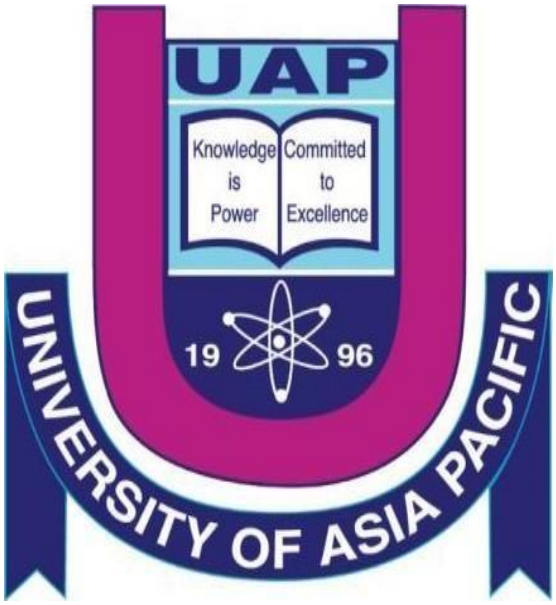

Department of Computer Science & Engineering

Course Title: Operating System Lab

Course Code: CSE 406

Lab Report No: 09

Lab Report: C Scan Disk Scheduling

Submitted To:

Atia Rahman Orthi

Lecturer,

Department of CSE, UAP

Submitted By:

Name: Amirul Islam Papon

Reg No: 21201076

Sec: B

Problem Statement:

To implement the C-SCAN (Circular SCAN) disk scheduling algorithm that services requests in one direction only and jumps to the start when the end is reached.

Steps:

- Accept the request queue and initial head position.
- Sort requests and divide into two parts: higher than and lower than head.
- Service all requests in one direction (toward the higher numbers).
- Once the end of the disk is reached, jump to the beginning without servicing.
- Continue servicing from the lowest request upwards.
- Compute the total seek time and servicing order.

Code:

C_scan_disk_scheduling.py > c_scan

```python
def c_scan(requests, head, disk_size=200):
    requests = sorted(requests)
    sequence = []
    total_seek = 0
    current = head

    left = [r for r in requests if r < head]
    right = [r for r in requests if r >= head]

    for r in right:
        sequence.append(r)
        total_seek += abs(current - r)
        current = r

    if current != disk_size - 1:
        total_seek += abs(current - (disk_size - 1))
        current = 0
        total_seek += disk_size - 1

    for r in left:
        sequence.append(r)
        total_seek += abs(current - r)
        current = r

    print("\nC-SCAN Disk Scheduling:")
    print("Sequence:", sequence)
    print("Total Seek Time:", total_seek)

requests = [0, 14, 41, 53, 65, 67, 98, 122, 124, 183, 199]
head = 53

c_scan(requests, head)

```

Output:

```
C-SCAN Disk Scheduling:
Sequence: [53, 65, 67, 98, 122, 124, 183, 199, 0, 14, 41]
Total Seek Time: 386

[Done] exited with code=0 in 0.06 seconds
```

Discussion:

C-SCAN provides a more uniform wait time than SCAN because it always services in one direction, resetting to the beginning after reaching the end.

Conclusion:

C-SCAN offers better predictability and equal waiting times for all requests. It is ideal for systems with constant loads and real-time requirements.

Git Link: https://github.com/Amirul-Islam-Papon/Operating-System/blob/main/C_scan_disk_scheduling.py