OPERATING SYSTEM
WEEK 15 ASSIGNMENT

1. Compare the functionalities of FCFS, SSTF, C-SCAN and C-LOOK with example.

FCFS :
First Come First Serve (FCFS) is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.

- It supports non-preemptive and pre-emptive scheduling algorithm.
- Jobs are always executed on a first-come, first-serve basis.
- It is easy to implement and use.
- This method is poor in performance, and the general wait time is quite high.

Example of FCFS scheduling

A real-life example of the FCFS method is buying a movie ticket on the ticket counter. In this scheduling algorithm, a person is served according to the queue manner. The person who arrives first in the queue first buys the ticket and then the next one. This will continue until the last person in the queue purchases the ticket. Using this algorithm, the CPU process works in a similar manner.

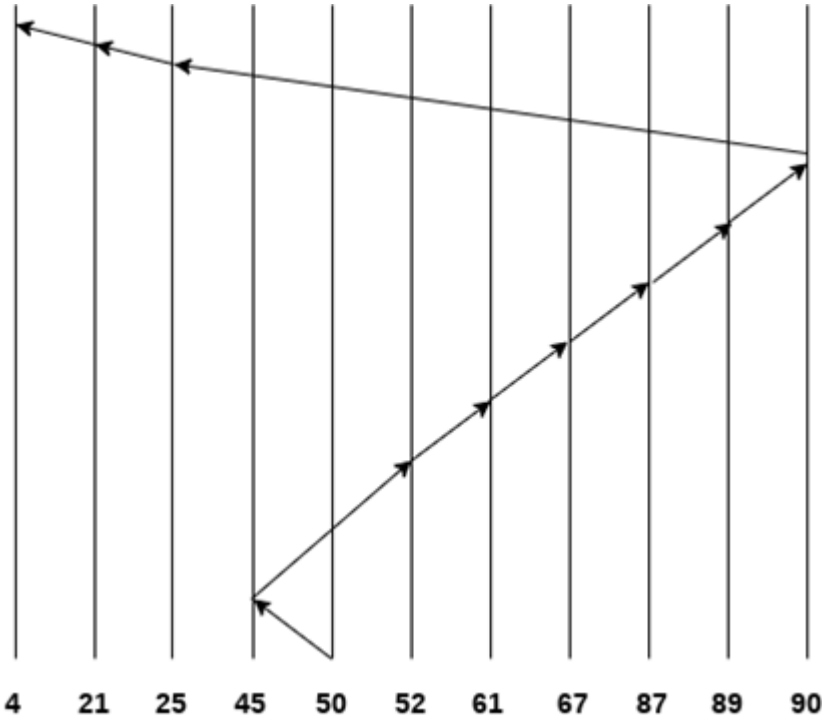
SSTF :

Shortest seek time first (SSTF) algorithm selects the disk I/O request which requires the least disk arm movement from its current position regardless of the direction. It reduces the total seek time as compared to FCFS.

It allows the head to move to the closest track in the service queue.

Consider the following disk request sequence for a disk with 100 tracks

45, 21, 67, 90, 4, 89, 52, 61, 87, 25

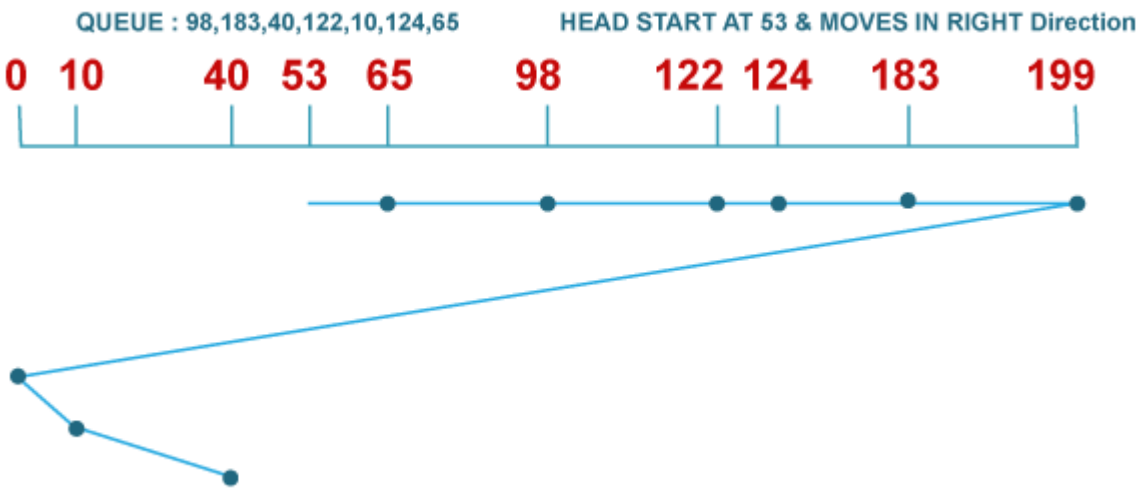


Number of cylinders = 5 + 7 + 9 + 6 + 20 + 2 + 1 + 65 + 4 + 17 = 136

C-SCAN :

C-SCAN algorithm, also known as the Circular Elevator algorithm, is the modified version of the SCAN algorithm. In this algorithm, the head pointer starts from one end of the disk and moves towards the other end, serving all requests in between. After reaching the other end, the head reverses its direction and goes to the starting point, and it then satisfies the remaining requests in the same direction as before. Unlike C-LOOK, the head pointer will move till the end of the disk, whether there is a request or not.

For example: Consider a disk with 200 tracks (0-199) and the disk queue having I/O requests in the following order as follows:The current head position of the Read/Write head is 53 and will move in the right direction. Calculate the total number of track movements of the Read/Write head using the C-SCAN algorithm.



Total head movements

= (65 - 53) + (98 - 65) + (122 - 98) + (124 - 122) + (183 - 124) + (199 - 183) + (199 - 0)

+ (10 - 0) + (40 - 10)

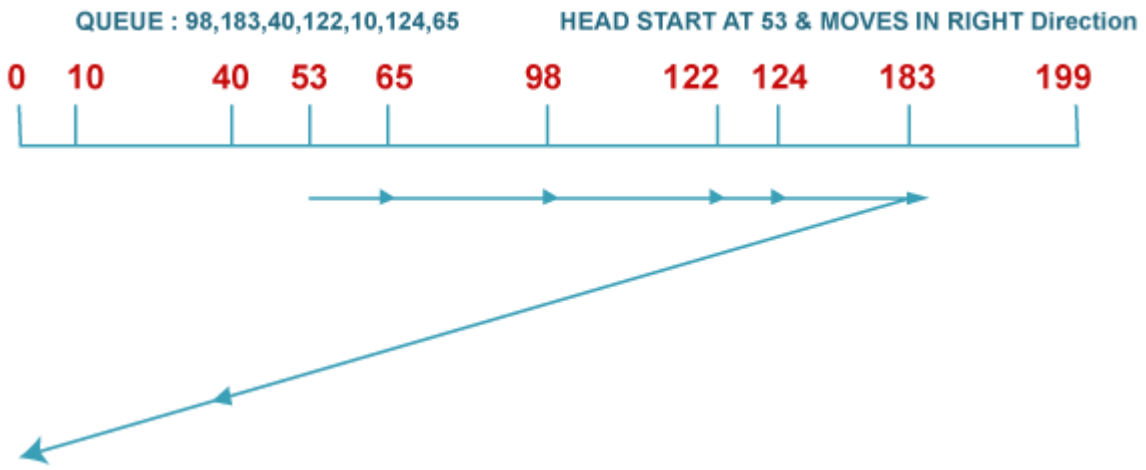
= 395

C- LOOK :

C-LOOK is the modified version of both LOOK and C-SCAN algorithms. In this algorithm, the head starts from the first request in one direction and moves towards the last request at another end, serving all requests in between. After reaching the last request in one end, the head jumps in another direction, moves towards the remaining requests, and then satisfies them in the same direction as before. Unlike C-SCAN, the head pointer will move till the end request of the disk.

For example: Consider a disk with 200 tracks (0-199) and the disk queue having I/O requests in the following order as follows:

The current head position of the Read/Write head is 53 and will move in the right direction. Calculate the total number of track movements of the Read/Write head using the C-LOOK algorithm.



1. Total head movements
2. = (65 - 53) + (98 - 65) + (122 - 98) + (124 - 122) + (183 - 124) + (183 - 10) + (40 - 10)
3. = 333

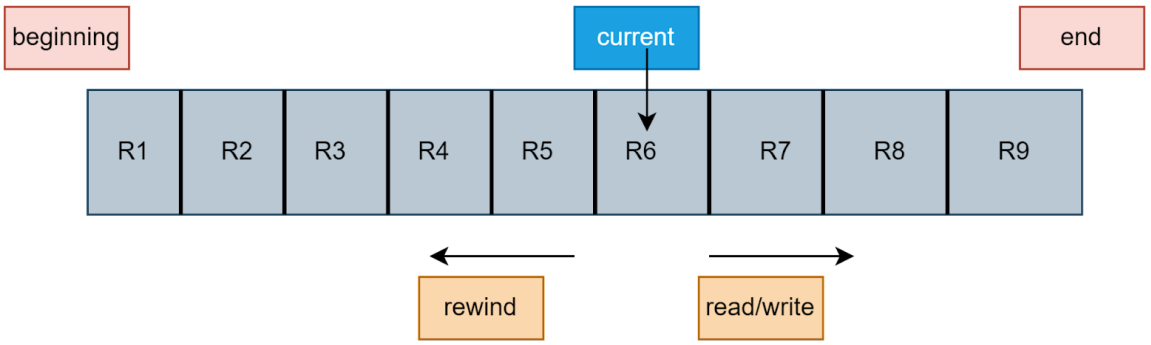
2. What are files and explain the access methods for files?

Files contain a lot of information required by the system. The computer memory may require certain files during execution. We need very efficient methods to retrieve the information by accessing the files in the least time possible.
In this article, we will learn about the three types of file access methods. They are:

1. Sequential access
2. Direct access
3. Indexed sequential access

Sequential access method

It is one of the simplest access methods and is widely used in editors and compilers. You must have seen the audio cassettes. Even they use the same access method.
The files are a collection of records. Accessing the file is equivalent to accessing the records. In the sequential access method, each record is accessed sequentially, one after the other. Consider the below image for more clarity.



The figure represents a file. The current pointer is pointing to the record currently being accessed. In the sequential access method, the current pointer cannot directly jump to any record. It has to "cross" every record that comes in its path. Suppose there are nine records in the file from R1 to R9. The current pointer is at record R6. If we want to access record R8, we have to first access record R6 and record R7. This is one of the major disadvantages of the sequential access method.
The sequential access method has three operations:

- Read next: It will read the next record in the file. The file pointer (current pointer) will advance to the next record. It is similar to how we traverse the nodes in a linked list.
- Write next: This operation is used when some more information is to be included in the file. A new node (a record) will be added at the end of the file. The end pointer will now point to the node which has been added, marking it as the new end of the file.It is similar to adding a new node at the end of a linked list.
- Rewind: This will bring the read and write pointers to the beginning of the file.

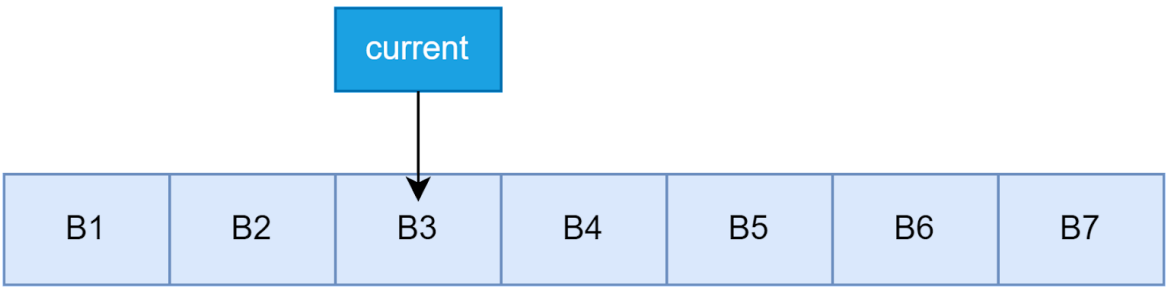
Analysis of sequential access method

1. It is very simple to implement. The work is similar to a linked list.
2. Since the records cannot be randomly accessed, it is not a very efficient method.
3. It is a slow method.

Direct access method

In the direct access method, the files are considered as a sequence of blocks or records, just like the disk was considered to be divided into equal-sized blocks. The benefit of this method is that we can access any block randomly. The direct access method is known as the relative access method. The exact block address is known only to the operating system. When a user wants to access a particular block, he/she provides the relative block number, which the operating system then uses to find the exact block address.

This type of access method is used in database management systems.



The direct access method has the following operations:

- Read n: This operation is used to read the nth block. Read 6 would allow us to read block B6.
- Write n: This operation is used to write in the nth block.
- Goto n: This operation is used to directly access the nth block.

Analysis of direct access method

1. Faster than the sequential access method.
2. Allows random access. Therefore there is no need to traverse all the blocks.
3. Implementation is easy.

The major issue with the sequential access method was that it did not allow random access to the file records/blocks. The index sequential access method solves this problem. In this method, there is an index that holds the pointers to various blocks of the file. In order to access any block of the file, one has to first access the index, and from there, we can get the pointers to various blocks.
This method is very similar to the indexed file allocation method, where we had an index block that held pointers to various other disk blocks that were allocated to the file.

Analysis of sequential access method

1. It is a modification of the sequential access method. It allows random access.
2. Apart from the file records, an extra index is required to keep track of the blocks.
3. In case file size increases, the index may not be able to hold all the pointers due to memory constraints. Hence in such a case multi-level index may be used.