LAB 13

QUESTION: Write a C program to simulate disk scheduling algorithms

a) FCFS b) SSTF c) SCAN

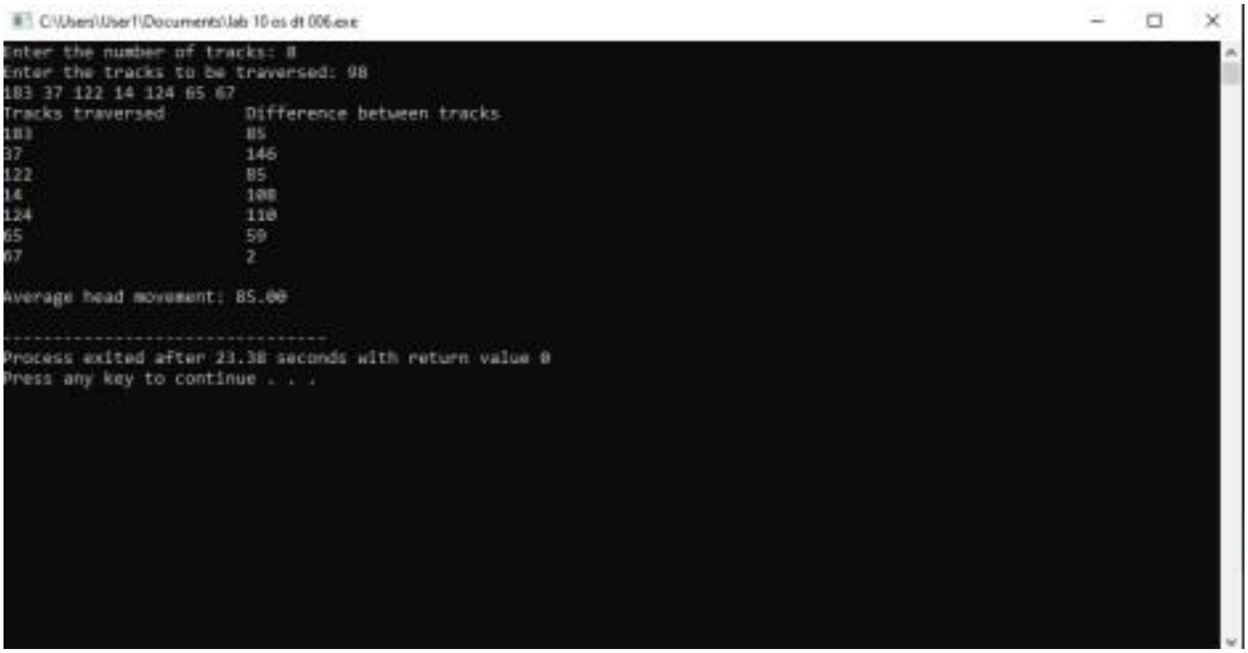
ANSWER:

CODE:

A) FCFS

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

int main() {
    int t[20], n, tohm[20], tot = 0;
    float avhm;

    printf("Enter the number of tracks: ");
    scanf("%d", &n);

    printf("Enter the tracks to be traversed: ");
    // Input the tracks, starting from t[0] for correct
    indexing for(int i = 0; i < n; i++) {
        scanf("%d", &t[i]);
    }

    // Calculate the differences between consecutive tracks
    for(int i = 1; i < n; i++) {
        tohm[i] = abs(t[i] - t[i-1]); // Absolute difference
    }

    // Calculate the total head movement
    for(int i = 1; i < n; i++) {
        tot += tohm[i];
    }

    // Calculate the average head movement
    avhm = (float)tot / (n-1); // The average is based on n-1 differences

    // Display the result
    printf("Tracks traversed\tDifference between
    tracks\n"); for(int i = 1; i < n; i++) {
        printf("%d\t\t\t%d\n", t[i], tohm[i]);
    }
    printf("\nAverage head movement: %.2f\n", avhm);

    return 0;
}
```

Output:



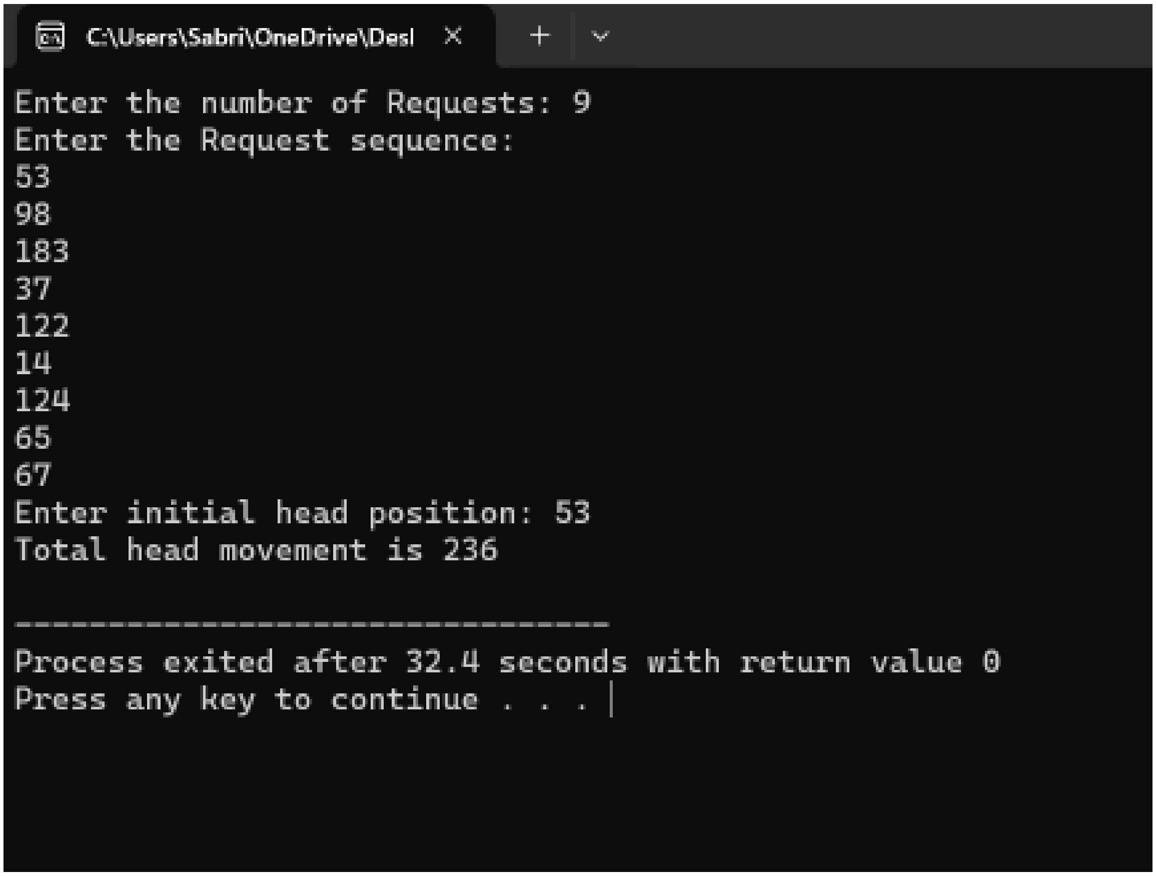
```
Enter the number of tracks: 8
Enter the tracks to be traversed: 98
183 37 122 14 124 65 67
Tracks traversed        Difference between tracks
183                     85
37                      146
122                     85
14                      108
124                     110
65                      59
67                      2

Average head movement: 85.00

--------------------------------
Process exited after 23.38 seconds with return value 0
Press any key to continue . . .
```

B) SSTF

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

int main() {
    int RQ[100], i, n, TotalHeadMoment = 0, initial, count = 0;

    // Asking user for the number of disk requests
    printf("Enter the number of Requests\n");
    scanf("%d", &n);

    // Asking user to input the sequence of requests
    printf("Enter the Requests sequence\n");
    for(i = 0; i < n; i++)
        scanf("%d", &RQ[i]);

    // Asking for the initial head position
    printf("Enter initial head position\n");
    scanf("%d", &initial);

    // SSTF Disk Scheduling logic
    while(count != n) {
        int min = 1000, d, index;

        // Iterate over all requests to find the closest one
        for(i = 0; i < n; i++) {
            // Calculate distance between the request and current head
            position d = abs(RQ[i] - initial);
            if(min > d) {
                min = d; // Update the minimum distance
                index = i; // Save the index of the closest request
            }
        }

        // Accumulate total head movement
        TotalHeadMoment += min;
        initial = RQ[index]; // Move the head to the selected request

        // Mark this request as "processed" (by setting it to a large number, 1000
        here) RQ[index] = 1000;
        count++;
    }

    // Output the total head movement
```

```
   printf("Total head movement is %d", TotalHeadMoment);
   return 0;
}
```

Output:

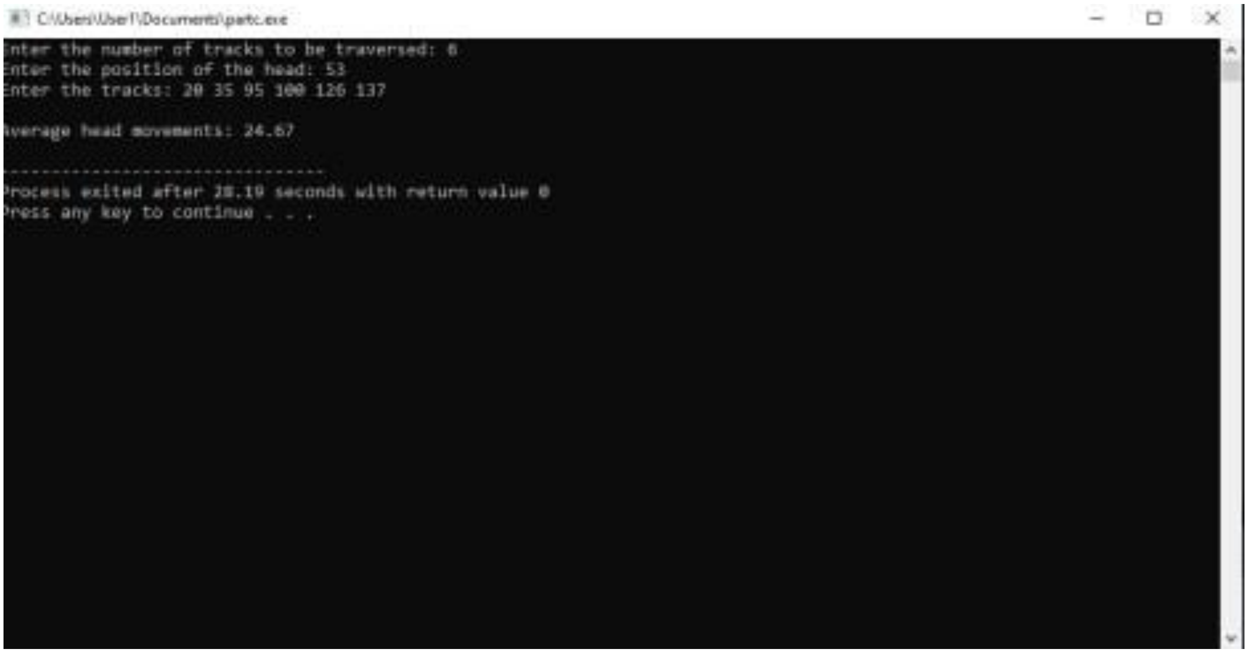


```
Enter the number of Requests: 9
Enter the Request sequence:
53
98
183
37
122
14
124
65
67
Enter initial head position: 53
Total head movement is 236

--------------------------------
Process exited after 32.4 seconds with return value 0
Press any key to continue . . .
```

C) SCAN

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

int main() {
    int t[20], d[20], h, i, j, n, temp, k, atr[20], tot, p, sum = 0;

    // Input the number of tracks and the initial position of the disk
    head printf("Enter the number of tracks to be traversed: ");
    scanf("%d", &n);
    printf("Enter the position of the head: ");
    scanf("%d", &h);

    t[0] = 0; // Add the starting position of the head to the tracks
    t[1] = h; // Store the initial head position
    printf("Enter the tracks: ");
    for(i = 2; i < n + 2; i++) {
        scanf("%d", &t[i]);
    }

    // Sort the tracks in ascending order
    for(i = 0; i < n + 2; i++) {
        for(j = 0; j < (n + 2) - i - 1; j++) {
            if(t[j] > t[j + 1]) {
                temp = t[j];
                t[j] = t[j + 1];
                t[j + 1] = temp;
            }
        }
    }

    // Find the index of the initial head position
    for(i = 0; i < n + 2; i++) {
        if(t[i] == h) {
            j = i; // j is the index of the head
            break;
        }
    }

    // Traverse the tracks in the direction towards the end (right
    side) p = 0;
    while(t[j] != 0) {
        atr[p] = t[j];
        j--;
        p++;
    }
    atr[p] = t[j]; // Add the 0 position (beginning of the track)

    // Now, process the remaining tracks in the reverse direction (left side)
    for(p = p + 1, k = j + 1; k < n + 2; k++, p++) {
        atr[p] = t[k];
    }

    // Calculate total head movement by adding differences
    for(i = 0; i < n + 1; i++) {
        if(atr[i] > atr[i + 1]) {
            d[i] = atr[i] - atr[i + 1];
        } else {
            d[i] = atr[i + 1] - atr[i];
        }
        sum += d[i]; // Add the absolute difference to the total head movement
    }

    // Output the average head movement
    printf("\nAverage head movements: %.2f\n", (float)sum / n);

    return 0;
}
```

Output:



```
Enter the number of tracks to be traversed: 6
Enter the position of the head: 53
Enter the tracks: 20 35 95 100 126 137

Average head movements: 24.67

--------------------------------
Process exited after 28.19 seconds with return value 0
Press any key to continue . . .
```