Experiment No.10
File Management & I/O Management Implement disk scheduling algorithms FCFS, SSTF.
Date of Performance:
Date of Submission:
Marks:
Sign:

Aim: To study and implement disk scheduling algorithms FCFS.

Objective:

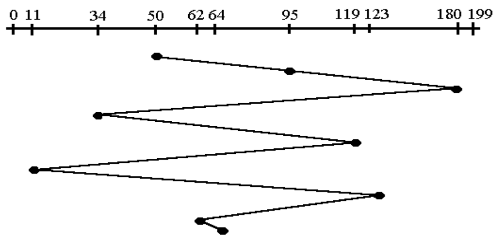
The main purpose of disk scheduling algorithm is to select a disk request from the queue of IO requests and decide the schedule when this request will be processed.

Theory:

TYPES OF DISK SCHEDULING ALGORITHMS

Although there are other algorithms that reduce the seek time of all requests, I will only concentrate on the following disk scheduling algorithms:
1.First Come-First Serve (FCFS)
2.Shortest Seek Time First (SSTF)
3.Elevator (SCAN)
4.Circular SCAN (C-SCAN)
5.C-LOOK

Given the following queue -- 95, 180, 34, 119, 11, 123, 62, 64 with the Read-write head initially at the track 50 and the tail track being at 199 let us now discuss the different algorithms.



FCFS

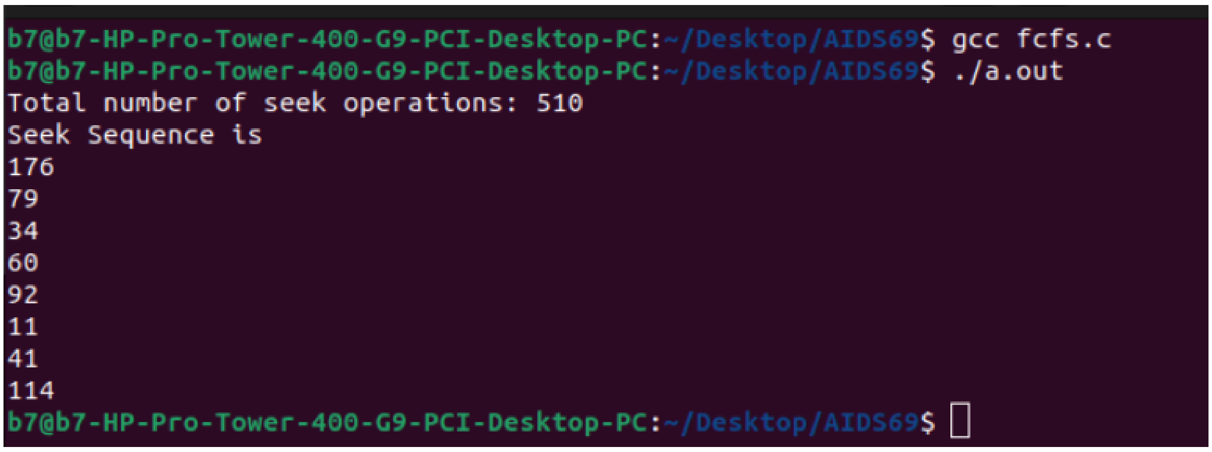
First Come -First Serve (FCFS)

All incoming requests are placed at the end of the queue. Whatever number that is next in the queue will be the next number served. Using this algorithm doesn't provide the best results. To determine the number of head movements you would simply find the number of tracks it took to move from one request to the next. For this case it went from 50 to 95 to 180 and so on. From 50 to 95 it moved 45 tracks. In this example, it had a total head movement of 640 tracks. The disadvantage of this algorithm is noted by the oscillation from track 50 to track 180 and then back to track 11 to 123 then to 64. As you will soon see, this is the worse algorithm that one can use.

Program:

```
#include <stdio.h>
#include <math.h>

int size = 8;

void FCFS(int arr[],int head)
{
	int seek_count = 0;
	int cur_track, distance;

	for(int i=0;i<size;i++)
	{
		cur_track = arr[i];

		// calculate absolute distance
		distance = fabs(head - cur_track);

		// increase the total count
		seek_count += distance;

		// accessed track is now new head
		head = cur_track;
	}

	printf("Total number of seek operations: %d\n",seek_count);

	// Seek sequence would be the same
	// as request array sequence
	printf("Seek Sequence is\n");

	for (int i = 0; i < size; i++) {
		printf("%d\n",arr[i]);
	}
}

//Driver code
int main()
{
        // request array
        int arr[8] = { 176, 79, 34, 60, 92, 11, 41, 114 };
        int head = 50;

        FCFS(arr,head);

        return 0;
}
```

Output:



```
b7@b7-HP-Pro-Tower-400-G9-PCI-Desktop-PC:~/Desktop/AIDS69$ gcc fcfs.c
b7@b7-HP-Pro-Tower-400-G9-PCI-Desktop-PC:~/Desktop/AIDS69$ ./a.out
Total number of seek operations: 510
Seek Sequence is
176
79
34
60
92
11
41
114
b7@b7-HP-Pro-Tower-400-G9-PCI-Desktop-PC:~/Desktop/AIDS69$
```

Conclusion:

Why is Disk Scheduling important?

Disk scheduling is a critical component of modern operating systems, particularly in systems where multiple processes are contending for access to the disk. The primary purpose of disk scheduling algorithms is to optimize the order in which disk requests from different processes are serviced, with the goal of reducing disk seek time and maximizing disk throughput.

Disk seek time, the time it takes for the disk's read/write head to move to the appropriate track on the disk, is a significant contributor to overall disk access latency. By arranging disk requests in an efficient order, disk scheduling algorithms aim to minimize seek time and improve system responsiveness and performance.