

```

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

#define MAX_REQUESTS 10

void fcfs(int requests[], int num_requests, int start) {
    int seek_count = 0;
    int distance;
    int cur_track = start;

    printf("FCFS Disk Scheduling:\n");
    for (int i = 0; i < num_requests; i++) {
        distance = abs(requests[i] - cur_track);
        seek_count += distance;
        cur_track = requests[i];
        printf("Move to track %d\n", requests[i]);
    }
    printf("Total seek time: %d\n", seek_count);
}

void scan(int requests[], int num_requests, int start, int direction) {
    int seek_count = 0;
    int cur_track = start;
    int distance;

    printf("SCAN Disk Scheduling:\n");
    int direction_change = (direction == 1) ? 0 : 1;
    int max = 0, min = 0;
    for (int i = 0; i < num_requests; i++) {
        if (requests[i] > max)
            max = requests[i];
        if (requests[i] < min)
            min = requests[i];
    }
    if (direction == 1) {
        for (int i = 0; i < num_requests; i++) {
            if (requests[i] >= cur_track) {
                distance = abs(requests[i] - cur_track);
                seek_count += distance;
                cur_track = requests[i];
                printf("Move to track %d\n", requests[i]);
            }
        }
        if (cur_track != max) {
            seek_count += abs(max - cur_track);
            cur_track = max;
            printf("Move to track %d\n", max);
        }
        for (int i = 0; i < num_requests; i++) {
            if (requests[i] < cur_track) {
                distance = abs(requests[i] - cur_track);
                seek_count += distance;
                cur_track = requests[i];
                printf("Move to track %d\n", requests[i]);
            }
        }
    }
    else {
        for (int i = 0; i < num_requests; i++) {
            if (requests[i] <= cur_track) {
                distance = abs(requests[i] - cur_track);
                seek_count += distance;
                cur_track = requests[i];
                printf("Move to track %d\n", requests[i]);
            }
        }
    }
}

```

```

    }
}
if (cur_track != min) {
    seek_count += abs(min - cur_track);
    cur_track = min;
    printf("Move to track %d\n", min);
}
for (int i = 0; i < num_requests; i++) {
    if (requests[i] > cur_track) {
        distance = abs(requests[i] - cur_track);
        seek_count += distance;
        cur_track = requests[i];
        printf("Move to track %d\n", requests[i]);
    }
}
}
printf("Total seek time: %d\n", seek_count);
}

void look(int requests[], int num_requests, int start, int direction) {
    int seek_count = 0;
    int cur_track = start;
    int distance;
    int max = 0, min = 0;

    printf("LOOK Disk Scheduling:\n");
    for (int i = 0; i < num_requests; i++) {
        if (requests[i] > max)
            max = requests[i];
        if (requests[i] < min)
            min = requests[i];
    }
    if (direction == 1) {
        for (int i = 0; i < num_requests; i++) {
            if (requests[i] >= cur_track) {
                distance = abs(requests[i] - cur_track);
                seek_count += distance;
                cur_track = requests[i];
                printf("Move to track %d\n", requests[i]);
            }
        }
        for (int i = 0; i < num_requests; i++) {
            if (requests[i] < cur_track) {
                distance = abs(requests[i] - cur_track);
                seek_count += distance;
                cur_track = requests[i];
                printf("Move to track %d\n", requests[i]);
            }
        }
    }
    else {
        for (int i = 0; i < num_requests; i++) {
            if (requests[i] <= cur_track) {
                distance = abs(requests[i] - cur_track);
                seek_count += distance;
                cur_track = requests[i];
                printf("Move to track %d\n", requests[i]);
            }
        }
        for (int i = 0; i < num_requests; i++) {
            if (requests[i] > cur_track) {
                distance = abs(requests[i] - cur_track);
                seek_count += distance;
                cur_track = requests[i];
                printf("Move to track %d\n", requests[i]);
            }
        }
    }
}

```

```

    }
}
printf("Total seek time: %d\n", seek_count);
}

int main() {
    int requests[] = {55, 58, 39, 18, 90, 160, 150};
    int num_requests = sizeof(requests) / sizeof(requests[0]);
    int start = 50;
    int direction = 1;

    printf("Disk Scheduling Algorithms Simulation\n");

    fcfs(requests, num_requests, start);

    printf("\n");
    scan(requests, num_requests, start, direction);

    printf("\n");
    look(requests, num_requests, start, direction);

    return 0;
}

```