praveenkumar.R
192324012

Experiment-39:Develop a C program to simulate C-SCAN disk scheduling algorithm.

Aim:
To simulate the C-SCAN (Circular SCAN) disk scheduling algorithm in C.

Procedure:
1. Take the number of disk requests, the initial position of the disk head, and the direction of movement (left or right) as input.
2. Sort the disk requests in increasing order.
3. Process the disk requests in the direction of the disk head movement until the end, then jump to the opposite end of the disk and continue processing the requests in the same direction.
4. Calculate and display the seek sequence and total seek time.

C Program:
```c
#include <stdio.h>
#include <stdlib.h>

int compare(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}

int main() {
    int n, initial_position, total_seek_time = 0, direction;
    printf("Enter the number of disk requests: ");
    scanf("%d", &n);

    int requests[n];
    printf("Enter the disk requests: \n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &requests[i]);
    }

    printf("Enter the initial disk head position: ");
    scanf("%d", &initial_position);

    printf("Enter the direction (0 for left, 1 for right): ");
    scanf("%d", &direction);

    qsort(requests, n, sizeof(int), compare);

    int current_position = initial_position;
    int total_distance = 0;
    int i;

    if (direction == 1) {
        for (i = 0; i < n; i++) {
            if (requests[i] >= current_position) {
                break;
```

```c
        }
    }

    for (int j = i; j < n; j++) {
        total_seek_time += abs(current_position - requests[j]);
        current_position = requests[j];
    }

    total_seek_time += abs(current_position - requests[n - 1]);
    current_position = requests[n - 1];

    total_seek_time += abs(current_position - 0);
    current_position = 0;

    for (int j = 0; j < i; j++) {
        total_seek_time += abs(current_position - requests[j]);
        current_position = requests[j];
    }
} else {
    for (i = n - 1; i >= 0; i--) {
        if (requests[i] <= current_position) {
            break;
        }
    }

    for (int j = i; j >= 0; j--) {
        total_seek_time += abs(current_position - requests[j]);
        current_position = requests[j];
    }

    total_seek_time += abs(current_position - 0);
    current_position = 0;

    total_seek_time += abs(current_position - requests[n - 1]);
    current_position = requests[n - 1];

    for (int j = n - 2; j >= i; j--) {
        total_seek_time += abs(current_position - requests[j]);
        current_position = requests[j];
    }
}

printf("Total Seek Time: %d\n", total_seek_time);
return 0;
}
```

Output:

Output

```
Enter the number of disk requests: 4
Enter the disk requests:
44
56
25
12
Enter the initial disk head position: 12
Enter the direction (0 for left, 1 for right): 1
Total Seek Time: 100
```