# I/O Management and Disk scheduling

Tasneem Mirza

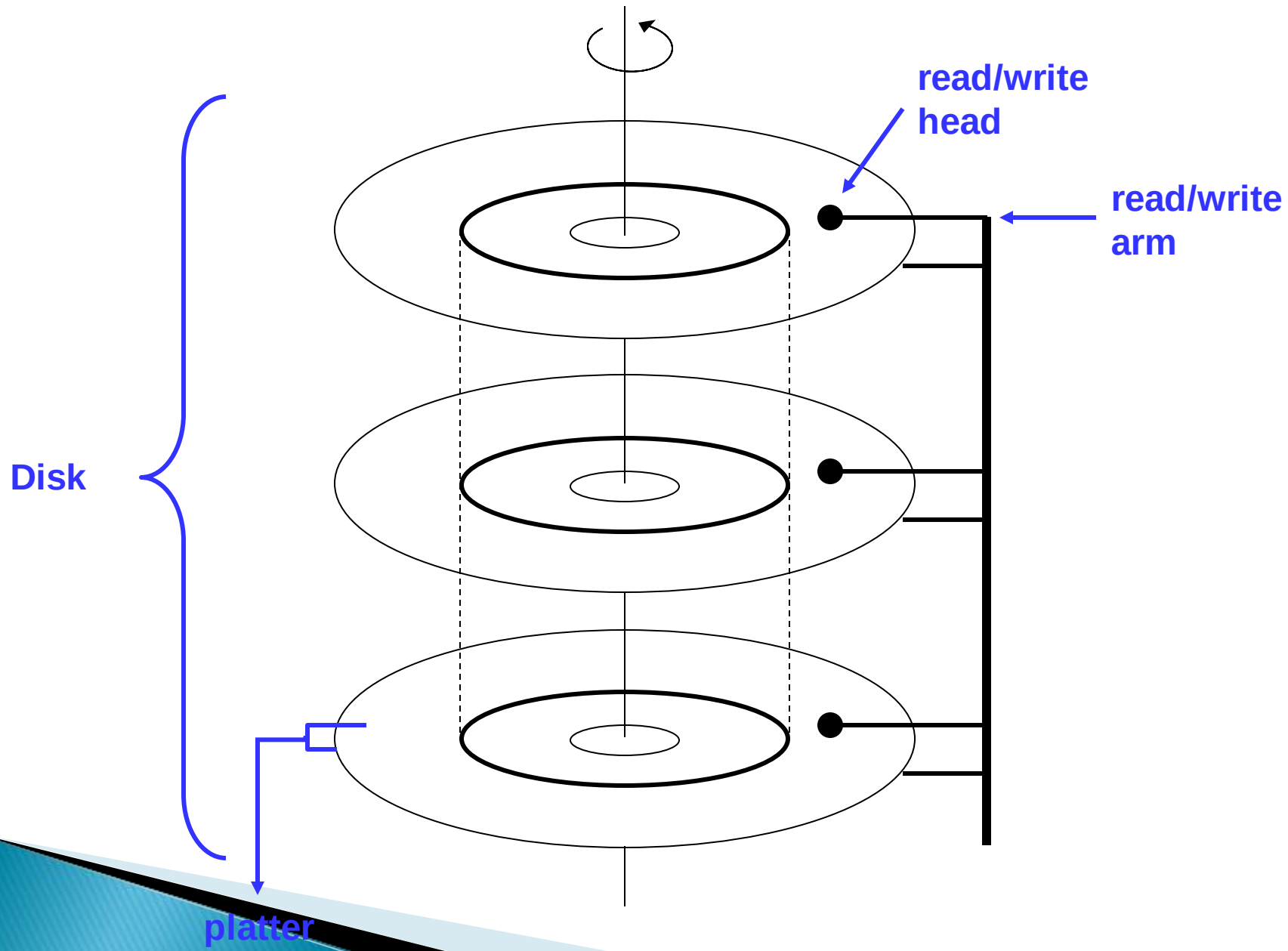
# Introduction

- I/O management – Messiest part of the design of an OS due to a variety of devices attached to a system and speed mismatch.
- With time processor speeds are increasing and so is the internal memory access speeds.
- But I/O remains a significant performance challenge especially in the case of disk storage.

# Disk Organization

- Disk is a secondary storage device that is used to store data.
- Disks provide a means to store a large amount of information for modern computer. Examples : Hard Disk
- A Disk is usually divided into TRACKS, CYLINDERS AND SECTORS .
- Hard disks drives are organized as a concentric stack of disks or ‘platters’.
- Each platter has 2 surfaces and two read/write heads for each surface.

# Multi-platter Disk



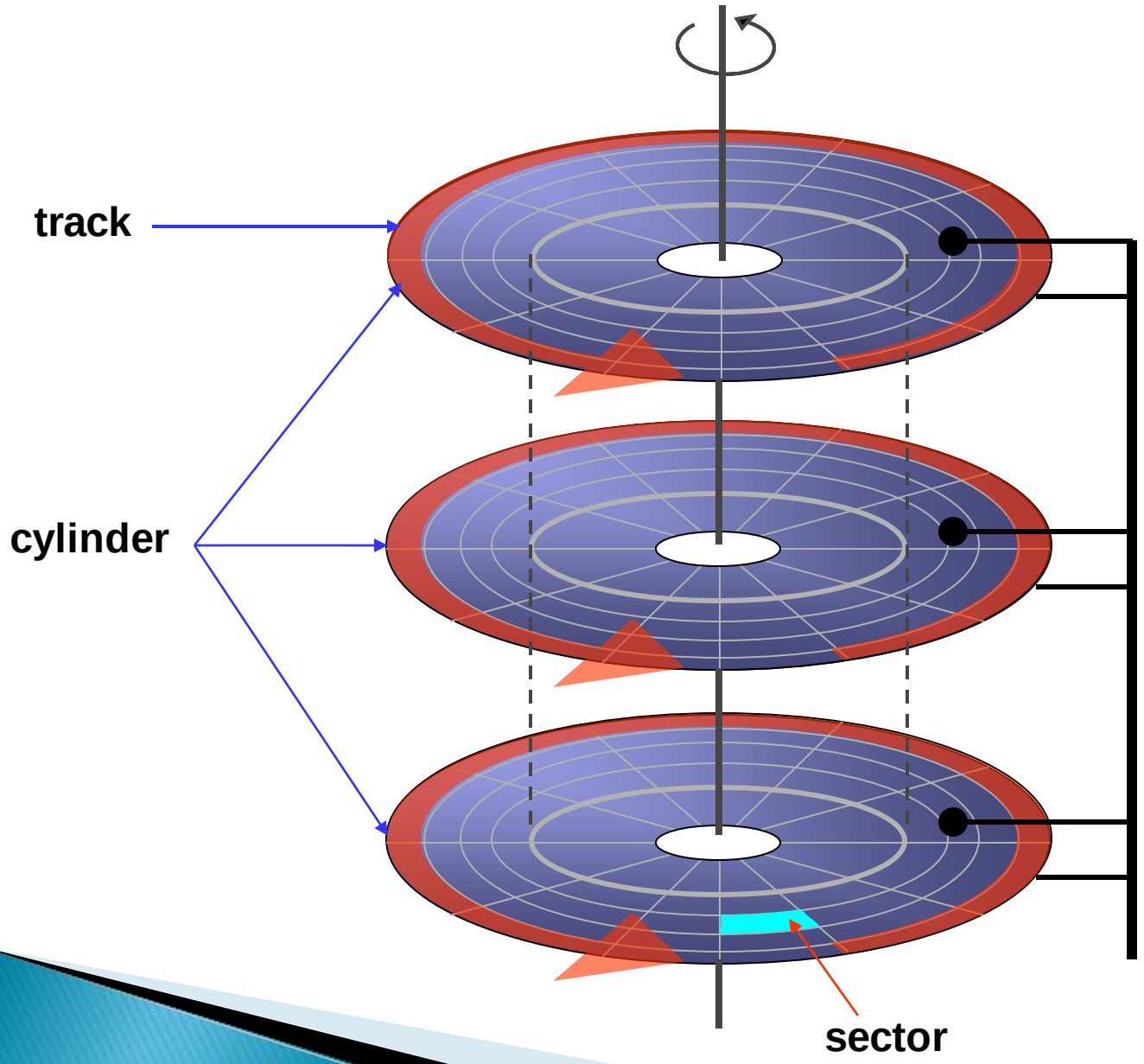
# Disk Organization

- Tracks- circular areas of the disk
- Sectors- Divide tracks into sections
- Cylinders- Logical groupings of the same track on each disk surface in a disk unit.

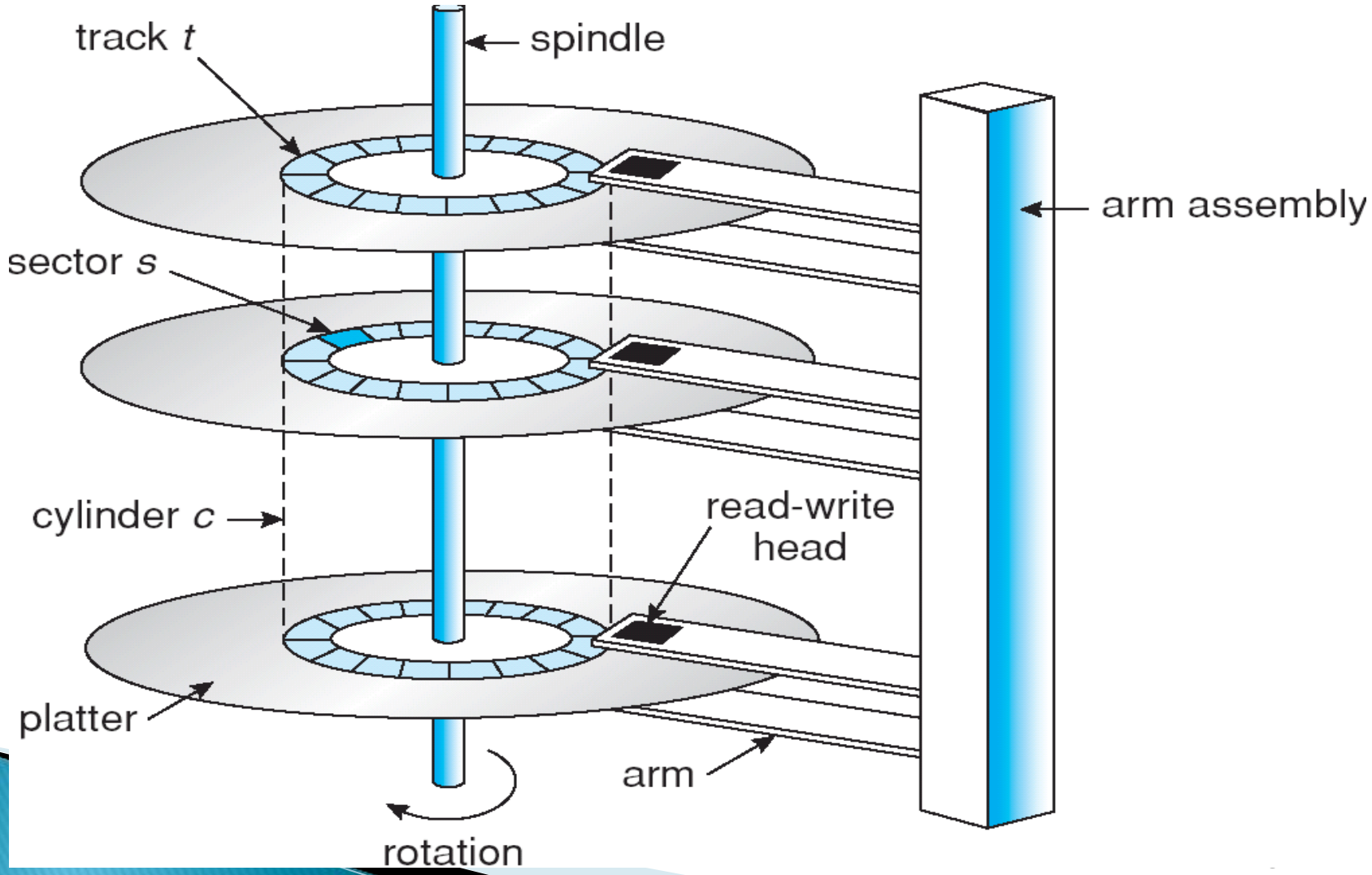
OR

- All the tracks with the same radius are known as a CYLINDER.

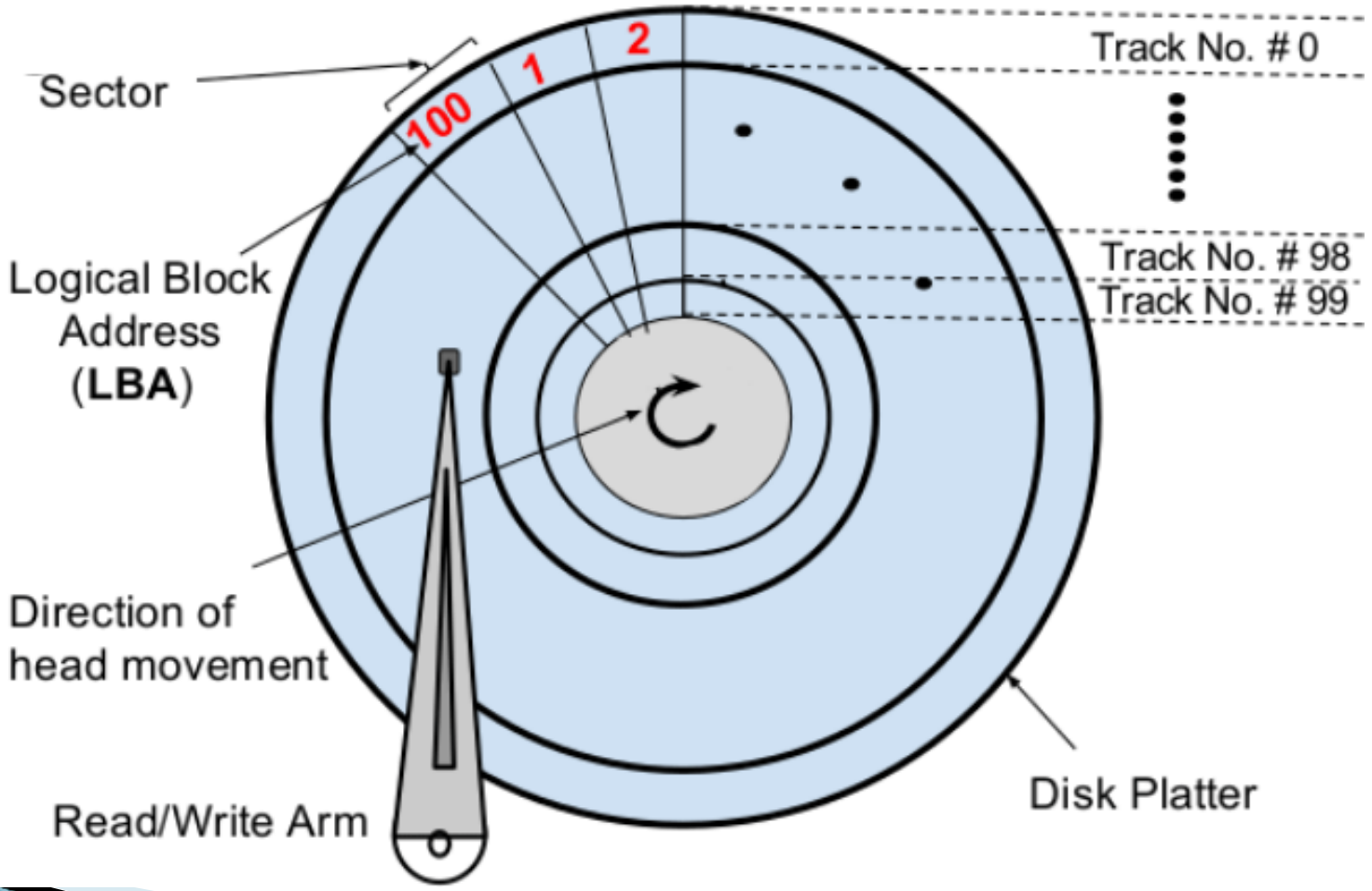
# Multi-platter Disk



# Moving-head Disk Mechanism



# Disk Organization



# Structure of the Disk

- Disk drives are addressed as large 1-dimensional arrays of *logical blocks*, where the logical block is the smallest unit of transfer.

- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially.
  - Sector 0 is the first sector of the first track on the outermost cylinder.
  - Numbering physical sectors within a track starts with 1. Sector 1 Track 0 Sector 2 Track 0
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost.

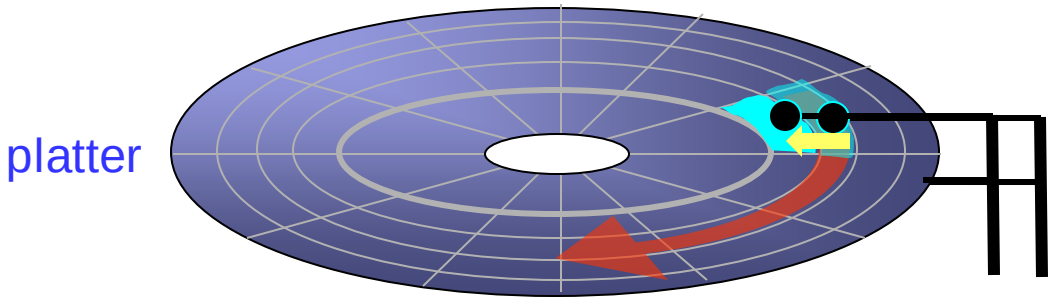
# Disk performance parameters

## Seek Time

Measures the amount of time required for the read/write heads to move between tracks over the surface of the platters

`0.0 ms`

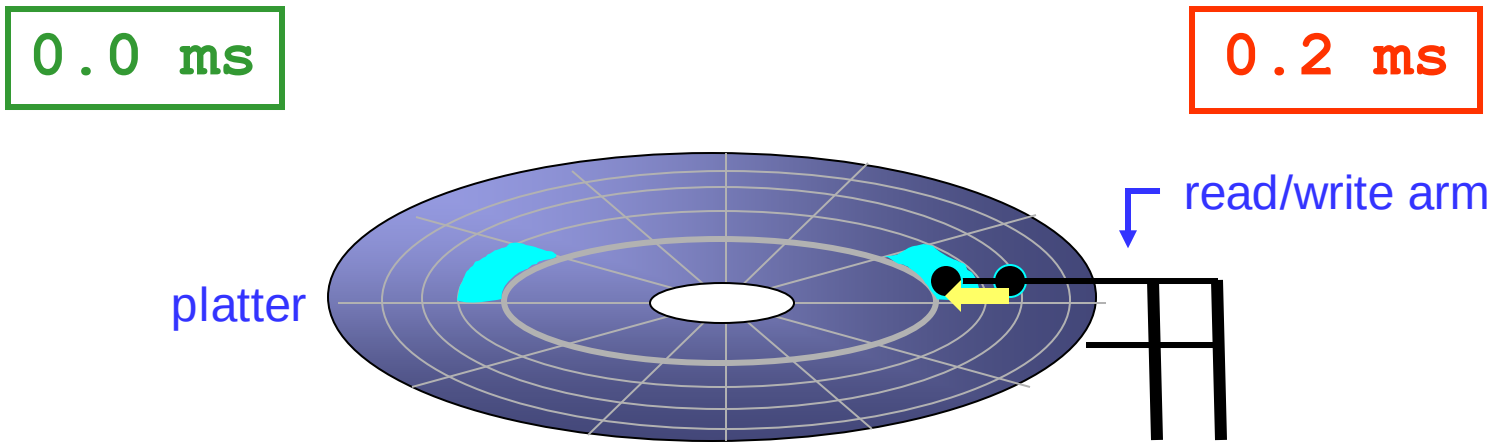
`0.8 ms`



# Disk performance parameters

## Rotational delay

After the read/write arm has picked the right track, the time it takes for the requested sector (on that track) to come under the read/write head



# Disk performance parameters

- To perform a read and write the head must be positioned at the desired track and at the beginning of the desired sector on that track.
- Assuming a moving head system , the time it takes to position the head at the desired track is called as the seek time.
- Next , once the track is selected, the disk controller waits until the appropriate sector rotates to line up with the head.
- The time it takes for the beginning of the sector to reach the head is called the rotational delay.
- Once the head is in position , the read /write is performed as the sector moves under the head. This is the data transfer portion of the operation i.e the transfer time.

# Transfer Time

**Transfer time** = the time it takes to transfer a block of bits, typically a sector, under the read/write head.

**Hence :**

**The total access time =**

**Seek time + Rotational delay + Transfer time**

# Disk Scheduling Policies

- In a multiprogramming environment, there will be I/O requests(read/write) from multiple processes competing for the same disk.
- Seek time is very crucial.
- When there are multiple requests to access the disk, the requests are in the form of which track which sector.
- Disk scheduler is that component of the operating system which decides in which order the requests from processes for disk access should be served.
- There are several disk scheduling policies that the OS uses :

# Disk Scheduling Policies

- In a multiprogramming environment, the operating system maintains a queue of requests from various processes for the disk.
- Assume a disk has 200 tracks.
- The requested tracks, in the order received by the disk scheduler are ------→

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 98 | 183 | 37 | 122 | 14 | 124 | 65 | 67 | |

Currently the head is on track no. 53

# Disk Scheduling Policies

- The disk scheduling policies :

1. FCFS
2. SSTF
3. SCAN
4. LOOK

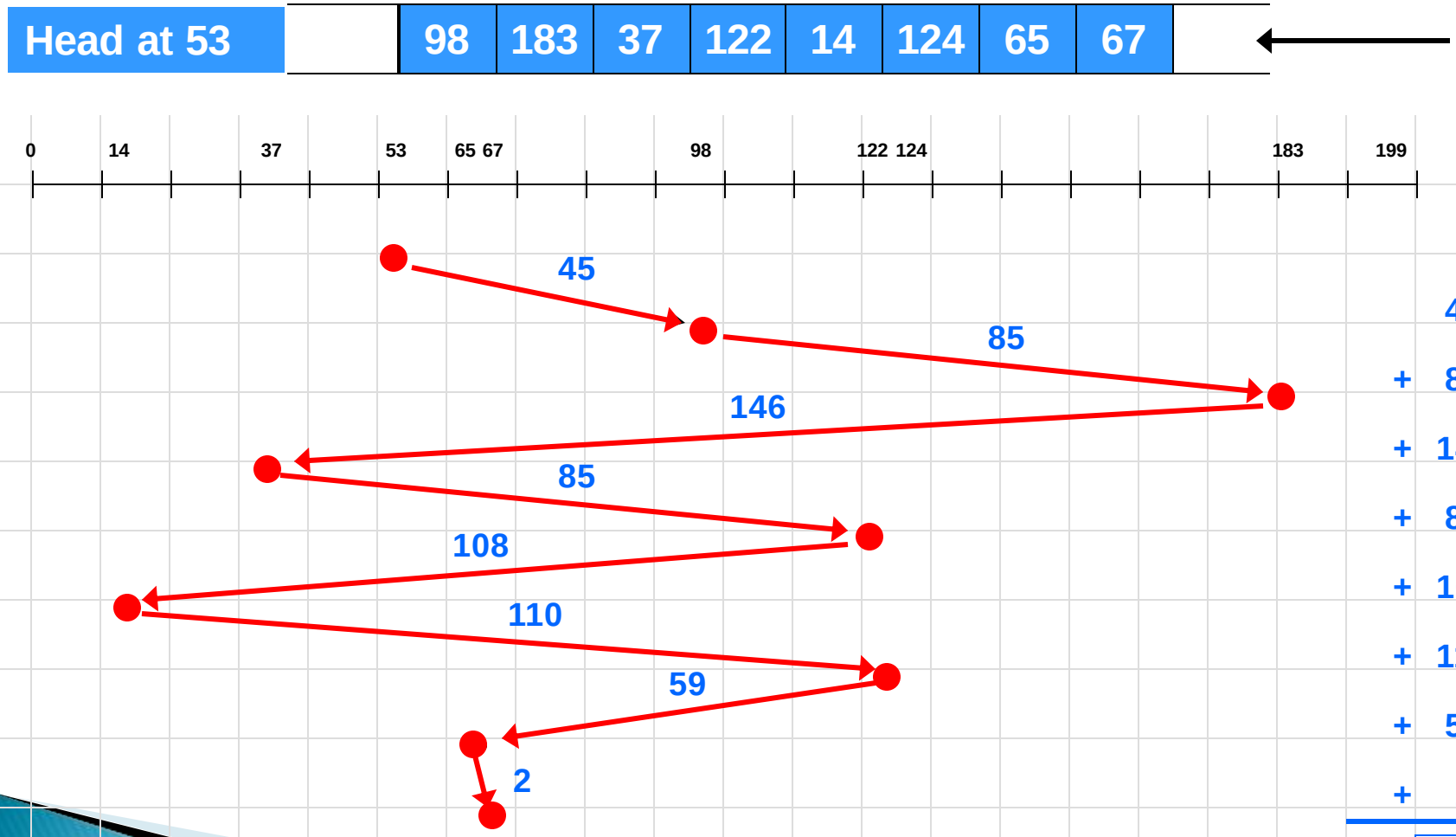
# Disk Scheduling Policies

1. **First come first serve**

- Simplest form of scheduling
- Services the requests in sequential order in the queue.
- Fair to all processes but does not provide the fastest service time.

# Disk Scheduling

## FCFS/FIFO



# Disk Scheduling Policies

- There is a total head movement of 640 tracks.
- Huge swing from 122 to 14 and then back to 124 with this policy.
- If the requests for tracks 37 and 14 could be serviced together , and so also 122 and 124, then the total head movement could be decreased substantially and hence there would be improvement in performance.

# Disk Scheduling Policies

2. Shortest seek time first :

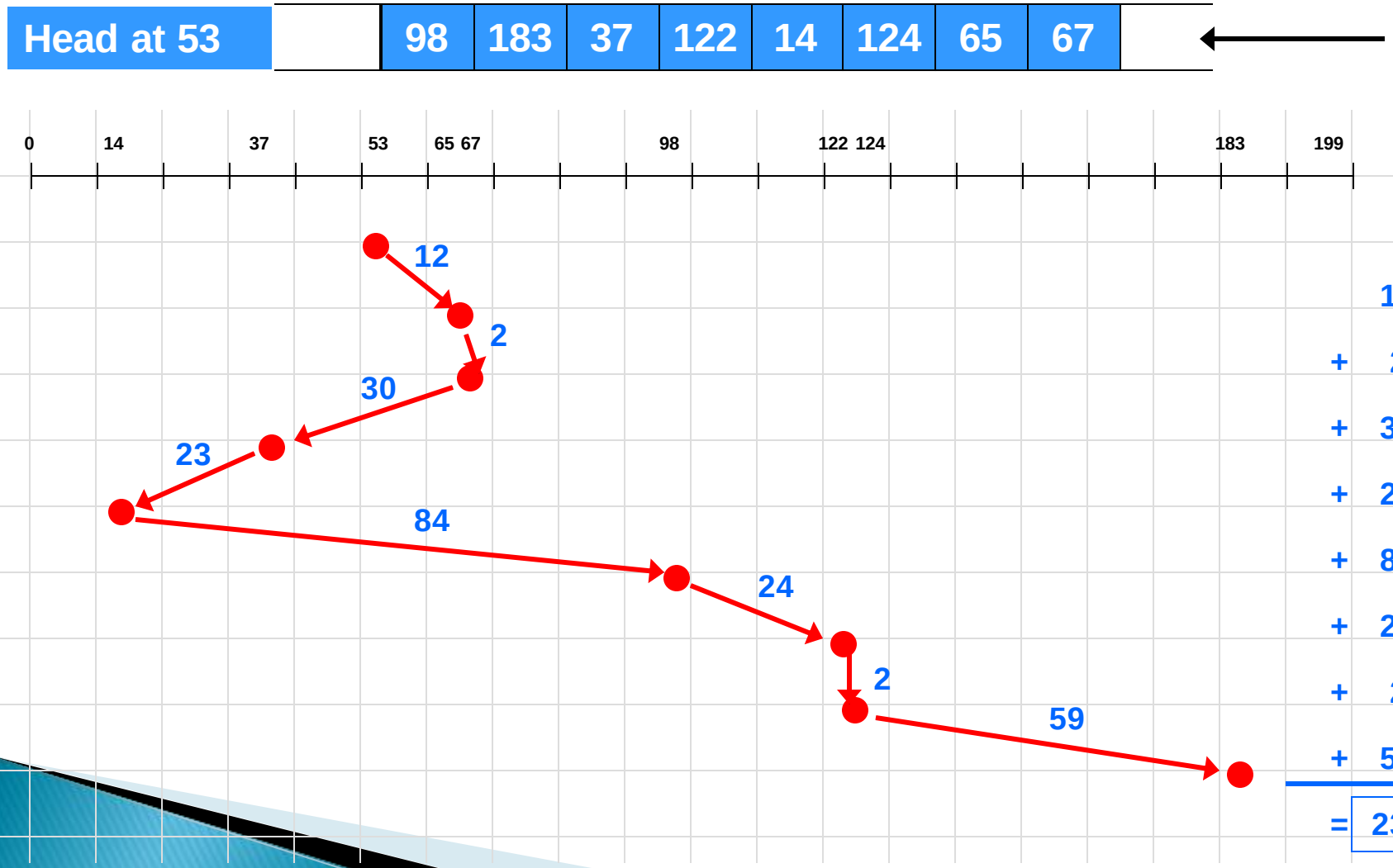
- Services the requests close to the current head position, before moving the head far away to service other requests.

Currently the head is on track no. 53

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 98 | 183 | 37 | 122 | 14 | 124 | 65 | 67 | |

# Disk Scheduling

## Shortest Seek Time First - SSTF



# Disk Scheduling Policies

3. SCAN scheduling

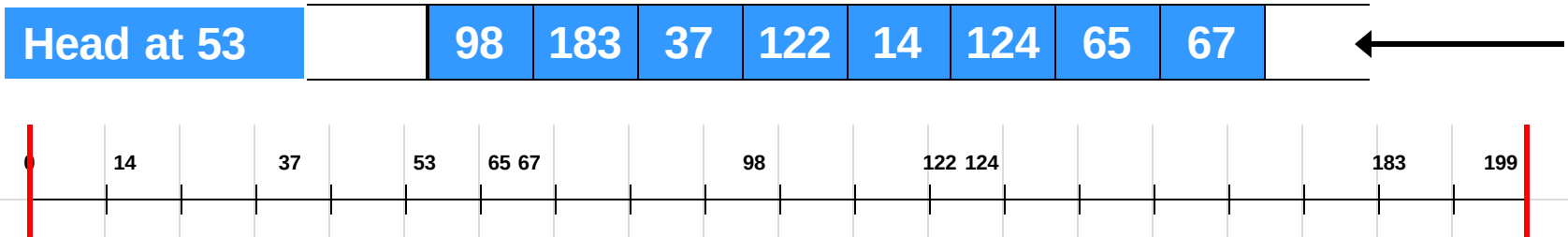
- In the SCAN algorithm, the disk arm starts at one end of the disk and moves toward the other end, servicing requests as it reaches each track, until it gets to the other end of the disk.
- At the other end, the direction of head movement is reversed, and servicing continues.
- The head continuously scans back and forth across the disk.
- The SCAN algorithm is sometimes called the elevator algorithm, since the disk arms behaves just like an elevator in a building, first servicing all the requests going up and then reversing to service requests the other way.

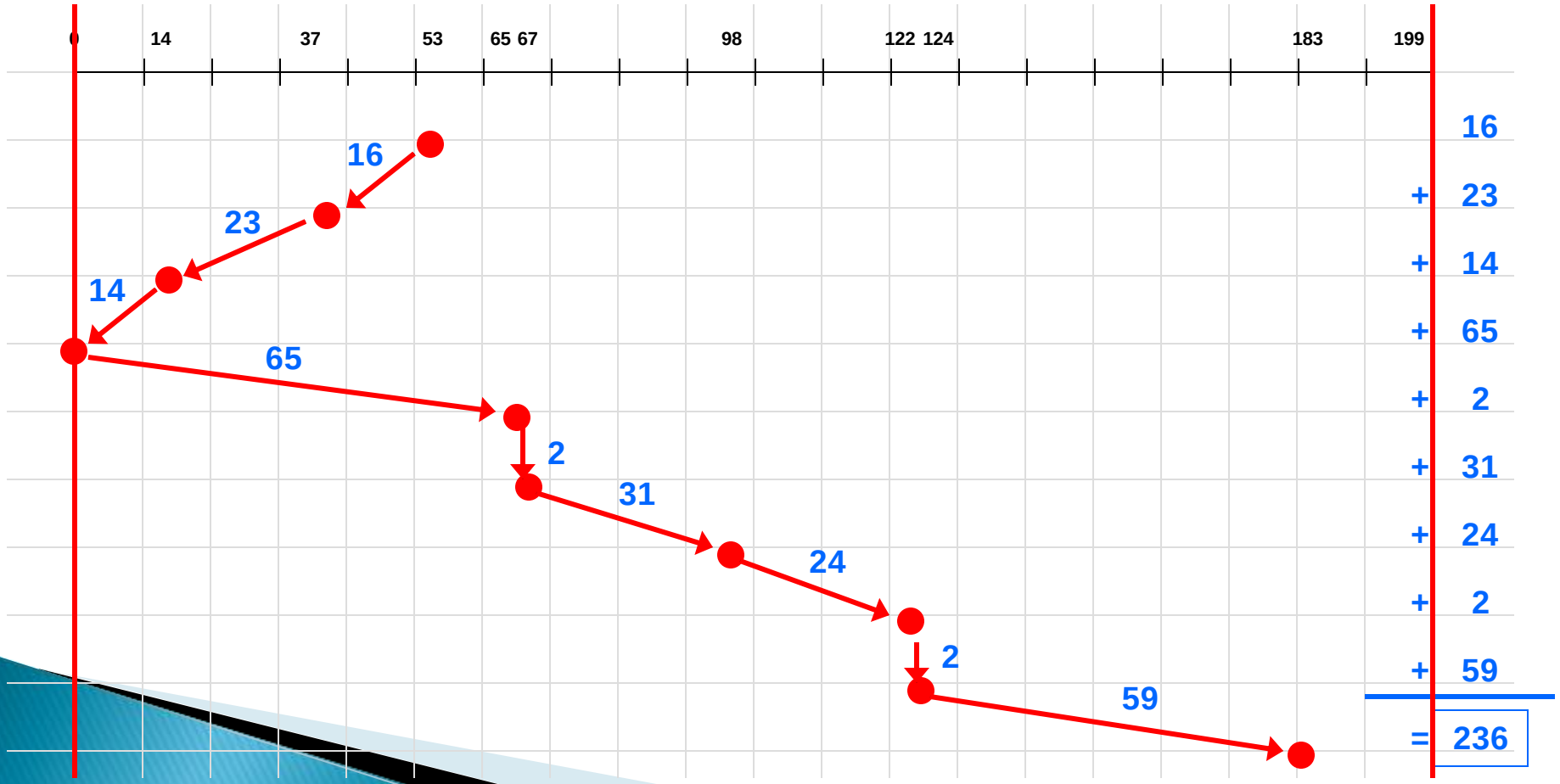
# Disk Scheduling Policies

- If a request arrives in the queue just in front of the head, it will be serviced almost immediately.
- If a request arriving just behind the head will have to wait until the arm moves to the end of the disk, reverses direction, and comes back.

# Disk Scheduling

## Scan



# Disk Scheduling Policies

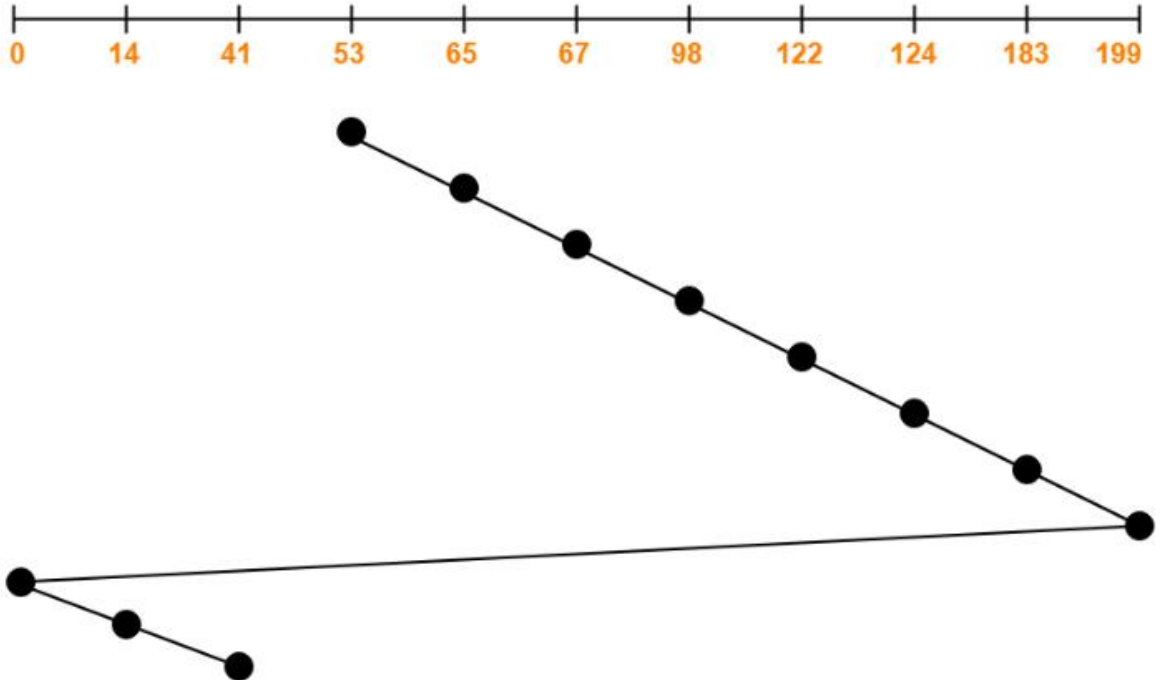
C-scan

- Circular SCAN (C-SCAN) scheduling is a variant of SCAN designed to provide a more uniform wait time.
- Like SCAN, CSCAN 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.
- Treats the tracks as a circular list that wraps around from the final track to the first one

# C-Scan eg.

Currently the head is on track no. 53

Consider a disk queue with requests for I/O to blocks on cylinders 98, 183, 41, 122, 14, 124, 65, 67. The C-SCAN scheduling algorithm is used. The head is initially at cylinder number 53 moving towards larger cylinder numbers on its servicing pass.




Total head movements incurred while servicing these requests

$= (65 - 53) + (67 - 65) + (98 - 67) + (122 - 98) + (124 - 122) + (183 - 124) + (199 - 183) + (199 - 0) + (14 - 0) + (41 - 14)$

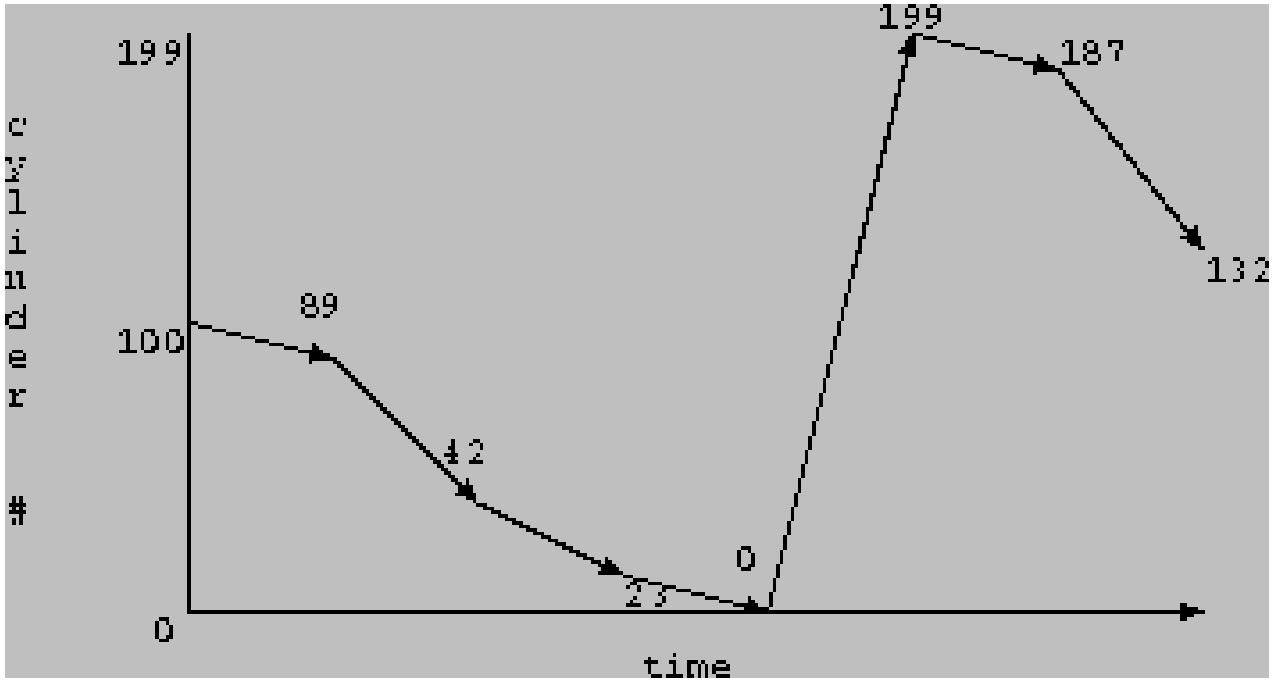
$= 12 + 2 + 31 + 24 + 2 + 59 + 16 + 199 + 14 + 27$

$= 386$

·**Work Queue**: 23, 89, 132, 42, 187

·there are 200 cylinders numbered from 0 – 199

·the disk head stars at number 100




C-SCAN

# Disk Scheduling Policies

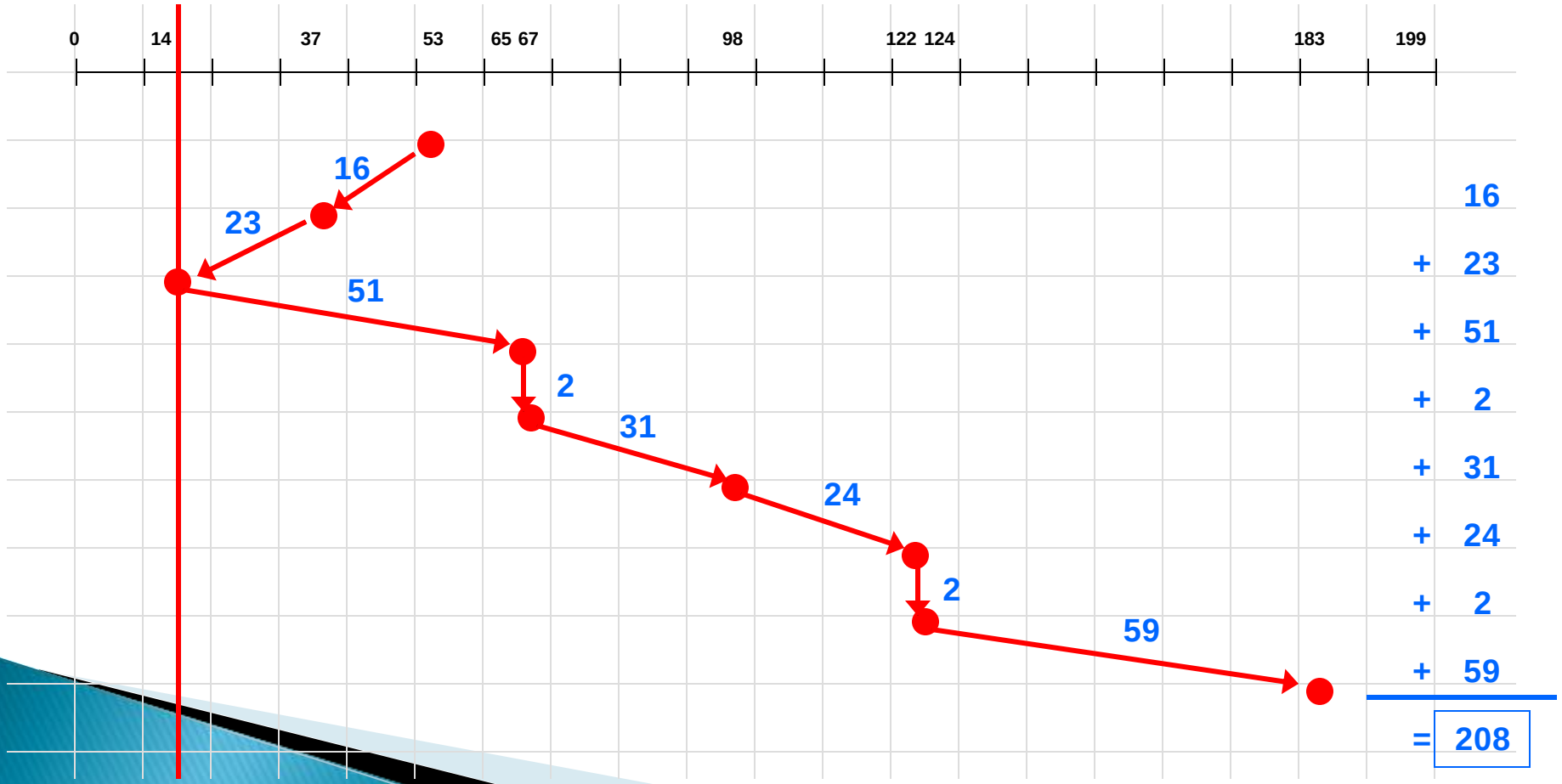
4. LOOK

- The arm goes only as far as 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.

- Look Algorithm is actually an improves version of SCAN Algorithm.
- In this algorithm, the head starts from first request at one side of disk and moves towards the other end by serving all requests in between.
- After reaching the last request of one end, the head reverse its direction and returns to first request, servicing all requests in between.
- Unlike SCAN, in this the head instead of going till last track, it goes till last request and then direction is changed.

# Disk Scheduling

## Look

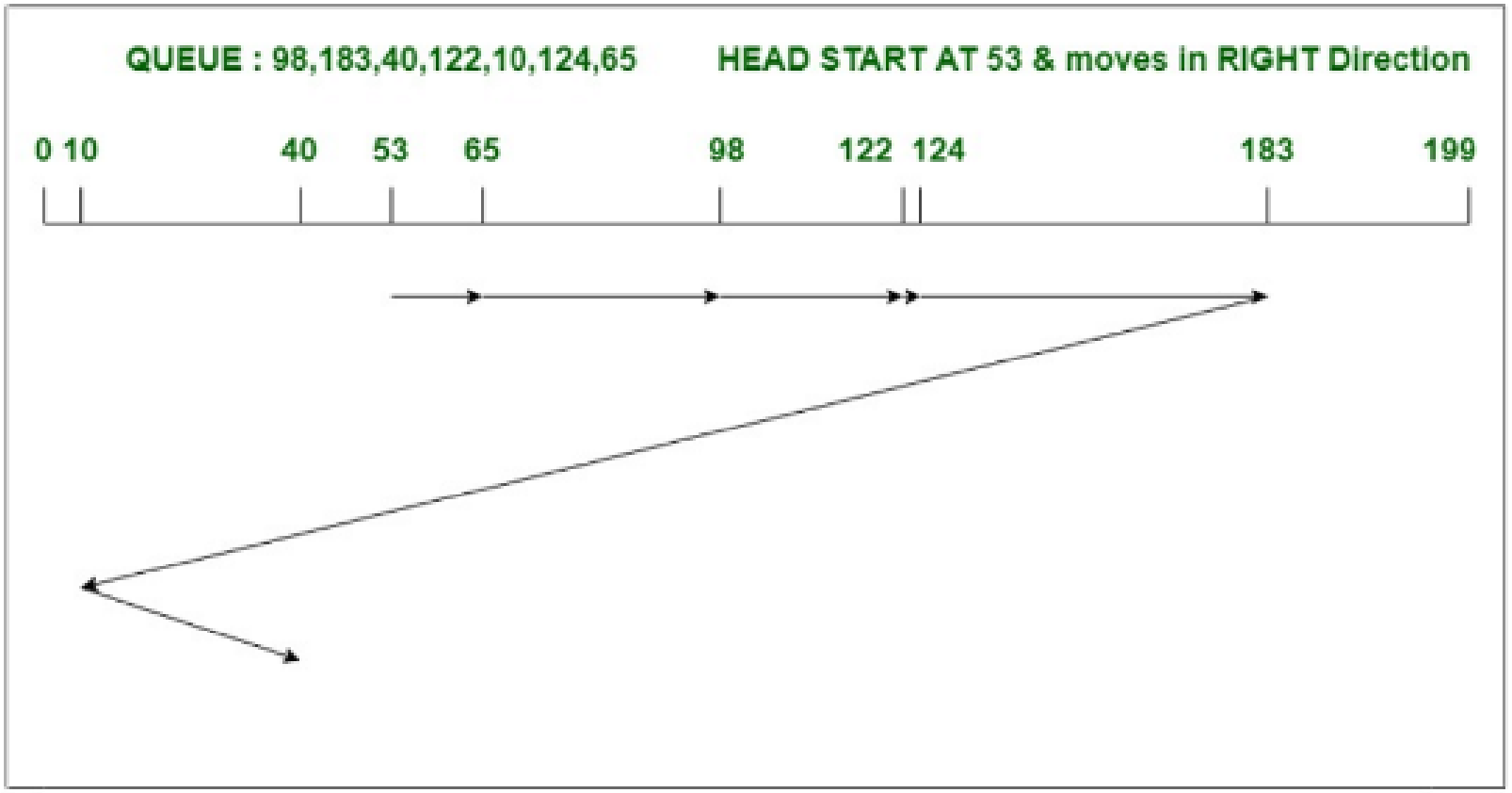
# C-Look

- In this algorithm, the head starts from first request in one direction and moves towards the last request at other end, serving all request in between.
- After reaching last request in one end, the head jumps in other direction and move towards the remaining requests and then satisfies them in same direction as before.
- Unlike LOOK, it satisfies requests only in one direction.

# C-Look

- Consider a disk with 200 tracks (0-199) and the disk queue having I/O requests in the following order as follows:
- 98, 183, 40, 122, 10, 124, 65.
- The current head position of the Read/Write head is 53 and will move in Right direction .
- Calculate the total number of track movements of Read/Write head using C-LOOK algorithm.

# C-Look




Total head movements,

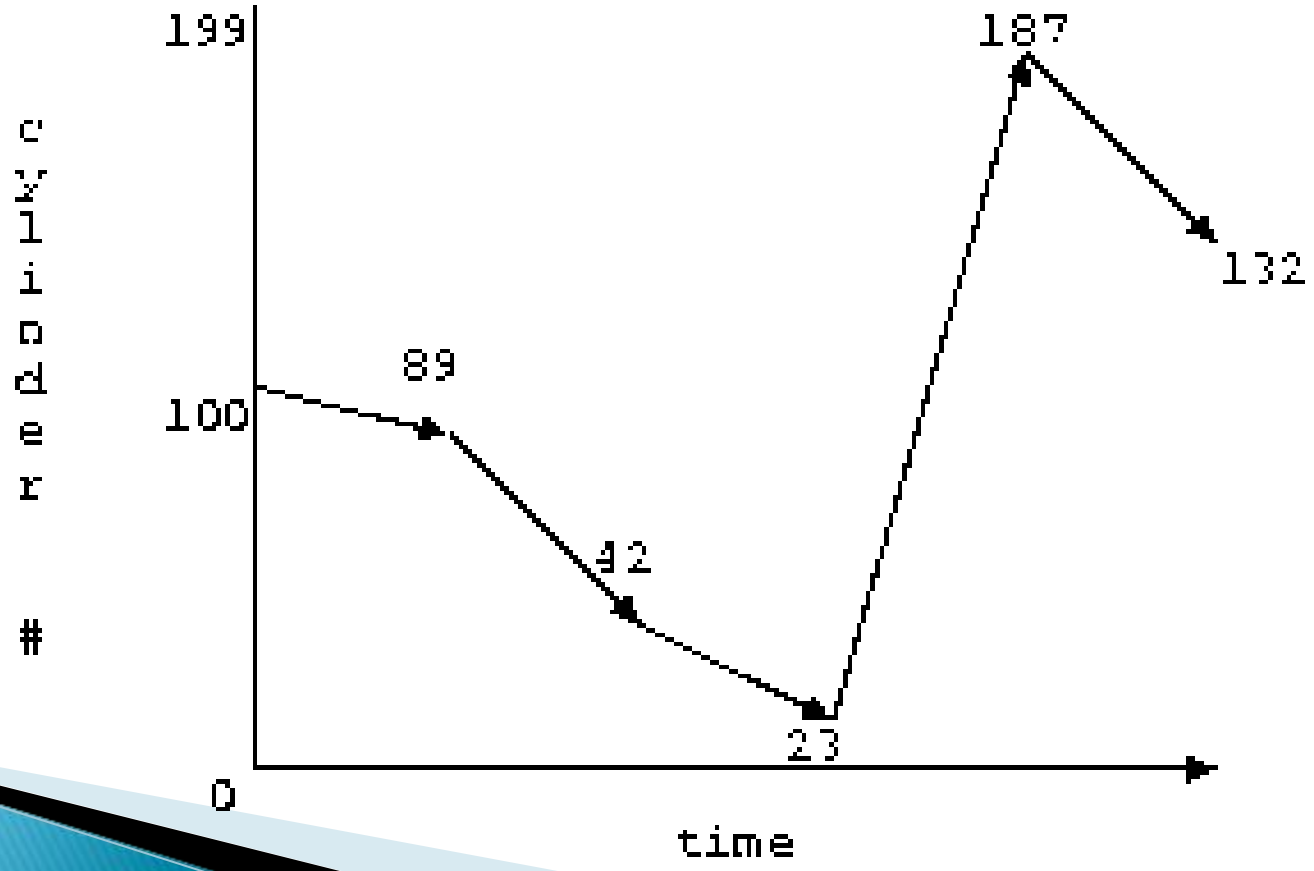
= (65-53)+(98-65)+(122-98) +(124-122)+(183-124)+(183-10)+(40-10) = 333

# C LOOK

·**Work Queue**: 23, 89, 132, 42, 187

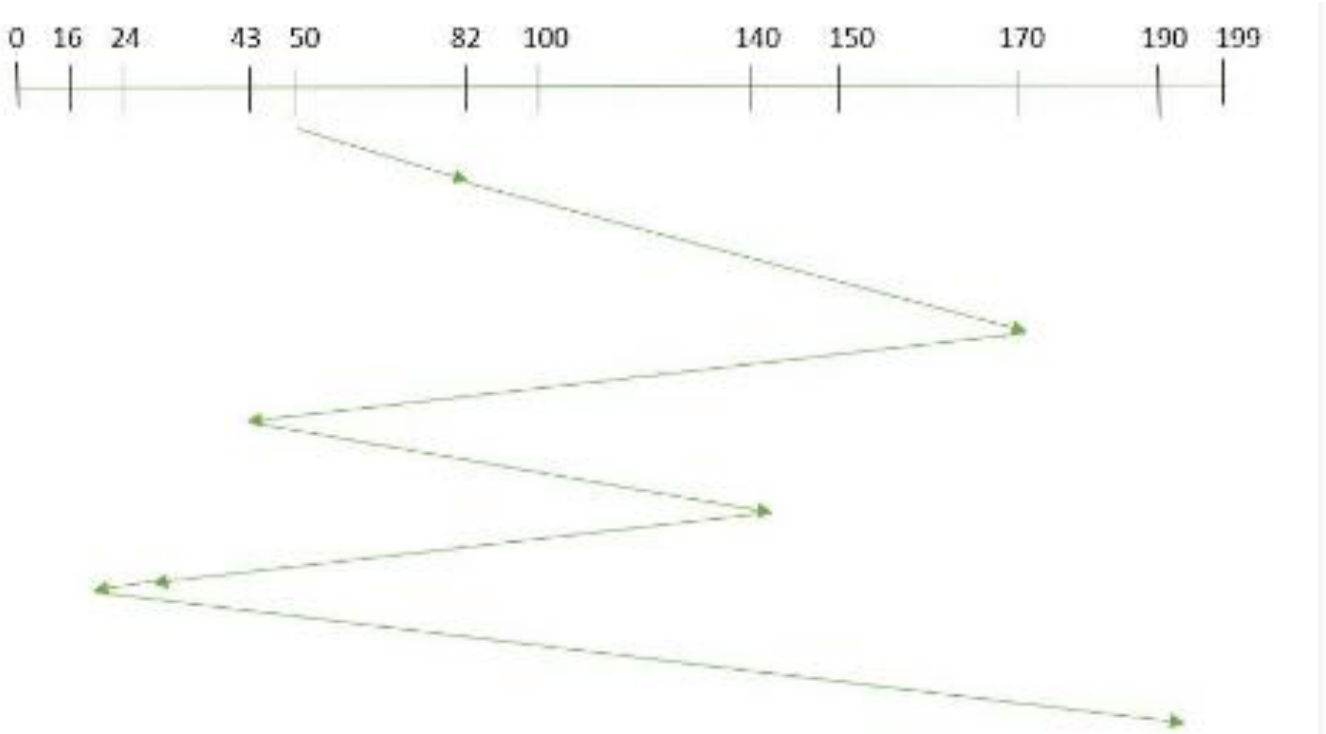
·there are 200 cylinders numbered from 0 – 199

·the disk head stars at number 100



# Example: FCFS

- Suppose the order of request is- (82,170,43,140,24,16,190)
  And current position of Read/Write head is at track: 50



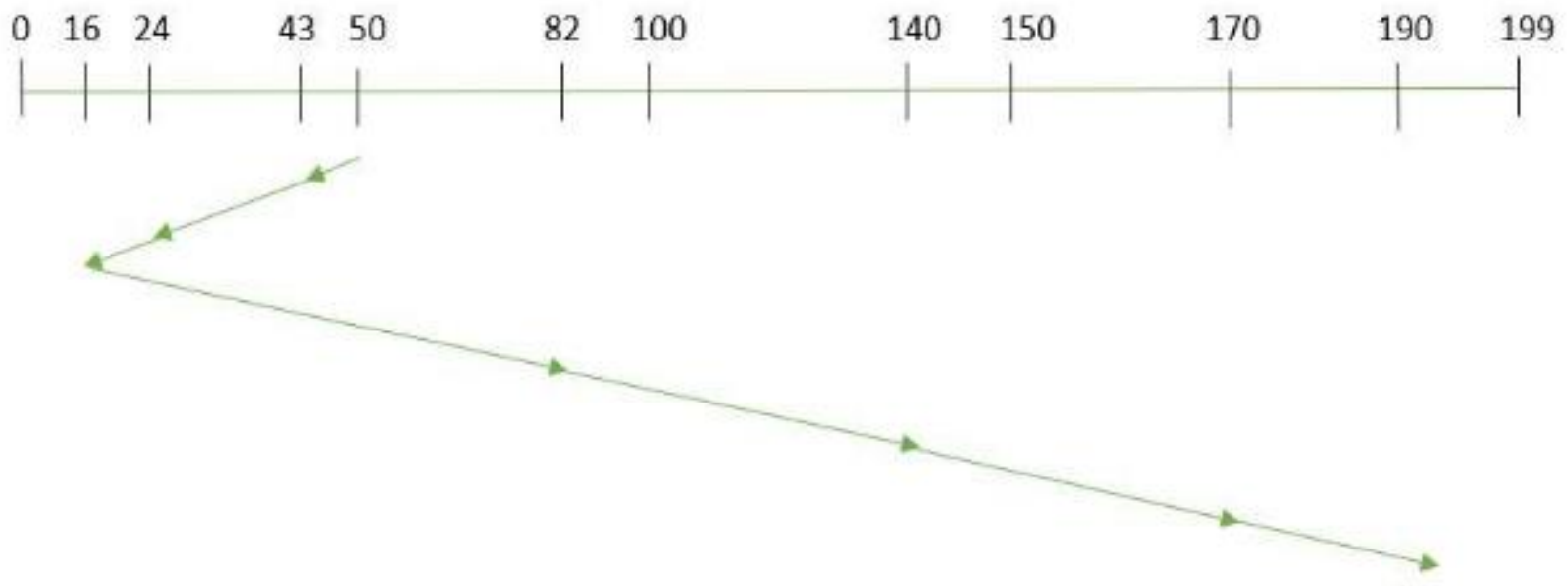

Total head movement

=(82-50)+(170-82)+(170-43)+(140-43)+(140-24)+(24-16)+(190-16)

=642

# Example: SSTF

- Suppose the order of request is- (82,170,43,140,24,16,190)
  And current position of Read/Write head is at track: 50

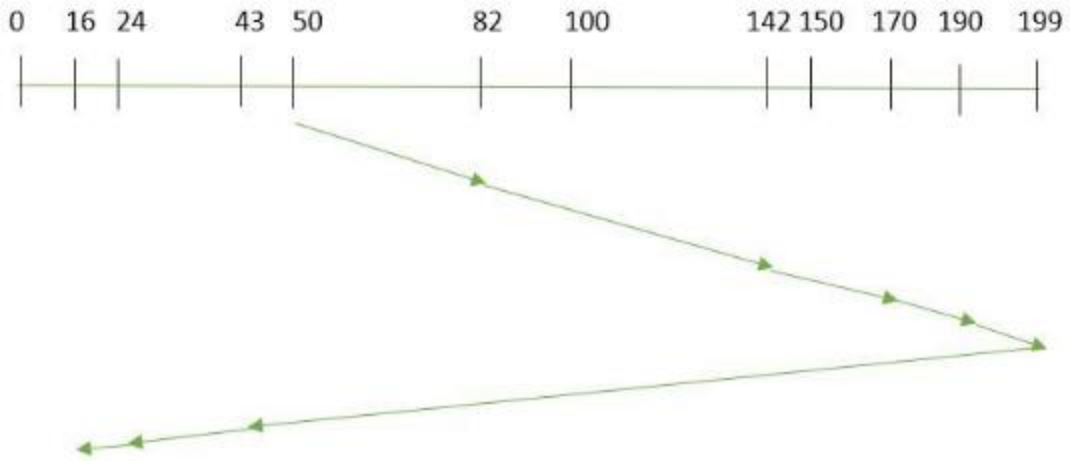



Total head movement

=(50-43)+(43-24)+(24-16)+(82-16)+(140-82)+(170-40)+(190-170)
=208

# Example: Scan

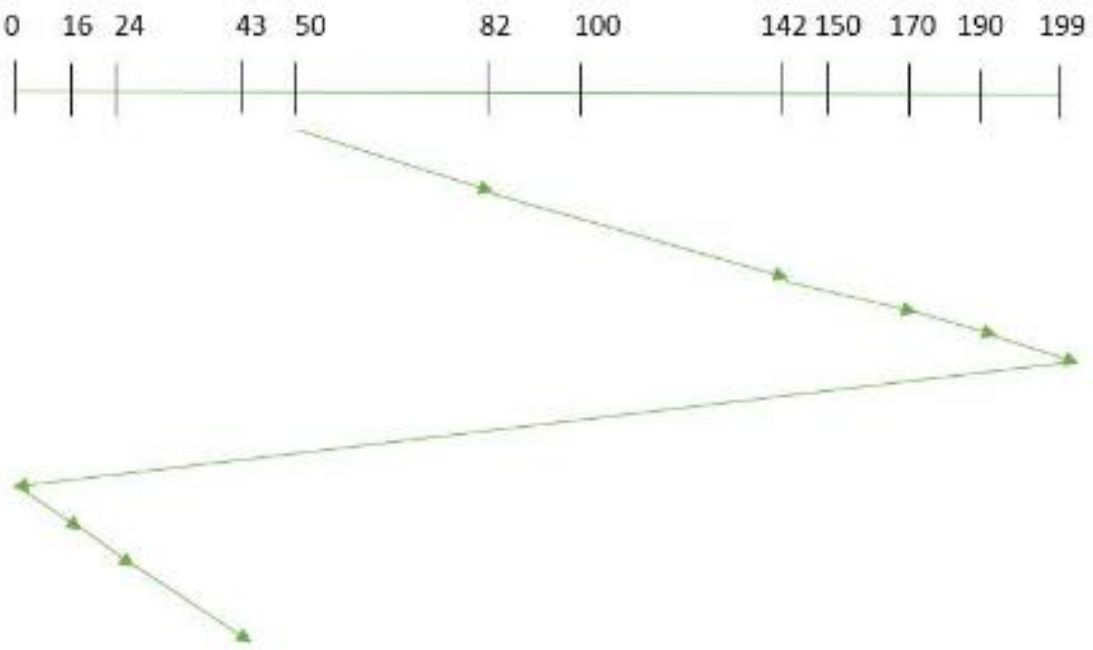
- Suppose the order of request is- (82,170,43,140,24,16,190) And current position of Read/Write head is at track: 50 and it is also given that the disk arm should move “towards the larger value”.




Total head movement =(199-50)+(199-16)
=332

# Example: C-Scan

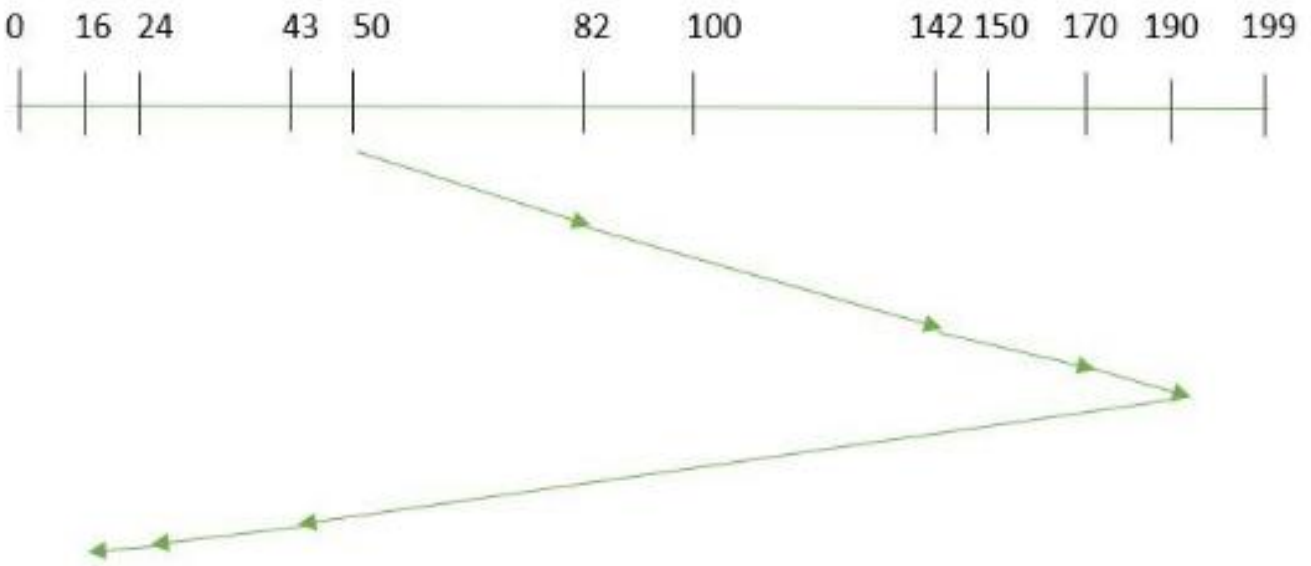
- Suppose the order of request is- (82,170,43,140,24,16,190) And current position of Read/Write head is at track: 50 and it is also given that the disk arm should move “towards the larger value”.




Total Head movement =(199-50)+(199-0)+(43-0)
=391

# Example: LOOK

- Suppose the order of request is- (82,170,43,140,24,16,190) And current position of Read/Write head is at track: 50 and it is also given that the disk arm should move “towards the larger value”.



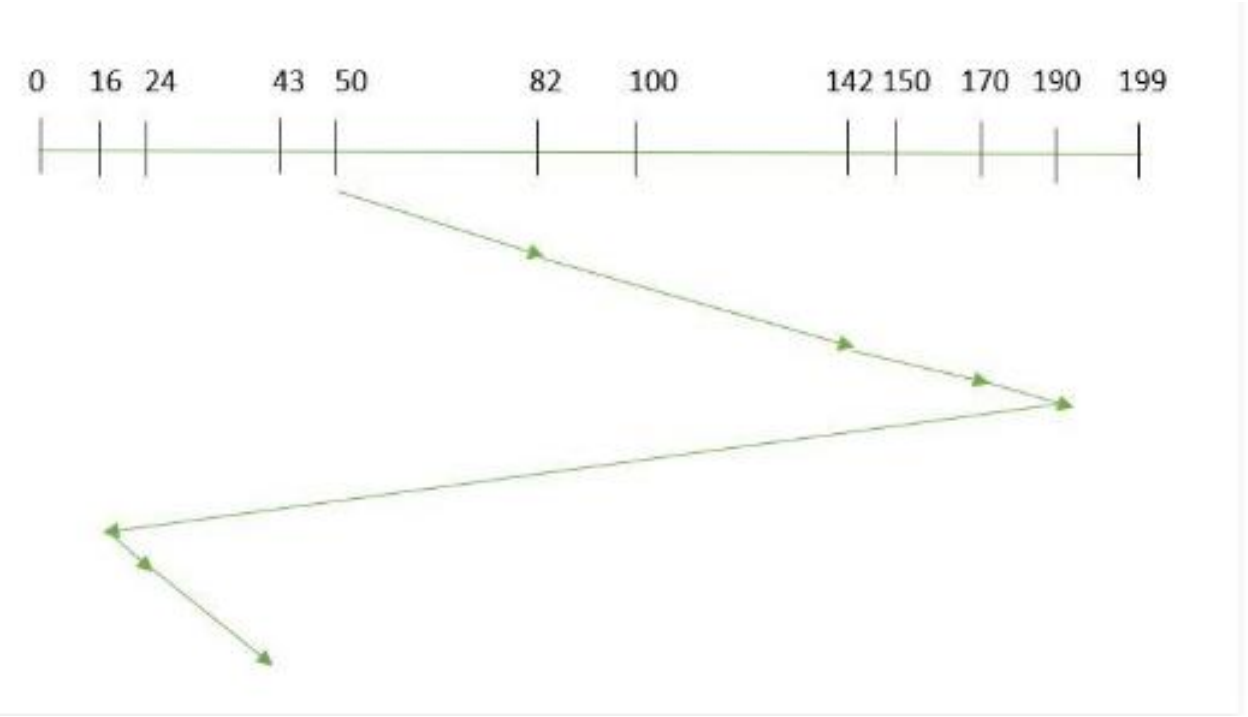

Total Head movement

=(190-50)+(190-16)

=314

# Example: C-LOOK

- Suppose the order of request is- (82,170,43,140,24,16,190) And current position of Read/Write head is at track: 50 and it is also given that the disk arm should move "towards the larger value".




Total Head movement =(190-50)+(190-16)+(43-16)
=341