192311035

M.RAVI SAI VINAY

EXPERIMENT-38

Design a C program to simulate SCAN disk scheduling algorithm.

AIM:-

The aim of this program is to simulate the **SCAN disk scheduling algorithm**. In SCAN, the disk arm moves towards one end of the disk, servicing all requests in its path, and then reverses direction at the end of the disk to service requests in the opposite direction.

ALGORITHM:-

1. Input: A list of disk requests, the initial position of the disk head, and the direction in which the disk arm is initially moving (left or right).
2. Sort the Requests: Sort the disk requests in ascending order for one direction and in descending order for the other direction.
3. Service the Requests: Move the disk arm towards one end, servicing requests as it goes, and when it reaches the end, reverse the direction and service the remaining requests.
4. Calculate the Total Seek Time: Add up all the individual seek times.
5. Display the Disk Head Movement: Display the movement of the disk arm from one request to the next and the total seek time**.**

PROCEDURE:-

1. Initialize the disk head position and direction.
2. Accept the disk requests and the total number of requests.
3. Sort the disk requests in ascending and descending order.
4. Simulate the movement of the disk arm to service the requests in the SCAN algorithm.
5. Display the movement of the disk arm and the total seek time.

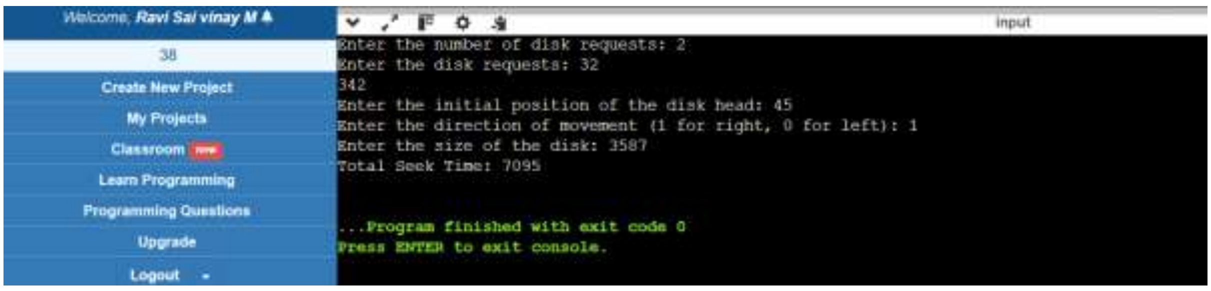
CODE:-

```
#include <stdio.h>

#include <stdlib.h>


// Function to sort an array

void sort(int arr[], int n) {

    for (int i = 0; i < n-1; i++) {

        for (int j = i+1; j < n; j++) {

            if (arr[i] > arr[j]) {

                int temp = arr[i];

                arr[i] = arr[j];

                arr[j] = temp;

            }

        }

    }

}


// Function to simulate SCAN Disk Scheduling algorithm

void SCAN(int requests[], int n, int initialPosition, int direction, int diskSize) {

    int totalSeekTime = 0;

    int currentPosition = initialPosition;

    int left[diskSize], right[diskSize];

    int leftCount = 0, rightCount = 0;
```

```
// Categorize requests into left and right based on the initial position
for (int i = 0; i < n; i++) {
    if (requests[i] < currentPosition) {
        left[leftCount++] = requests[i];
    } else {
        right[rightCount++] = requests[i];
    }
}

// Sort the left and right arrays
sort(left, leftCount);
sort(right, rightCount);

// Calculate seek time
int seekTime = 0;
if (direction == 1) {
    // Moving right first
    for (int i = 0; i < rightCount; i++) {
        seekTime += abs(currentPosition - right[i]);
        currentPosition = right[i];
    }
    // Move to the farthest right
    seekTime += abs(currentPosition - (diskSize - 1));
    currentPosition = diskSize - 1;
    // Move left now
    for (int i = leftCount - 1; i >= 0; i--) {
        seekTime += abs(currentPosition - left[i]);
        currentPosition = left[i];
    }
} else {
    // Moving left first
    for (int i = leftCount - 1; i >= 0; i--) {
        seekTime += abs(currentPosition - left[i]);
        currentPosition = left[i];
    }
    // Move to the farthest left
    seekTime += abs(currentPosition - 0);
    currentPosition = 0;

    // Move right now
    for (int i = 0; i < rightCount; i++) {
        seekTime += abs(currentPosition - right[i]);
        currentPosition = right[i];
    }
}

// Display disk head movements and total seek time
```

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


int main() {
    int n, initialPosition, direction, diskSize;


    // Accept the number of disk requests and disk size
    printf("Enter the number of disk requests: ");
    scanf("%d", &n);
    int requests[n];
    printf("Enter the disk requests: ");
    for (int i = 0; i < n; i++) {
        scanf("%d", &requests[i]);
    }


    // Accept the initial position of the disk head and direction (1 for right, 0 for left)
    printf("Enter the initial position of the disk head: ");
    scanf("%d", &initialPosition);
    printf("Enter the direction of movement (1 for right, 0 for left): ");
    scanf("%d", &direction);


    // Accept the disk size
    printf("Enter the size of the disk: ");
    scanf("%d", &diskSize);
    // Simulate SCAN disk scheduling

    SCAN(requests, n, initialPosition, direction, diskSize);



    return 0;

}
```

OUTPUT:-



```
Enter the number of disk requests: 2
Enter the disk requests: 32
342
Enter the initial position of the disk head: 45
Enter the direction of movement (1 for right, 0 for left): 1
Enter the size of the disk: 3587
Total Seek Time: 7095


...Program finished with exit code 0
Press ENTER to exit console.
```

RESULT:-

T The program successfully simulates the **SCAN disk scheduling algorithm**. It processes disk requests by moving the disk arm first in one direction (either left or right) and then reversing at the end of the disk.