K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Batch: C2	**Roll No.:** 16010122323
Experiment No. 09	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

TITLE: Disk Scheduling Algorithms

AIM: Implementation of Disk Scheduling Algorithm like FCFS, SSTF, SCAN, CSCAN, LOOK.

Expected Outcome of Experiment:

CO 4. To understand various Memory, I/O and File management techniques.

Books/ Journals/ Websites referred:

1. **Silberschatz A., Galvin P., Gagne G. "Operating Systems Principles", Willey Eight edition.**
2. **Achyut S. Godbole , Atul Kahate "Operating Systems" McGraw Hill Third Edition.**
3. **William Stallings, "Operating System Internal & Design Principles", Pearson.**
4. **Andrew S. Tanenbaum, "Modern Operating System", Prentice Hall.**

Pre Lab/ Prior Concepts:

- Knowledge of disk scheduling algorithm.
- Calculation of seek time and transfer time etc.

Description of the application to be implemented:
1. First Come-First Serve (FCFS):

Working: Disk requests are addressed in the order they arrive, without any reordering or optimization based on their position on the disk.

Advantages:

1. Simple to implement.
2. Fair, as each request is processed in the order it arrives.

Disadvantages:

1. Can lead to inefficient disk movement if requests are spread out (long seek times).
2. Can suffer from the "convoy effect," where a single large request delays many smaller ones.

Example:

If the requests are: 98, 183, 37, 122, 14, 124, 65, 67 (and the current head is at 53), the requests are served in this order: 98 → 183 → 37 → 122 → 14 → 124 → 65 → 67.

2. Shortest Seek Time First (SSTF):

Working: The request closest to the current head position (in terms of seek time) is served next. This minimizes the distance the disk arm travels, optimizing seek time.

Advantages:

1. More efficient than FCFS in terms of average seek time.

Disadvantages:

1. Can lead to starvation. Requests that are far away from the current head position may never get served if there are always closer requests (e.g., if requests keep arriving for nearby positions).

Example:

With the same request queue: 98, 183, 37, 122, 14, 124, 65, 67 (and head at 53), SSTF would first serve 37 (closest to 53), then 65, 67, 98, 122, 124, 183, 14.

3. Elevator (SCAN):

Working: The disk arm moves in one direction (either up or down the disk) and serves requests along the way. Once it reaches the end of the disk (or last request in that direction), it reverses direction and continues serving requests in the opposite direction.

Advantages:

1. Avoids starvation.
2. Reduces the seek time for requests in one direction, providing more fairness compared to SSTF.

Disadvantages:

1. The disk head travels all the way to the end of the disk even if there are no requests near the end, which may cause unnecessary delay.

Example:

With the same request queue (head at 53, moving towards 183), the order would be: 65 → 67 → 98 → 122 → 124 → 183. Once the disk head reaches 183, it reverses and serves: 37 → 14.

4. Circular SCAN (C-SCAN):

Working: The disk arm moves in one direction (like SCAN) and serves requests along the way. When it reaches the end of the disk, instead of reversing, it jumps back to the beginning and continues serving requests.

Advantages:

1. Provides more uniform wait times compared to SCAN, as the disk head consistently moves in one direction.

Ensures no requests are skipped or delayed for long.

Disadvantages:

1. The head has to move from the end to the beginning of the disk without serving requests, which might cause some delay for lower-priority requests.

Example:

With the same request queue (head at 53), C-SCAN serves: 65 → 67 → 98 → 122 → 124 → 183, then jumps back to 0, and serves: 14 → 37.

5. LOOK:

Working: Similar to SCAN, but the disk arm does not move all the way to the end of the disk. Instead, it stops at the last request in the current direction before reversing. This reduces unnecessary movement when there are no requests near the end of the disk.

Advantages:

1. More efficient than SCAN as it avoids unnecessary travel to the end of the disk.
2. Prevents long waiting times for requests in both directions.

Disadvantages:

1. Still may have some higher wait times for requests far from the current head direction.

Example:

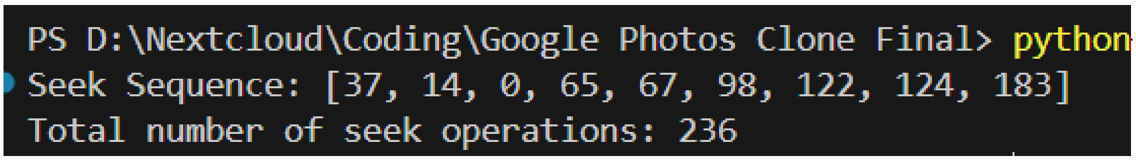
With the same request queue (head at 53), LOOK would serve: 65 → 67 → 98 → 122 → 124 → 183, then reverse to serve: 37 → 14.

Implementation details: (printout of code)

```
def SCAN(disk_size, requests, head, direction):
    seek_sequence = []
    seek_count = 0
    distance = 0
    cur_track = 0

    left = []
    right = []

    if direction == "left":
        left.append(0)
    elif direction == "right":
        right.append(disk_size - 1)

    for i in range(len(requests)):
        if requests[i] < head:
            left.append(requests[i])
        if requests[i] > head:
            right.append(requests[i])

    left.sort()
    right.sort()

    if direction == "left":
        for i in range(len(left) - 1, -1, -1):
            cur_track = left[i]
            seek_sequence.append(cur_track)
            distance = abs(cur_track - head)
            seek_count += distance
            head = cur_track

        for i in range(len(right)):
            cur_track = right[i]
            seek_sequence.append(cur_track)
            distance = abs(cur_track - head)
            seek_count += distance
            head = cur_track

    elif direction == "right":
        for i in range(len(right)):
            cur_track = right[i]
            seek_sequence.append(cur_track)
            distance = abs(cur_track - head)
            seek_count += distance
            head = cur_track

        for i in range(len(left) - 1, -1, -1):
            cur_track = left[i]
            seek_sequence.append(cur_track)
            distance = abs(cur_track - head)
            seek_count += distance
            head = cur_track

    return seek_sequence, seek_count



disk_size = 200
requests = [98, 183, 37, 122, 14, 124, 65, 67]
```

```
head = 53
direction = "left"
seek_sequence, total_seek_count = SCAN(disk_size, requests, head,
direction)
print(f"Seek Sequence: {seek_sequence}")
print(f"Total number of seek operations: {total_seek_count}")
```

Conclusion:



```
PS D:\Nextcloud\Coding\Google Photos Clone Final> python
Seek Sequence: [37, 14, 0, 65, 67, 98, 122, 124, 183]
Total number of seek operations: 236
```

<u>Post Lab Descriptive Questions</u>

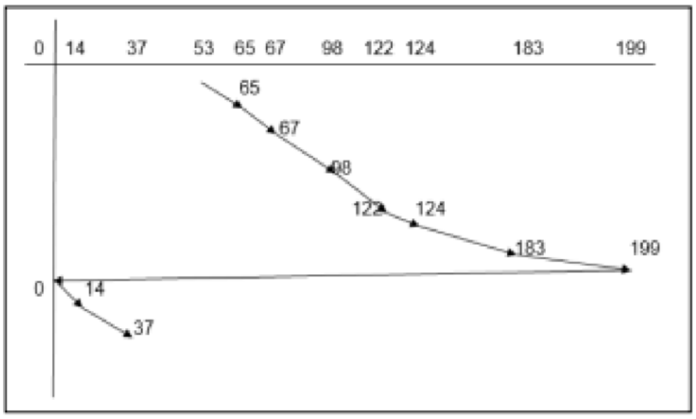
1. A disk drive has 200 cylinders numbered from 0 to 199. The disk head is initially at cylinder 53. The queue of pending requests in FIFO order is :
98, 183, 37, 122, 14, 124, 65, 67.

Starting from the current head position, what is the total distance travelled (in cylinders) by disk arm to satisfy the requests using CSCAN and Look. Illustrate with figures in each case.

Ans) CSCAN:
Circular scanning works just like the elevator to some extent. It beginsits scan toward the nearest end and works itsway all the way to the end of thesystem. Once it hits the bottom or top it jumps to the other end and moves in thesame direction. Keep in mind that the huge jump doesn't count as a headmovement.The heaviest density of request is at the other end of the disk. The request haswaited longer.This scheduling algorithm provides more uniform wait time C-SCAN moves the head from one end of the disk to the other, servicing request along the way.When the head reaches the other end, it immediately returns to the beginning ofthe disk without servicing any request on the return trip.

Consider an e.g.,Queue=98, 183, 37, 122, 14, 124, 65, 67.

Head starts at 53.

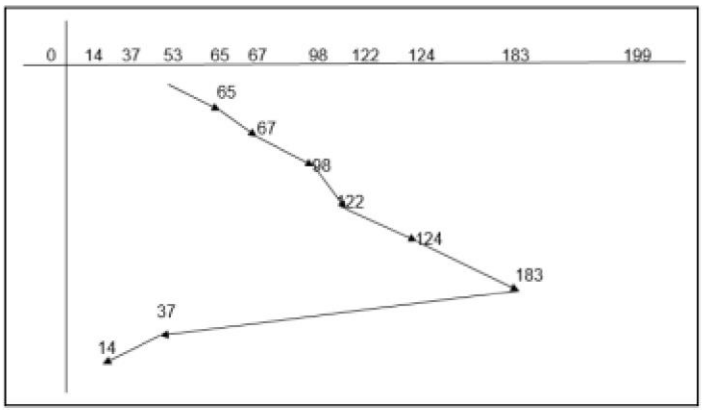


The totaldistance

(53-65)+(67-65)+(98-67)+(122-98)+(124-122)+(183-124)+(199-183)+(140)+(37-14)= 159 Cylinders

In another way:
C-Scan: 53 , 65 , 67 , 98 , 122 , 124 , 183 , 199 , 0 , 14 , 37
Look:
Let us see the same example for look scheduling queue:
98, 183, 37, 122, 14, 124, 65, 67.
Head starts: 53

Look and C-Look are the versions of SCAN and C-SCAN because they look for a request before continuing to move in a given direction.



The total distance

(65-53)+(67-65)+(98-67)+(122-98)+(124-122)+(183-124)+(183-37)+(37-14)= 299 Cylinders

In another way

LOOK: 53 , 65 , 67 , 98 , 122 , 124 , 183 , 37 , 14

1. In a hard disk, what rotates about a central spindle ______________
a. Disk
b. Platter
c. Sector
d. None of the above

Ans:

2. The time required to move the disk arm to the required track is known as ________
a. Latency time
b. Access time
c. Seek time
d. None of the above

Ans:

Date: _____________ **Signature of faculty in-charge**