Assignment 8 : Disk Scheduling

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

#define MAX_REQUESTS 100

void calculateSSTF(int requests[], int n, int head) {
    bool visited[MAX_REQUESTS] = { false };
    int total_head_movement = 0;
    int current_position = head;

    for (int i = 0; i < n; i++) {
        int closest_index = -1;
        int min_distance = 99999;

        for (int j = 0; j < n; j++) {
            if (!visited[j]) {
                int distance = abs(requests[j] - current_position);
                if (distance < min_distance) {
                    min_distance = distance;
                    closest_index = j;
                }
            }
        }

        visited[closest_index] = true;
        total_head_movement += min_distance;
        current_position = requests[closest_index];
        printf("Servicing request at track %d\n", current_position);
    }

    printf("Total head movements (SSTF): %d\n", total_head_movement);
}

void calculateSCAN(int requests[], int n, int head, int max_track) {
    int total_head_movement = 0;
    int current_position = head;

    // Sort the requests
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (requests[j] > requests[j + 1]) {
                int temp = requests[j];
                requests[j] = requests[j + 1];
                requests[j + 1] = temp;
            }
        }
    }

    // Move in the direction of SCAN
    for (int i = 0; i < n; i++) {
        if (requests[i] >= current_position) {
```

```
            total_head_movement += abs(current_position - requests[i]);
            current_position = requests[i];
            printf("Servicing request at track %d\n", current_position);
        }
    }

    // Move to the end
    total_head_movement += abs(current_position - max_track);
    current_position = max_track;

    // Now go back
    for (int i = n - 1; i >= 0; i--) {
        if (requests[i] < head) {
            total_head_movement += abs(current_position - requests[i]);
            current_position = requests[i];
            printf("Servicing request at track %d\n", current_position);
        }
    }

    printf("Total head movements (SCAN): %d\n", total_head_movement);
}

void calculateCLOOK(int requests[], int n, int head) {
    int total_head_movement = 0;
    int current_position = head;

    // Sort the requests
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (requests[j] > requests[j + 1]) {
                int temp = requests[j];
                requests[j] = requests[j + 1];
                requests[j + 1] = temp;
            }
        }
    }

    // Move to the right and wrap around
    for (int i = 0; i < n; i++) {
        if (requests[i] >= current_position) {
            total_head_movement += abs(current_position - requests[i]);
            current_position = requests[i];
            printf("Servicing request at track %d\n", current_position);
        }
    }

    // Wrap around to the first request
    if (current_position < requests[0]) {
        total_head_movement += abs(current_position - requests[0]);
        current_position = requests[0];
    }

    for (int i = 0; i < n; i++) {
        if (requests[i] < current_position) {
            total_head_movement += abs(current_position - requests[i]);
```

```
            current_position = requests[i];
            printf("Servicing request at track %d\n", current_position);
        }
    }

    printf("Total head movements (C-LOOK): %d\n", total_head_movement);
}

int main() {
    int requests[MAX_REQUESTS], n, head, max_track;
    int choice;

    // Input for requests and head position
    printf("Enter the number of requests (max %d): ", MAX_REQUESTS);
    scanf("%d", &n);

    printf("Enter the requests:\n");
    for (int i = 0; i < n; i++) {
        printf("Request %d: ", i + 1);
        scanf("%d", &requests[i]);
    }

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

    printf("Enter the maximum track: ");
    scanf("%d", &max_track);

    do {
        printf("Disk Scheduling Algorithms\n");
        printf("1. SSTF\n");
        printf("2. SCAN\n");
        printf("3. C-LOOK\n");
        printf("4. Exit\n");
        printf("Choose an algorithm (1-4): ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                calculateSSTF(requests, n, head);
                break;
            case 2:
                calculateSCAN(requests, n, head, max_track);
                break;
            case 3:
                calculateCLOOK(requests, n, head);
                break;
            case 4:
                printf("Exiting program.\n");
                break;
            default:
                printf("Invalid choice! Please try again.\n");
        }
        printf("\n");
    } while (choice != 4);
```

```
    return 0;
}
```

OUTPUT:

```
meit@meit-OptiPlex-3046:~/33265$ gcc -o disk_scheduling ass8.c
meit@meit-OptiPlex-3046:~/33265$ ./disk_scheduling
Enter the number of requests (max 100): 8
Enter the requests:
Request 1: 98
Request 2: 183
Request 3: 41
Request 4: 122
Request 5: 14
Request 6: 124
Request 7: 65
Request 8: 67
Enter the initial head position: 53
Enter the maximum track: 199
Disk Scheduling Algorithms
1. SSTF
2. SCAN
3. C-LOOK
4. Exit
Choose an algorithm (1-4): 1
Servicing request at track 41
Servicing request at track 65
Servicing request at track 67
Servicing request at track 98
Servicing request at track 122
Servicing request at track 124
Servicing request at track 183
Servicing request at track 14
Total head movements (SSTF): 323

Disk Scheduling Algorithms
1. SSTF
2. SCAN
3. C-LOOK
4. Exit
Choose an algorithm (1-4): 2
Servicing request at track 65
Servicing request at track 67
Servicing request at track 98
Servicing request at track 122
Servicing request at track 124
Servicing request at track 183
Servicing request at track 41
Servicing request at track 14
Total head movements (SCAN): 331

Disk Scheduling Algorithms
1. SSTF
```

2. SCAN
3. C-LOOK
4. Exit
Choose an algorithm (1-4): 3
Servicing request at track 65
Servicing request at track 67
Servicing request at track 98
Servicing request at track 122
Servicing request at track 124
Servicing request at track 183
Servicing request at track 14
Total head movements (C-LOOK): 299

Disk Scheduling Algorithms
1. SSTF
2. SCAN
3. C-LOOK
4. Exit
Choose an algorithm (1-4): 4
Exiting program.