praveenkumar.R
192324012

Experiment-38:Design a C program to simulate SCAN disk scheduling algorithm.

Aim:
To simulate the 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 to handle them in the direction of the disk head movement.
3. Simulate the SCAN algorithm by processing the requests in the current direction until the end of the disk, then reverse the direction.
4. Calculate and display the seek sequence and total seek time.

C Program:

c

Copy code

```
#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, direction, total_seek_time = 0;
   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;
        }
    }
    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];

    for (int j = i - 1; j >= 0; 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 - requests[0]);
    current_position = requests[0];

    for (int j = i + 1; j < n; 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: 6
Enter the disk requests:
44
5
6
8
9
7
Enter the initial disk head position: 8
Enter the direction (0 for left, 1 for right): 0
Total Seek Time: 42
```