**ASSIGNMENT - V**

**TITLE :**

Implement various disk scheduling algorithms, including FIFO (First-Come-First-Serve),SSTF (Shortest Seek Time First), SCAN, and C-SCAN, and evaluate their performance based on average seek time.

**NAME :** Shinde Shubham Dnyandev, **DIV :** SY-B, **ROLL NO. :** 23107121.

**PROGRAM :**

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

int choice, track, no_req, head, head1, distance;
int disc_req[100], finish[100];

void sort()
{
   int i, j, temp;
   for (i = 0; i < no_req - 1; i++)
   {
      for (j = 0; j < no_req - i - 1; j++)
      {
         if (disc_req[j] > disc_req[j + 1])
         {
            temp = disc_req[j];
            disc_req[j] = disc_req[j + 1];
            disc_req[j + 1] = temp;
         }
      }
   }
}

int find_index()
{
   int i;
   for (i = 0; i < no_req; i++)
   {
      if (disc_req[i] >= head)
```

```
        {
            return i;
        }
    }
    return no_req;
}

void move(int start, int end, int direction)
{
    int i;
    if (direction == 1)
    {
        for (i = start; i < end; i++)
        {
            distance += abs(disc_req[i] - head);
            head = disc_req[i];
            printf(" %d =>", head);
        }
    }
    else
    {
        for (i = start; i >= end; i--)
        {
            distance += abs(head - disc_req[i]);
            head = disc_req[i];
            printf(" %d =>", head);
        }
    }
}

void fifo()
{
    int i;
    distance = 0;
    head = head1;

    printf("\nFIFO: %d =>", head);
    for (i = 0; i < no_req; i++)
    {
        distance += abs(head - disc_req[i]);
        head = disc_req[i];
        printf(" %d =>", head);
    }
```

```
    printf(" End\nTotal Distance Traversed = %d\n", distance);
}

void sstf()
{
    int min, diff, pending = no_req, i, index = 0;
    distance = 0;
    head = head1;

    for (i = 0; i < no_req; i++)
    {
        finish[i] = 0;
    }

    printf("\nSSTF: %d =>", head);
    while (pending > 0)
    {
        min = 9999;
        for (i = 0; i < no_req; i++)
        {
            diff = abs(head - disc_req[i]);
            if (finish[i] == 0 && diff < min)
            {
                min = diff;
                index = i;
            }
        }
        finish[index] = 1;
        distance += abs(head - disc_req[index]);
        head = disc_req[index];
        pending--;
        printf(" %d =>", head);
    }
    printf(" End\nTotal Distance Traversed = %d\n", distance);
}

void scan()
{
    int index;
    distance = 0;
    head = head1;

    sort();
```

```
    index = find_index();

    printf("\nSCAN: %d =>", head);

    move(index, no_req, 1);

    if (head < track - 1)
    {
        distance += abs((track - 1) - head);
        head = track - 1;
        printf(" %d =>", head);
    }

    move(index - 1, -1, 0);

    printf(" End\nTotal Distance Traversed = %d\n", distance);
}

void cscan()
{
    int index;
    distance = 0;
    head = head1;

    sort();
    index = find_index();

    printf("\nC-SCAN: %d =>", head);

    move(index, no_req, 1);

    if (head < track - 1)
    {
        distance += abs((track - 1) - head);
        head = track - 1;
        printf(" %d =>", head);
    }

    distance += head;
    head = 0;
    printf(" 0 =>");

    move(0, index, 1);
```

```
    printf(" End\nTotal Distance Traversed = %d\n", distance);
}

void menu()
{
    printf("\n\n*MENU*");
    printf("\n 1. Input data\n 2. FIFO \n 3. SSTF \n 4. SCAN \n 5. C-SCAN \n 6. Exit");
    printf("\n\n Enter your choice: ");
    scanf("%d", &choice);
}

void input()
{
    int i;
    printf("Enter Total number of tracks: ");
    scanf("%d", &track);
    printf("Enter total number of disk requests: ");
    scanf("%d", &no_req);
    printf("\nEnter disk requests in FCFS order:\n");
    for (i = 0; i < no_req; i++)
    {
        scanf("%d", &disc_req[i]);
    }
    printf("\nEnter current head position: ");
    scanf("%d", &head1);
}

int main()
{
    while (1)
    {
        menu();
        switch (choice)
        {
            case 1:
                input();
                break;
            case 2:
                fifo();
                break;
            case 3:
                sstf();
```

```
            break;
        case 4:
            scan();
            break;
        case 5:
            cscan();
            break;
        case 6:
            exit(0);
            break;
        default:
            printf("\nEnter a valid choice\n");
            break;
      }
   }
   return 0;
}
```

**OUTPUT :**

```
shubham@ShubhsPC:~$ gcc disk.c
shubham@ShubhsPC:~$ ./a.out


*MENU*
 1. Input data
 2. FIFO
 3. SSTF
 4. SCAN
 5. C-SCAN
 6. Exit

 Enter your choice: 1
Enter Total number of tracks: 200
Enter total number of disk requests: 7

Enter disk requests in FCFS order:
82
170
43
140
24
16
190

Enter current head position: 50


*MENU*
 1. Input data
 2. FIFO
 3. SSTF
 4. SCAN
 5. C-SCAN
 6. Exit

 Enter your choice: 2

FIFO: 50 => 82 => 170 => 43 => 140 => 24 => 16 => 190 => End
Total Distance Traversed = 642


*MENU*
 1. Input data
 2. FIFO
 3. SSTF
 4. SCAN
 5. C-SCAN
 6. Exit
```

```
 Enter your choice: 3

SSTF: 50 => 43 => 24 => 16 => 82 => 140 => 170 => 190 => End
Total Distance Traversed = 208


*MENU*
 1. Input data
 2. FIFO
 3. SSTF
 4. SCAN
 5. C-SCAN
 6. Exit

 Enter your choice: 4

SCAN: 50 => 82 => 140 => 170 => 190 => 199 => 43 => 24 => 16 => 0 => End
Total Distance Traversed = 348


*MENU*
 1. Input data
 2. FIFO
 3. SSTF
 4. SCAN
 5. C-SCAN
 6. Exit

 Enter your choice: 5

C-SCAN: 50 => 82 => 140 => 170 => 190 => 199 => 0 => 16 => 24 => 43 => End
Total Distance Traversed = 391


*MENU*
 1. Input data
 2. FIFO
 3. SSTF
 4. SCAN
 5. C-SCAN
 6. Exit

 Enter your choice: 6
shubham@ShubhsPC:~$
```