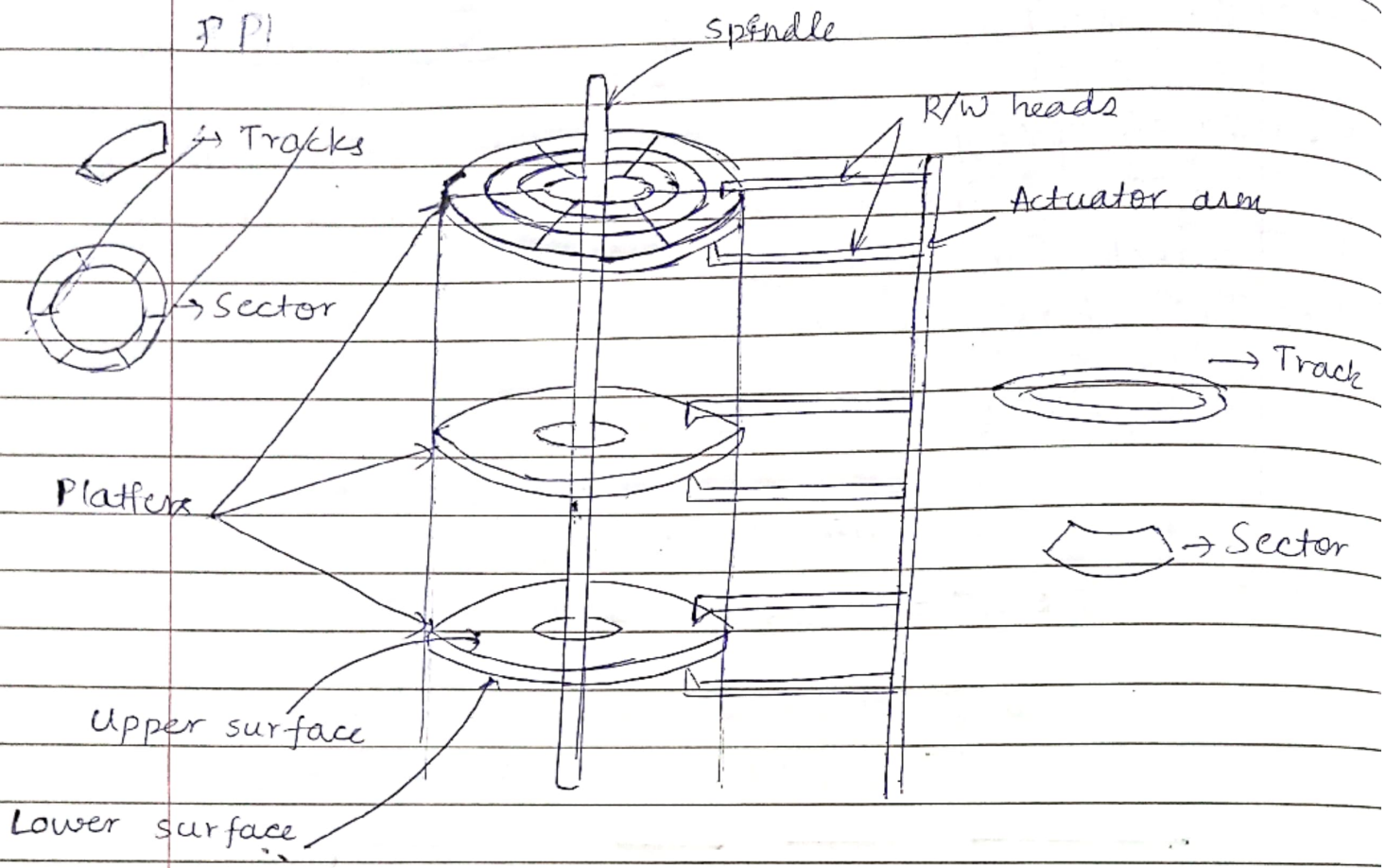
Disk Architecture



Platter → Surface → Track → Sector → data

Disk size = P X S X T X S X D

Disk Access Time

1) Seek Time : Time taken by R/W head to reach desired target

2) Rotation time : Time taken for one full rotation (360°)

3) Rotational latency : Time taken to reach desired sector (half rotation time)

4) Transfer Time = Data to be transferred / Transfer rate

Transfer rate (Data rate) = (No. of heads (Surfaces)) × (Capacity of one track) × (No. of rotations in one sec.)

5) Disk access time = Seek time + Rotational time + Transfer time + Controller time + Queue time

DISK SCHEDULING ALGORITHMS

Goal: To minimize the seek time

→ FCFS (First come first serve)

→ SSTF (Shortest seek time first)

→ SCAN &

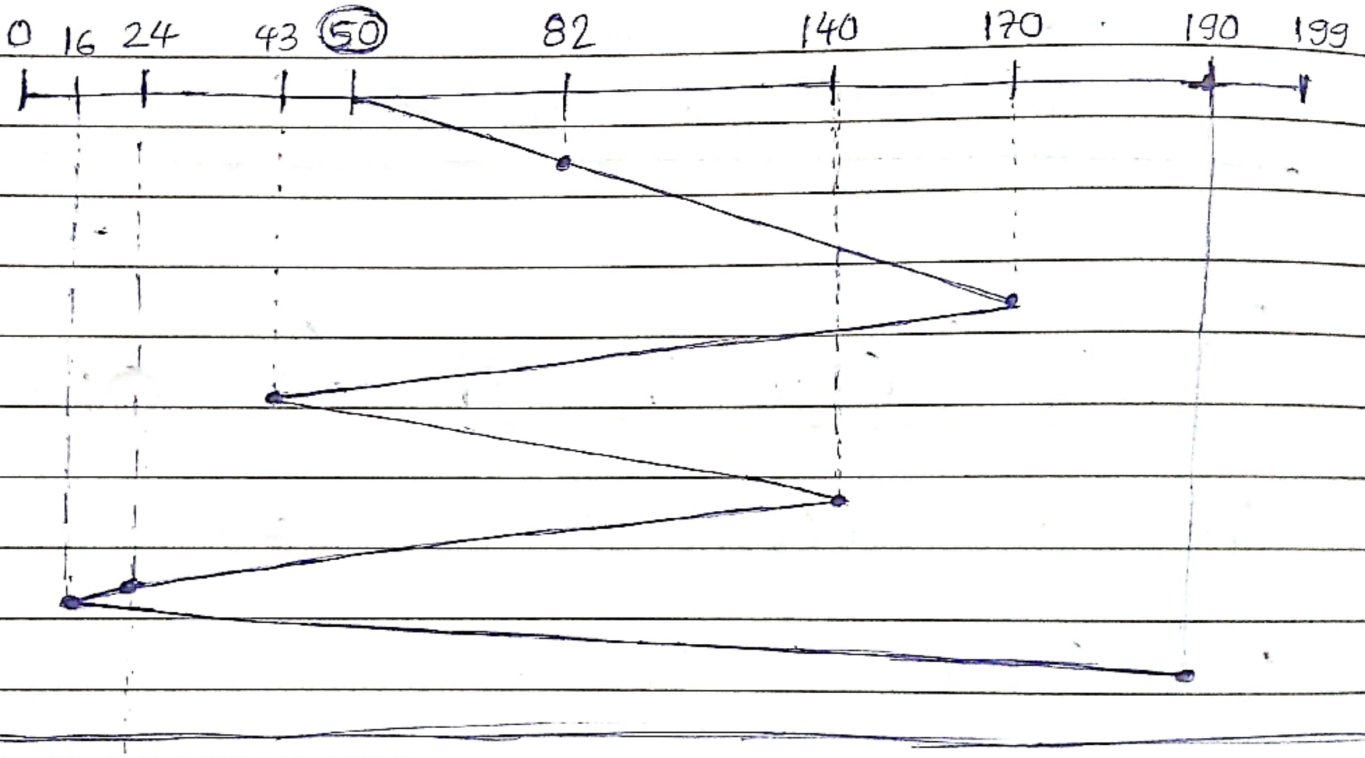
→ LOOK

→ C-SCAN (Circular SCAN)

→ C-LOOK (Circular LOOK)

FCFS

Q A disk contains 200 tracks (0-199), request queue queue contains track nos. 92, 170, 43, 140, 24, 16, 190 respectively. Current pos. of R/W head = 50 Calculate the total no. of tracks movement by R/W head.



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

§ In this question, we need to find out total no. of track movements

∴ Total overhead movement (Total distance covered by disk arm) = (82-50) + (170-82) + (170-43) + (140-43) + (140-24) + (24-16) + (190-16) = 642

Advantages:
→ Fair chance to every request
→ No indefinite postponement

Disadvantages
→ Does not try to optimize seek time
→ May not provide the best possible service

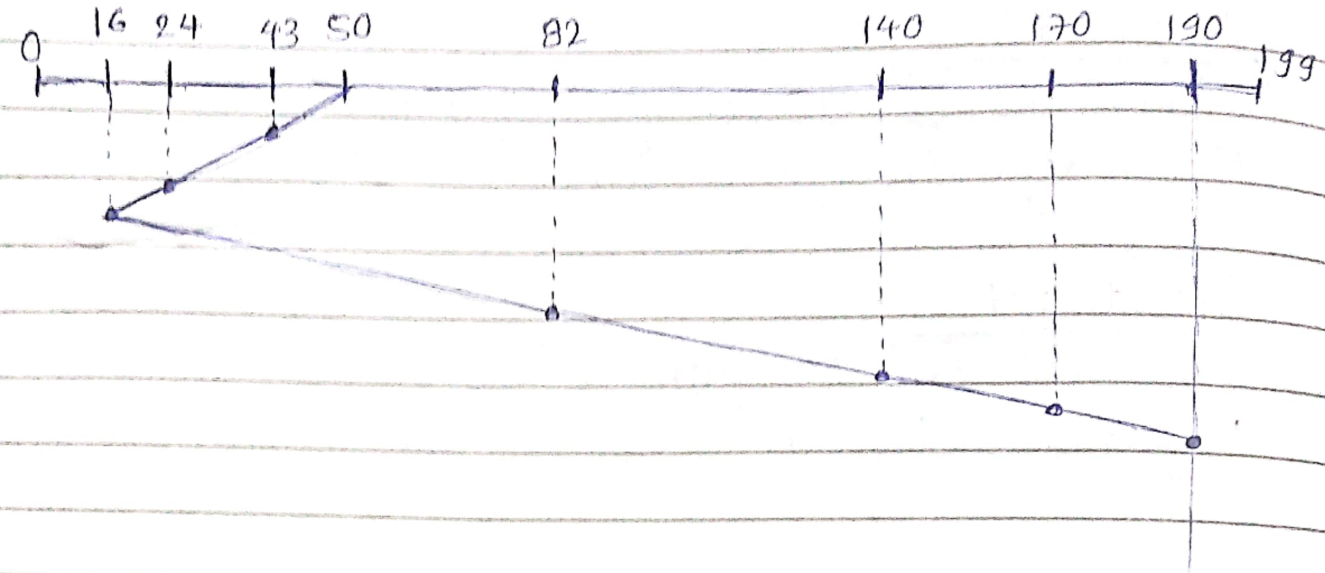
SSTF

Q A disk containing 200 tracks (0-199) request queue containing track no. ~~(0-199)~~ 82, 170, 43, 140, 24, 16, 190 resp. curr. pos. of R/W head = 50

→ Cal. total no. of tracks moved using SSTF
→ If SSTF takes ~~one track~~ 1ms to move from one track to another then total time taken ____ ?

request near the disk arm will get executed first

So first g'll check ~~who is~~ which request is nearest from 50, so 9 get 43
Now we check nearest from 43, i.e. 24
now nearest from 24 is 16
Now out of the left 4 requests nearest one from 16 is 82 & so on



Total overhead movement = (50-16) + (190-16)
= 208

Advantage:
- Average response time decreases
- Throughput increases

Disadvantages:
→ Overhead to calculate seek time in advance
→ Can cause starvation for a request if it has a higher seek time as compared to incoming requests
→ High variance of seek time as SSTF favours only some requests

Total time = 208 × 1 ns = **208 ns**

SCAN:

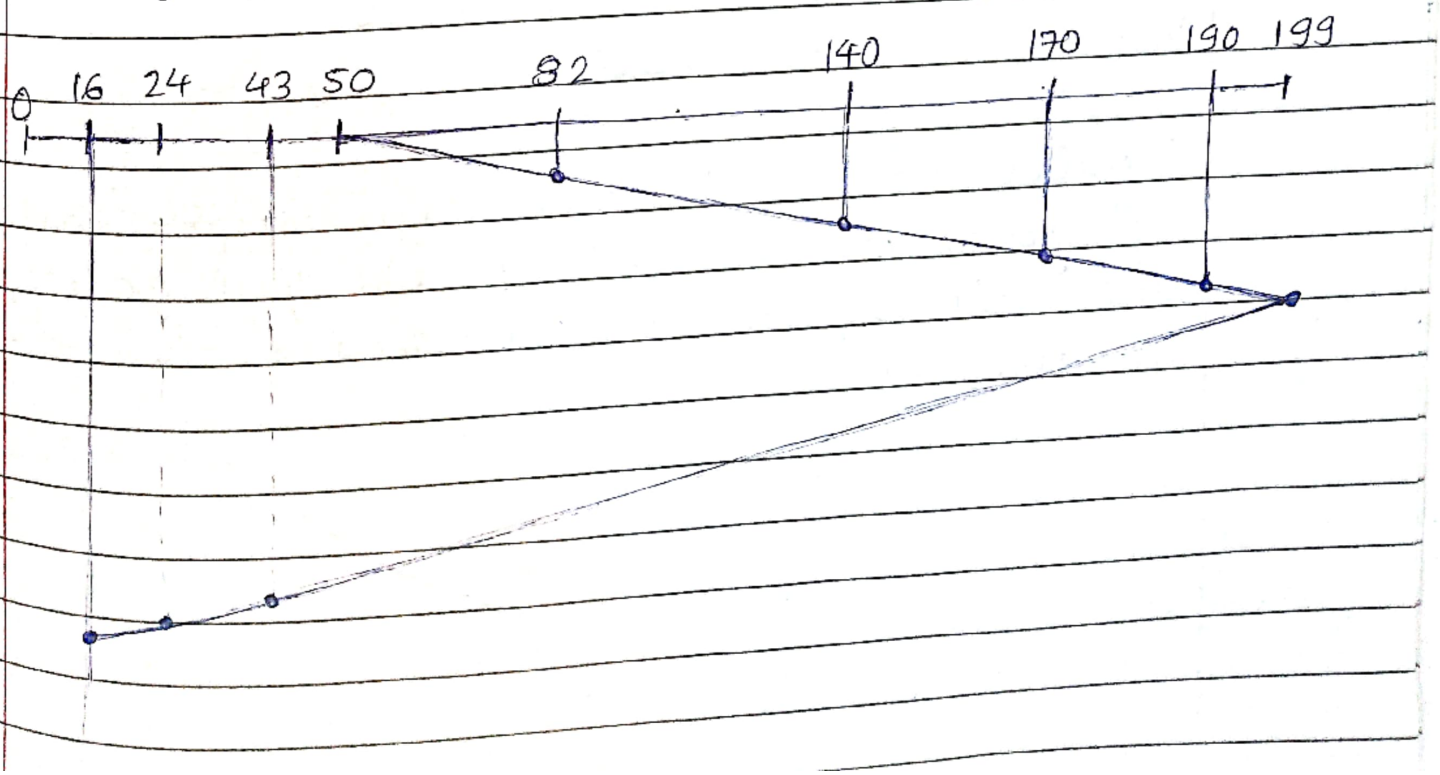
Same previous question

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

Current pos. of R/W head = 50

Disk arm moves in particular direction & services the requests coming in its path & after reaching the end of the disk, it reverses its direction & again services the requests arriving in its path

Note: First we move towards the end of the disk & when the direction of disk arm gets reversed it only goes uptill the last request.



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

Advantages
→ High throughput
→ Low variance of response time
→ Average response time

Disadvantages

→ Long waiting time for requests for locations just visited by disk arm

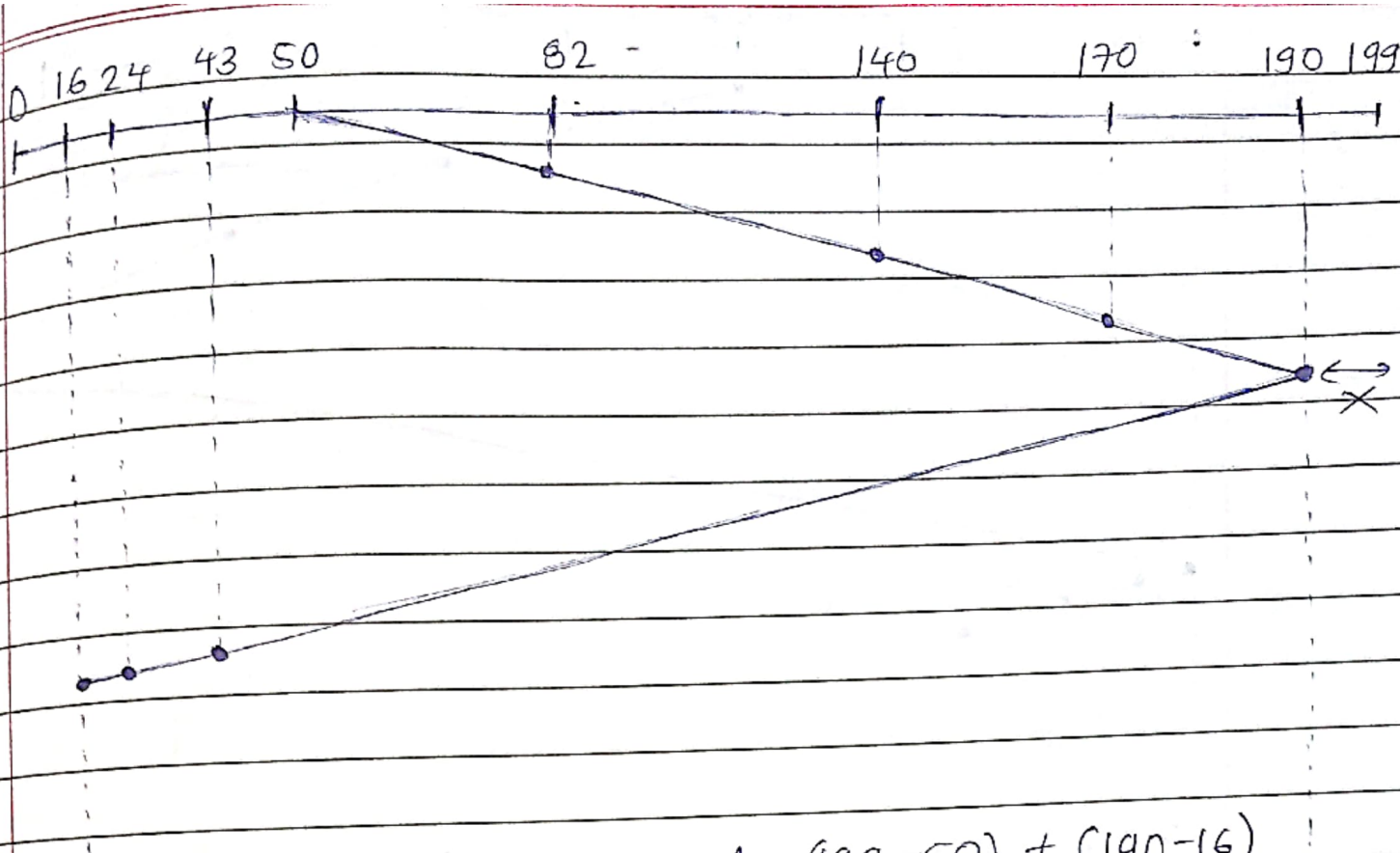
Total time = 332 * 1ns = 332 ns

LOOK

Same question

82, 170, 43, 140, 24, 16, 190
Curr. pos. of R/W head = 50

Similar to SCAN, disk arm instead of going to the end of disk just goes upto the last request to be serviced & then reverses its direction



Total overhead movement = (199-50) + (190-16)
= 314

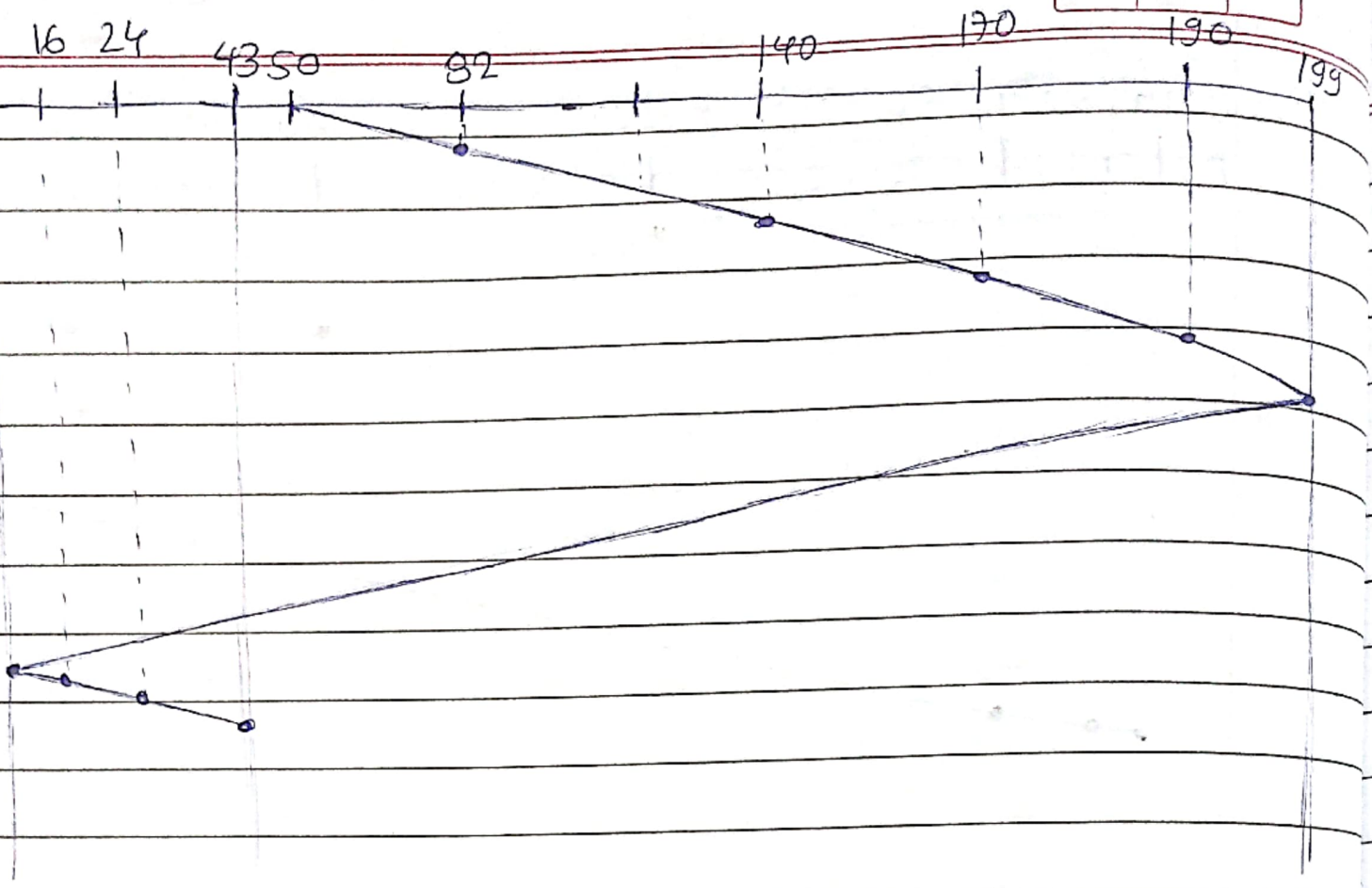
Time taken = 314 × 1ns = 314ns

Advantages:
→ prevents extra delay which occurred due to unnecessary traversal to the end of disk
→ Rest adv. & disad. same as SCAN

C-SCAN

Same question:
82, 170, 43, 140, 24, 16, 190

In SCAN, disk arm again ~~ser~~ scans the path that has been scanned, after reversing direction To avoid this, C-SCAN algo. ~~it~~ instead of reversing the ~~its~~ direction goes to the other end of the disk & starts ~~o~~ servicing req. from there.



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

Advantages:

→ Waiting time is uniformly distributed among requests
→ response time is good on it

Disadvantages

→ Time taken by disk arm to locate a spot is increased here
→ The head keeps going to the end of the disk

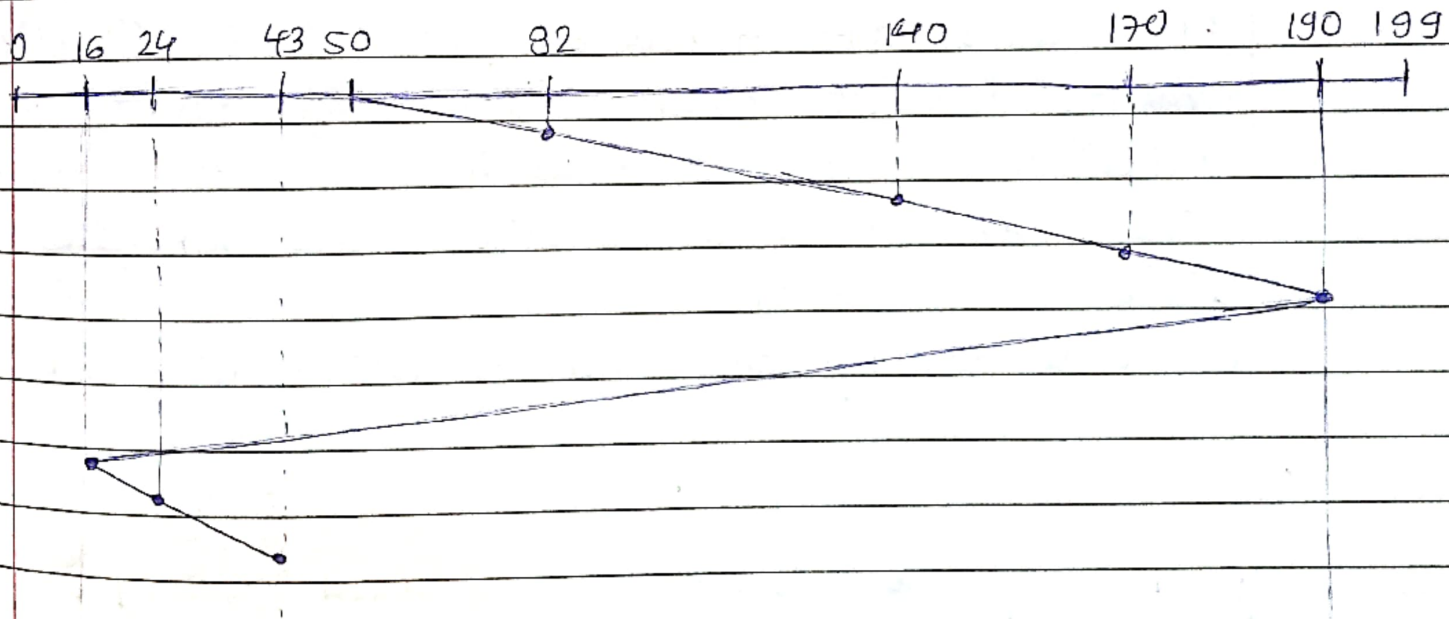
C-LOOK

Same question

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

Curr. pos. = 50

Similar to C-SCAN algo.,
Disk arm instead of going to the end of the disk, goes to the last request on one end & then goes to the last request at other end



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

Advantages
→ Decreased waiting time
→ If no req. till end, it reverses the head dir. immediately
→ No starvation

→ Time taken by disk arm to find desired spot is less

Disadvantage

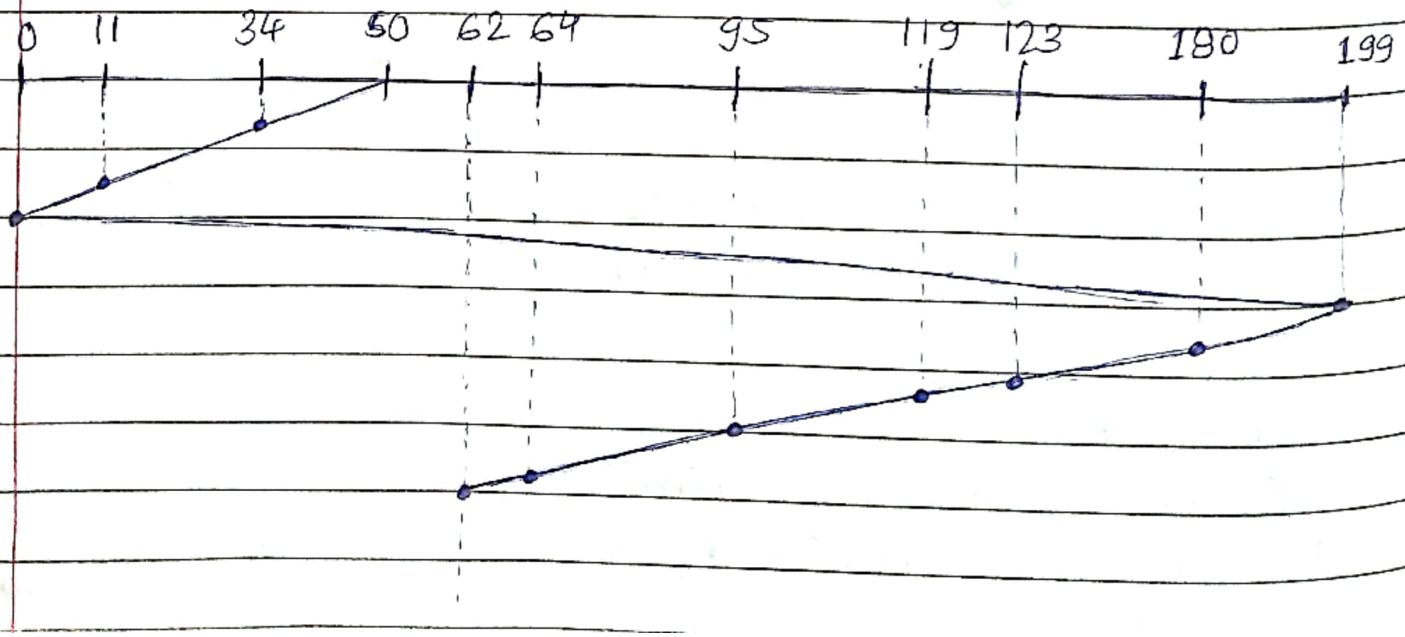
→ The arm has to be conscious about finding the last request

Q Consider the foll. track requests on disk queue:

95, 180, 34, 119, 11, 123, 62, 64

C-Scan algo. is used. R/W head is at loc. 50. if tracks are numbered from 0-199. Head is moving towards smaller track no. on its servicing total seek time needed with 2msec time to move from one track to another while servicing these requests is?
(Assume moving one end to other will take 10msec)

(1) 380 (2) 378 (3) 384 (4) 374



Total time = ~~(50-0)^{x2} + (199-62)^{y} + 10~~

$= ((50-0) + (199-62)) \times 2 + 10$

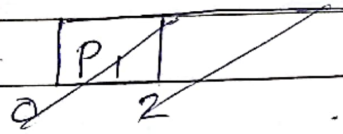
Time for moving from ~~one~~ one track to other

Time for moving from end of ~~disk~~ to other end

$= \boxed{384}$

Q Consider 3 CPU intensive processes which require 10, 20, 30 time units & arrives at 0, 2, 6 resp. How many context switches if OS use SRTF algo.? Do not count at time 0 & at the end.

A) 1 B) 2 C) 3 D) 4



	AT	BT
P_1	0	~~10~~ 0
P_2	2	~~20~~ 6
P_3	6	30