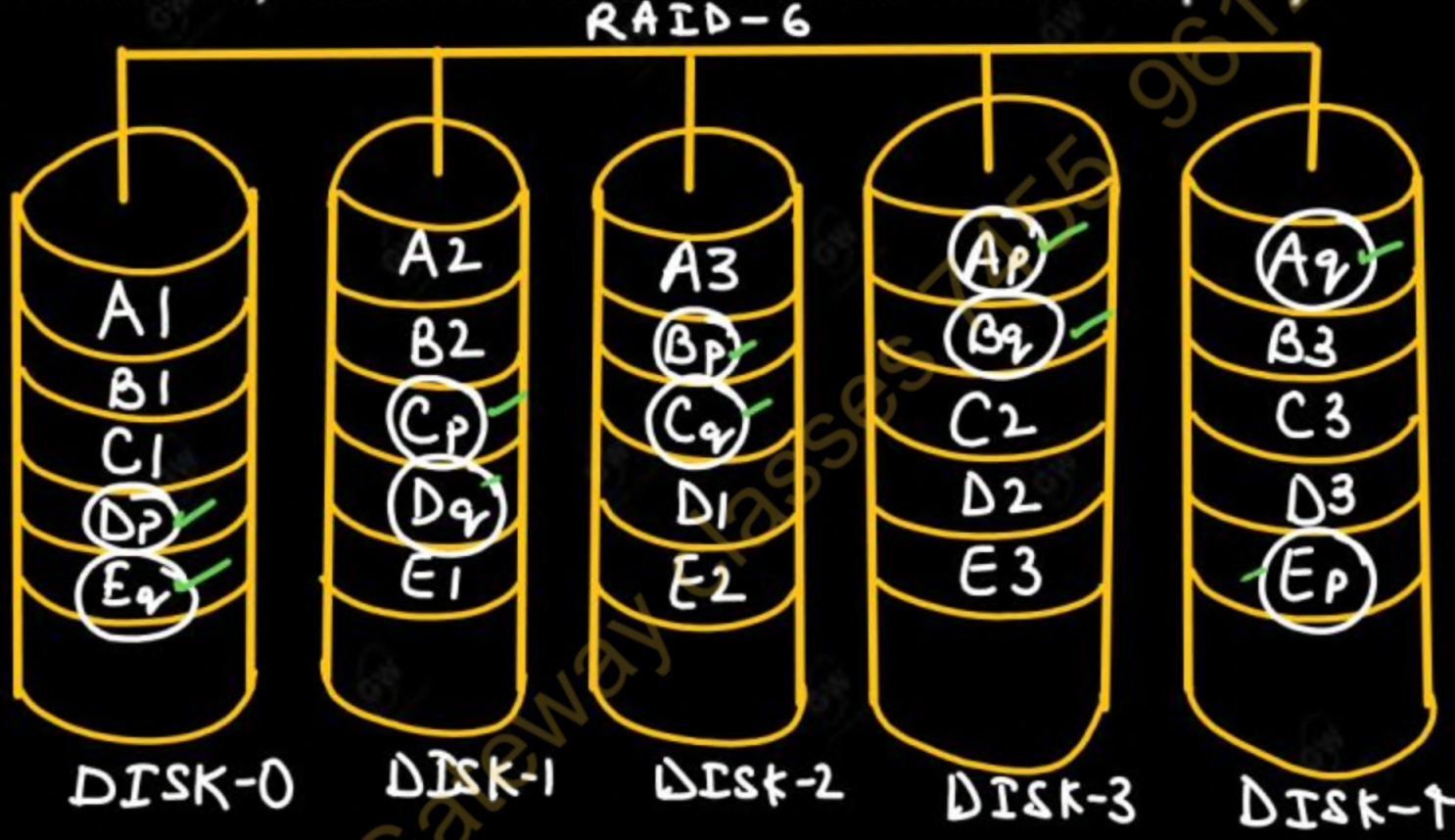


FIG:- DOUBLE PARITIES (P,q) ON DIFFERENT DISK IN RAID-6

Advantages -

- **High Fault Tolerance:** Can withstand up to two simultaneous disk failures without data loss.
- **Improved Read Performance:** Multiple disks can be read simultaneously, providing good read speeds.
- **Efficient Use of Disk Space:** Compared to mirroring (like in RAID 1), RAID 6 provides more usable storage capacity. The storage capacity of the array is (N-2) times the size of the smallest disk, where N is the number of disks.

Disadvantages -

- **Write Performance Overhead:** Write operations are slower compared to RAID 5 due to the need to calculate and write two sets of parity information.
- **Rebuild Time:** Rebuilding the array after a disk failure can be time-consuming and impacts performance, especially if two disks have failed.
- **Complexity:** More complex to implement and manage compared to RAID 5 or RAID 1.

AKTU PYQs

Q.1. Explain the term RAID and its characteristics. Also, explain various RAID levels with their advantages and disadvantages. **2022-23, 10 Marks**

Q.2. Explain RAID? **2018-19, 2 Marks**

File Concept

- Computers can store information on several different storage media, such as magnetic disks, magnetic tapes, and optical disk etc.
- The operating system abstracts from the physical properties of its storage devices to define a logical storage unit (the FILE).
- Files are mapped, by the operating system, onto physical devices.
- These storage devices are usually nonvolatile, so the contents are persistent through power failures and system reboots.
- A file is a named collection of related information that is recorded on secondary storage.
- From a user's perspective, a file is the smallest allotment of logical secondary storage; that is, data cannot be written to secondary storage unless they are within a file.
- Collection of files is known as file directory for example folder in windows operating system.

- Commonly, files represent programs (both source and object forms) and data.
- Data files may be numeric, alphabetic, alphanumeric, or binary.
- In general, a file is a sequence of bits, bytes, lines, or records, the meaning of which is defined by the file's creator and user.
- Many different types of information may be stored in a file-source programs, object programs, executable programs, numeric data, text, payroll records, graphic images, sound recordings, and so on.

File System

- Module of operating system which manages, controls and organizes files and related structures.
- Its provides the organized view of files in the system.

File - Structure

- A file has a certain defined structure according to its type.
- A text file is a sequence of characters organized into lines (and possibly pages).
- A source file is a sequence of subroutines and functions, each of which is further organized as declarations followed by executable statements.
- An object file is a sequence of bytes organized into blocks understandable by the system's linker.
- An executable file is a series of code sections that the loader can bring into memory and execute.

File Attributes

- A file has certain attributes -
- **Name:** The symbolic file name is the only information kept in human readable form.
- **Extension :** Extension represent the type of the file e.g. .doc, .pdf etc.
- **Date:** Date of creation, modification etc.
- **Type:** This information is needed for those systems that support different types.
- **Location:** This information is a pointer to a device and to the location of the file on that device.
- **Size:** The current size of the file (in bytes, words, or blocks), and possibly the maximum allowed size are included in this attribute.
- **Protection:** Access-control information determines who can do reading, writing, executing, and so on.
- **Time, date, and user identification:** This information may be kept for creation, last modification, and last use. These data can be useful for protection, security, and usage monitoring.

Various Operations Associated with File

- The operating system can provide system calls to create, write, read, reposition, delete, and truncate files.
- The basic file operations are:
- **Creating a file:**

- Two steps are necessary to create a file. First, space in the file system must be found for the file.
- Second, an entry for the new file must be made in the directory.
- The directory entry records the name of the file and the location in the file system, and possibly other information.

Writing a file:

- To write a file, we make a system call specifying both the name of the file and the information to be written to the file.
- Given the name of the file, the system searches the directory to find the location of the file.
- The system must keep a write pointer to the location in the file where the next write is to take place.
- The write pointer must be updated whenever a write occurs.

Reading a file:

- To read from a file, we use a system call that specifies the name of the file and where (in memory) the next block of the file should be put.
- Again, the directory is searched for the associated directory entry, and the system needs to keep a read pointer to the location in the file where the next read is to take place.
- Once the read has taken place, the read pointer is updated.
- A given process is usually only reading or writing a given file, and the current operation location is kept as a per-process current-file-position pointer.
- Both the read and write operations use this same pointer, saving space and reducing the system complexity.

Deleting a file:

- To delete a file, we search the directory for the named file.
- Having found the associated directory entry, we release all file space, so that it can be reused by other files, and erase the directory entry.

Truncating a file:

- The user may want to erase the contents of a file but keep its attributes.
- Rather than forcing the user to delete the file and then recreate it, this function allows all attributes to remain unchanged-except for file length-but lets the file be reset to length zero and its file space released.

File Types

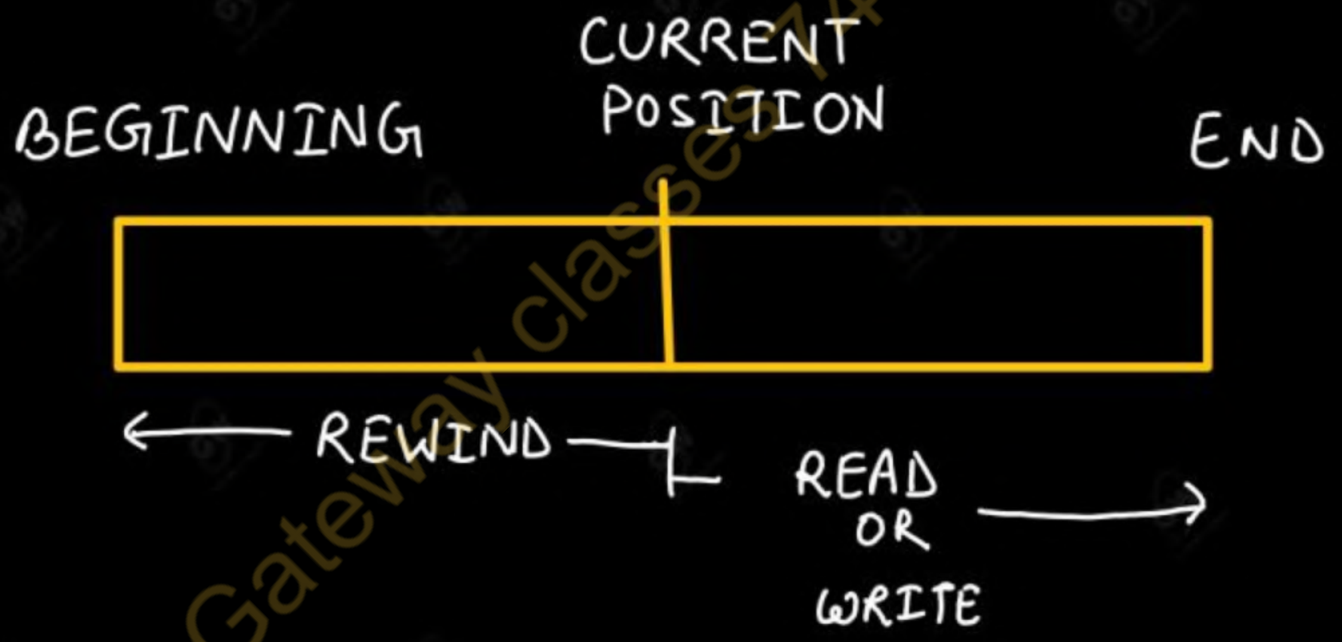
- A common technique for implementing file types is to include the type as part of the file name.
- The name is split into two parts-a name and an extension, usually separated by a period character.
- In this way, the user and the operating system can tell from the name alone what the type of a file is.
- For example, in MS-DOS, a name can consist of up to eight characters followed by a period and terminated by an extension of up to three characters.
- The system uses the extension to indicate the type of the file and the type of operations that can be done on that file.

File Type	Usual extension	Function
Executable	exe, com, bin or none	Ready – to run machine – language program
Object	obj, o	Compiled, machine language, not linked
Source code	c, cc, 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, rtf, doc	Various word-processor formats
Library	Lib, a, so, dll	Libraries of routines for programmers
Print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
Archive	arc, zip, tar	Related files grouped into one file, sometimes compressed, for archiving or storage
Multimedia	mpeg, mov, rm, mp3, avi	Binary file containing audio or A/V information

File Access Methods

- Files store information. When it is used, this information must be accessed and read into computer memory. The information in the file can be accessed in several ways.
- **Sequential Access**
- The simplest access method is sequential access.
- 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, editors and compilers usually access files in this fashion.
- The bulk of the operations on a file is reads and writes.
- A read operation reads the next portion of the file and automatically advances a file pointer, which tracks the I/O location.
- Similarly, a write appends to the end of the file and advances to the end of the newly written material (the new end of file).

- Such a file can be reset to the beginning and, on some systems; a program may be able to skip forward or backward n records, for some integer n-perhaps only for n = 1.
- Sequential access is based on a tape model of a file, and works as well on sequential-access devices as it does on random-access ones.



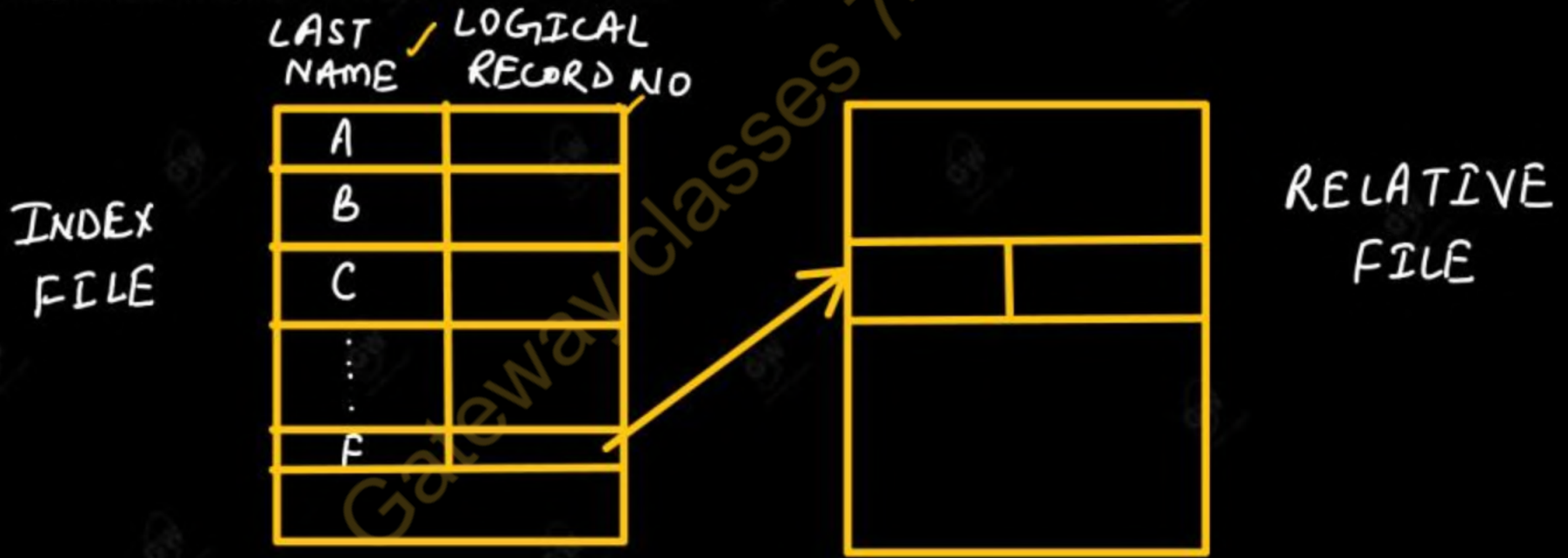
Direct Access

- Another method is direct access (or relative access).
- A file is made up of fixed length logical records that allow programs to read and write records rapidly in no particular order.
- The direct- access method is based on a disk model of a file, since disks allow random access to any file block.
- For direct access, the file is viewed as a numbered sequence of blocks or records.
- A direct- access file allows arbitrary blocks to be read or written. Thus, we may read block 14, then read block 53, and then write block 7. There are no restrictions on the order of reading or writing for a direct-access file.
- Direct-access files are of great use for immediate access to large amounts of information.
- Databases are often of this type. When a query concerning a particular subject arrives, we compute which block contains the answer, and then read that block directly to provide the desired information.

- For the direct-access method, the file operations must be modified to include the block number as a parameter. Thus, we have read n, where n is the block number, rather than read next, and write n rather than write next.
- The block number provided by the user to the operating system is normally a relative block number.
- A relative block number is an index relative to the beginning of the file. Thus, the first relative block of the file is 0, the next is 1, and so on, even though the actual absolute disk address of the block may be 14703 for the first block and 3192 for the second.
- The use of relative block numbers allows the operating system to decide where the file should be placed (called the allocation problem,), and helps to prevent the user from accessing portions of the file system that may not be part of his file.

Other Access Methods

- Other access methods can be built on top of a direct-access method.
- These methods generally involve the construction of an index for the file.
- The index, like an index in the back of a book, contains pointers to the various blocks.
- To find a record in the file, we first search the index, and then use the pointer to access the file directly and to find the desired record.



File System Protection and Security

- In computer systems, a lot of user's information is stored, the objective of the operating system is to keep safe the data of the user from the improper access to the system.
- Protection can be provided in number of ways.
- For a single laptop system, we might provide protection by locking the computer in a desk drawer or file cabinet. For multi-user systems, different mechanisms are used for the protection.
- **Types of Access :**
- The files which have direct access of the any user have the need of protection.
- The files which are not accessible to other users doesn't require any kind of protection.
- The mechanism of the protection provide the facility of the controlled access by just limiting the types of access to the file.

- Access can be given or not given to any user depends on several factors, one of which is the type of access required. Several different types of operations can be controlled:
- **Read** – Reading from a file.
- **Write** – Writing or rewriting the file.
- **Execute** – Loading the file and after loading the execution process starts.
- **Append** – Writing the new information to the already existing file, editing must be end at the end of the existing file.
- **Delete** – Deleting the file which is of no use and using its space for the another data.
- **List** – List the name and attributes of the file.
- Operations like renaming, editing the existing file, copying; these can also be controlled.
- There are many protection mechanism. Each of them mechanism have different advantages and disadvantages and must be appropriate for the intended application.

Access Control:

- There are different methods used by different users to access any file.
- The general way of protection is to associate identity-dependent access with all the files and directories an list called access-control list (ACL) which specify the names of the users and the types of access associate with each of the user.
- The main problem with the access list is their length.
- If we want to allow everyone to read a file, we must list all the users with the read access.

- To condense the length of the access-control list, many systems recognize three classification of users in connection with each file:

Owner – Owner is the user who has created the file.

Group – A group is a set of members who has similar needs and they are sharing the same file.

Universe – In the system, all other users are under the category called universe.

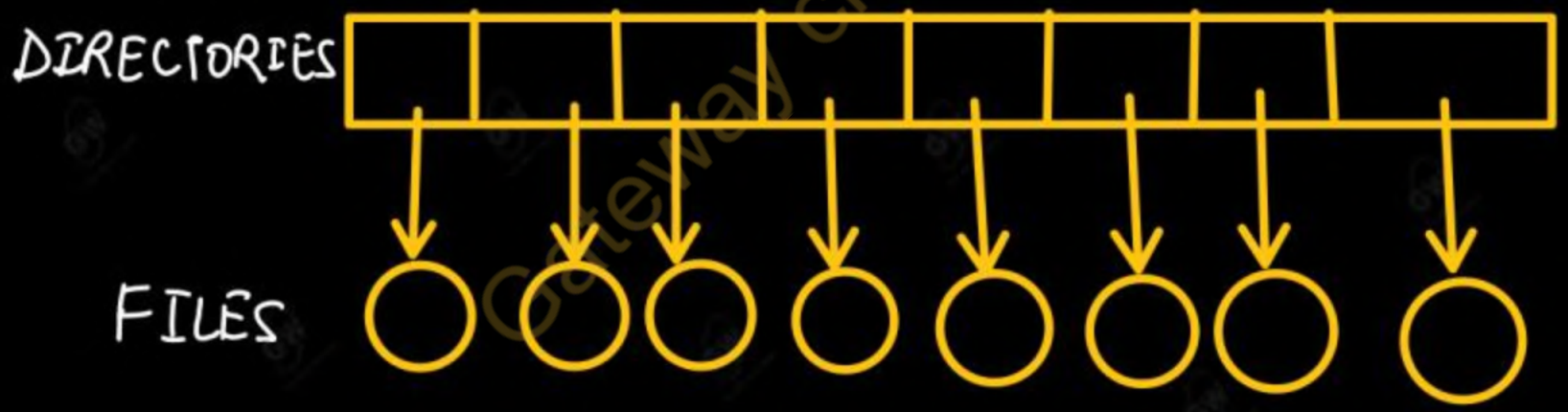
- The most common recent approach is to combine access-control lists with the normal general owner, group, and universe access control scheme.
- For example: Solaris uses the three categories of access by default but allows access-control lists to be added to specific files and directories when more fine-grained access control is desired.

Other Protection Approaches:

- The access to any system is also controlled by the password.
- If the use of password is random and it is changed often, this may be result in limit the effective access to a file.
- The use of passwords has a few disadvantages:
- The number of passwords are very large so it is difficult to remember the large passwords.
- If one password is used for all the files, then once it is discovered, all files are accessible; protection is on all-or-none basis.

File Directory Structure

- **Single-Level Directory**
- The simplest directory structure is the single-level directory.
- All files are contained in the same directory, which is easy to support and understand A single-level directory has significant limitations, however, when the number of files increases or when the system has more than one user.
- Since all files are in the same directory, they must have unique names. If two users call their data file test, then the unique-name rule is violated.



- Even a single user on a single-level directory may find it difficult to remember the names of all the files, as the number of files increases.
- It is not uncommon for a user to have hundreds of files on one computer system and an equal number of additional files on another system.
- In such an environment, keeping track of so many files is a daunting task.

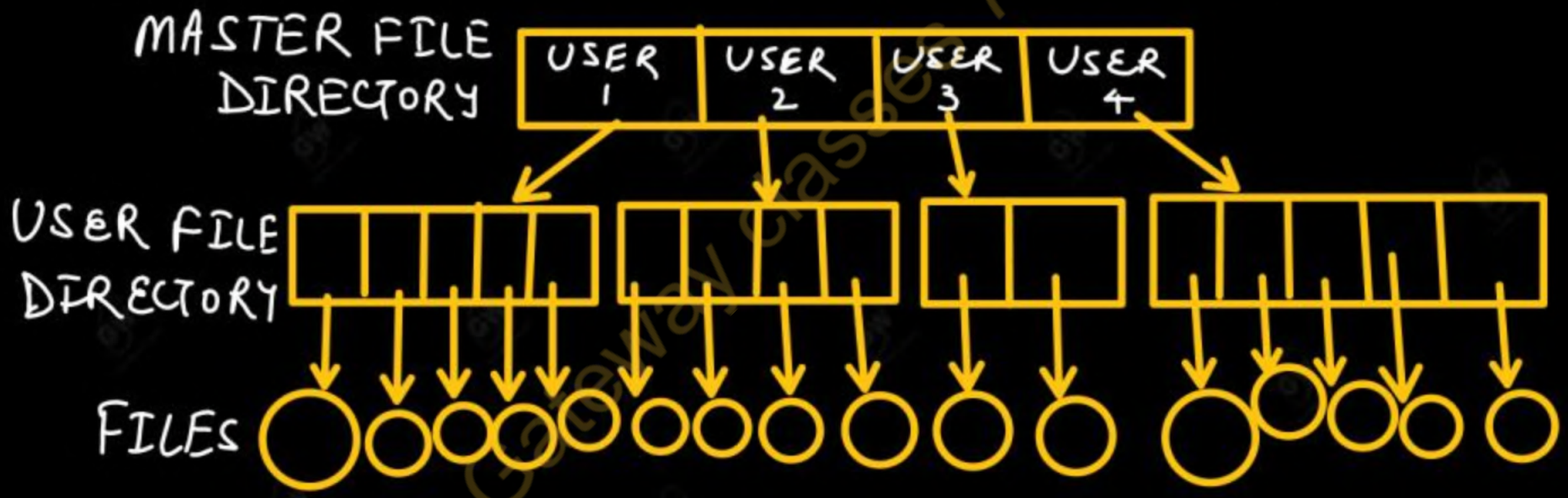


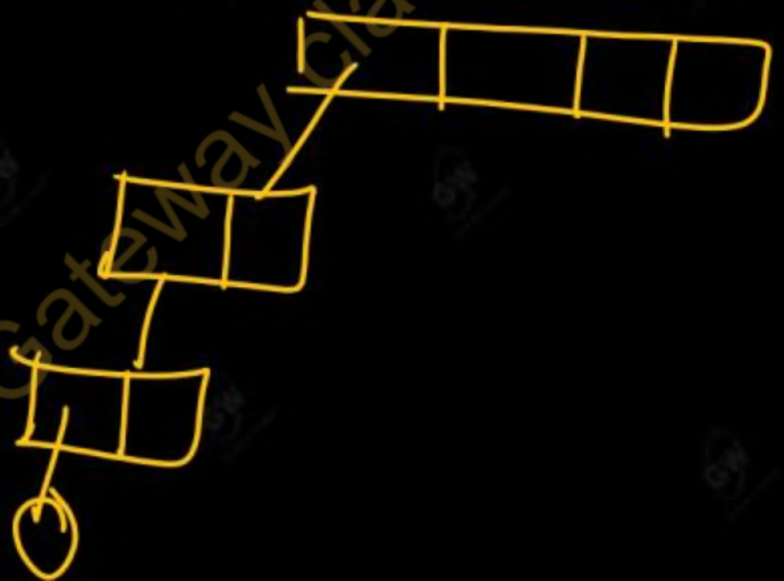
FIG:- TWO LEVEL DIRECTORY

Two-Level Directory

- A single-level directory often leads to confusion of file names between different users.
- The standard solution is to create a separate directory for each user.
- In the two-level directory structure, each user has her own user file directory (UFD).
- Each UFD has a similar structure, but lists only the files of a single user.
- When a user job starts or a user logs in, the system's master file directory (MFD) is searched.
- The MFD is indexed by user name or account number, and each entry points to the UFD for that user.

Tree Structured Directories

- The natural generalization of two-level directory is to extend the directory structure to a tree or arbitrary height.
- This generalization allows users to create their own sub directories and to organize their files accordingly.
- The tree has a root directory.
- Each file in the system has a unique path name.
- A path name is the path from the root, through all the subdirectories, to a specified file.



- Path names can be of two types: absolute path names or relative path names.
- An absolute path name begins at the root and follows a path down to the specified file, giving the directory names on the path.
- A relative path name defines a path from the current directory.
- For example, in the tree-structured file system of Figure, if the current directory is root/spell/mail, then the relative path name prt/first refers to the same file as does the absolute path name root/spell/mail/prt/fivst.

File Allocation Methods

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

1. Contiguous Allocation
2. Non contiguous Allocation

(a) Linked List Allocation

(b) Indexed Allocation

- The main idea behind these methods is to provide:

1. Efficient disk space utilization.
2. Fast access to the file blocks.

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.
- **Advantages:**
- Both the Sequential and Direct Accesses are supported by this allocation method.
- 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.

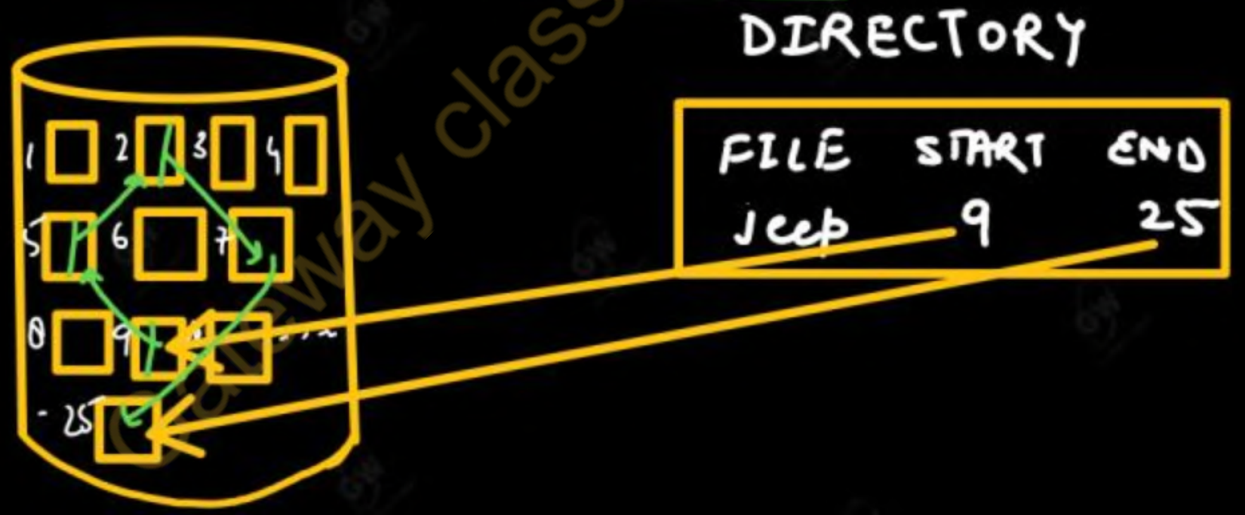
Block size = 5 KB
FILE size = 4 KB

	Start	Size
D1	2	2
D2	4	5

2. Non Contiguous Allocation

(a) 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 other any block.

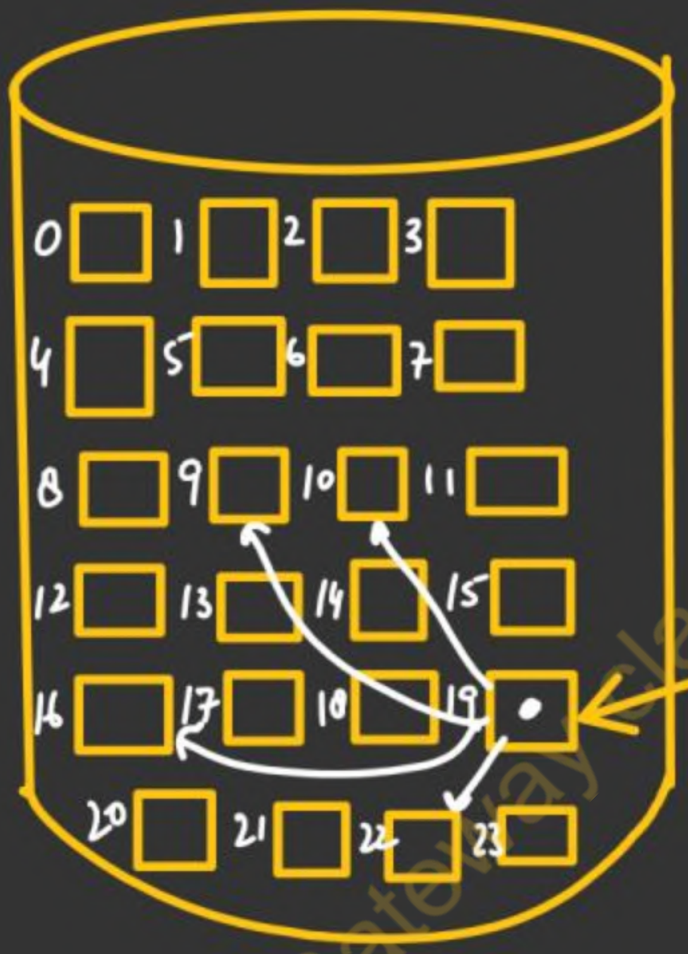


- **Advantages:**
  - This is very flexible in terms of file size.
  - File size can be increased easily since the system does not have to look for a contiguous chunk of memory.
  - This method does not suffer from external fragmentation.
  - This makes it relatively better in terms of memory utilization.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

FIG:- INDEXED ALLOCATION



DIRECTORY

File	index block
jeep	19 ✓

- **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.

AKTU PYQs

Q.1. Explain various operations associated with a file. 2022-23, 2 Marks

Q.2. Explain tree level directory structure. 2022-23, 2 Marks

Q.3. Explain the concept of file system management. Also, explain various file allocation and file access mechanisms in details. 2022-23, 2021-22, 10 Marks

Q.4. Explain in detail about the File system protection and security. 2021-22, 10 Marks

Q.5. Write short notes on following. 2021-22, 10 Marks, 2018-19, 7 Marks

(i) File system protection and security and

(ii) Linked File allocation methods

Q.6. Describe in term Directory system. 2018-19, 2 Marks

Q.7. What are files and explain the access methods for files. 2018-19, 7 Marks

Q.8. Discuss the Linked, Contiguous and Index and multilevel Indexing file allocation schemes. Which allocation scheme will minimize the amount of space required in directory structure and why? 2017-18, 7 Marks

Q.9. Difference between Directory and File. 2016-17, 2 Marks

Q.10. Explain File organization and Access mechanism. 2016-17,10 Marks

Q.11. What is directory? Explain any two ways to implement the directory. 2015-16, 10 Marks

Q.12. Describe schemes for defining logical structure of directory. 2014-15, 10 Marks

Download Gateway Classes Application
From Google Play store
Link in Description

Thank You

AKTU

B. Tech II-Year

Gateway Classes

Full Courses Available in App

AKTU B.Tech I- Year : All Branches

AKTU B.Tech II- Year

Branches : 1. CS IT & Allied

2. EC & Allied

3. ME & Allied

4. EE & Allied

Downlod App Now

Download App

Full Courses

- V. Lectures
- Pdf Notes
- AKTU PYQs
- DPP