**[Exp-13] Write a program for the implementation of various Disk scheduling**

**algorithms (FCFS, SCAN, SSTF, C- SCAN).**

**Aim:** To write a program to implement the following types of Disk Scheduling Algorithms: FCFS, SCAN, SSTF, C-SCAN.

**Theory:** Every process requests the Operating system to access the disk for assigning it the I/O time. The operating system thus has to pick an algorithm to satisfy each request by maintaining the speed and efficiency of the program execution. This technique of choosing the order of process requests is called disk scheduling. The common disk scheduling methods include-

FCFS, SCAN, SSTF, C-SCAN.

**First Come First Serve (FCFS)**   
FCFS is the simplest [disk scheduling algorithm](https://www.geeksforgeeks.org/disk-scheduling-algorithms/). As the name suggests, this algorithm entertains requests in the order they arrive in the disk queue.

**SCAN (Elevator) algorithm**   
In SCAN disk scheduling algorithm, head starts from one end of the disk and moves towards the other end, servicing requests in between one by one and reach the other end. Then the direction of the head is reversed and the process continues as head continuously scan back and forth to access the disk.

**Shortest Seek Time First (SSTF)**   
Basic idea is the tracks which are closer to current disk head position should be serviced first in order to minimise the seek operations.

**Circular SCAN (C-SCAN)**

It is a modified version of SCAN disk scheduling algorithm. Like SCAN, it moves the head from one end servicing all the requests to the other end. However, as soon as the head reaches the other end, it immediately returns to the beginning of the disk without servicing any requests on the return trip and starts servicing again once reaches the beginning.

**Program Code**:

(1) FCFS

|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;    int size = 8;    void FCFS(int arr[], int head)  {      int seek\_count = 0;      int distance, cur\_track;        for (int i = 0; i < size; i++) {          cur\_track = arr[i];          distance = abs(cur\_track - head);          seek\_count += distance;          head = cur\_track;      }        cout << "Total number of seek operations = "           << seek\_count << endl;      cout << "Seek Sequence is" << endl;        for (int i = 0; i < size; i++) {          cout << arr[i] << endl;      }  }    int main()  {      int arr[size] = { 176, 79, 34, 60, 92, 11, 41, 114 };      int head = 50;        FCFS(arr, head);        return 0;  } |

**OUTPUT:**

Total number of seek operations = 510

Seek Sequence is

176

79

34

60

92

11

41

114

(2) SCAN

#include <bits/stdc++.h>

using namespace std;

int size = 8;

int disk\_size = 200;

void SCAN(int arr[], int head, string direction)

{

    int seek\_count = 0;

    int distance, cur\_track;

    vector<int> left, right;

    vector<int> seek\_sequence;

    if (direction == "left")

        left.push\_back(0);

    else if (direction == "right")

        right.push\_back(disk\_size - 1);

    for (int i = 0; i < size; i++) {

        if (arr[i] < head)

            left.push\_back(arr[i]);

        if (arr[i] > head)

            right.push\_back(arr[i]);

    }

    sort(left.begin(), left.end());

    sort(right.begin(), right.end());

    int run = 2;

    while (run--) {

        if (direction == "left") {

            for (int i = left.size() - 1; i >= 0; i--) {

                cur\_track = left[i];

                seek\_sequence.push\_back(cur\_track);

                distance = abs(cur\_track - head);

                seek\_count += distance;

                head = cur\_track;

            }

            direction = "right";

        }

        else if (direction == "right") {

            for (int i = 0; i < right.size(); i++) {

                cur\_track = right[i];

                seek\_sequence.push\_back(cur\_track);

                distance = abs(cur\_track - head);

                seek\_count += distance;

                head = cur\_track;

            }

            direction = "left";

        }

    }

    cout << "Total number of seek operations = "

         << seek\_count << endl;

    cout << "Seek Sequence is" << endl;

    for (int i = 0; i < seek\_sequence.size(); i++) {

        cout << seek\_sequence[i] << endl;

    }

}

int main()

{

    int arr[size] = { 176, 79, 34, 60,

                      92, 11, 41, 114 };

    int head = 50;

    string direction = "left";

    SCAN(arr, head, direction);

    return 0;

}

**OUTPUT:**

Total number of seek operations = 226

Seek Sequence is

41

34

11

0

60

79

92

114

176

(3) SSTF

#include <bits/stdc++.h>

using namespace std;

void calculatedifference(int request[], int head,

                         int diff[][2], int n)

{

    for(int i = 0; i < n; i++)

    {

        diff[i][0] = abs(head - request[i]);

    }

}

int findMIN(int diff[][2], int n)

{

    int index = -1;

    int minimum = 1e9;

    for(int i = 0; i < n; i++)

    {

        if (!diff[i][1] && minimum > diff[i][0])

        {

            minimum = diff[i][0];

            index = i;

        }

    }

    return index;

}

void shortestSeekTimeFirst(int request[],

                           int head, int n)

{

    if (n == 0)

    {

        return;

    }

    int diff[n][2] = { { 0, 0 } };

    int seekcount = 0;

    int seeksequence[n + 1] = {0};

    for(int i = 0; i < n; i++)

    {

        seeksequence[i] = head;

        calculatedifference(request, head, diff, n);

        int index = findMIN(diff, n);

        diff[index][1] = 1;

        seekcount += diff[index][0];

        head = request[index];

    }

    seeksequence[n] = head;

    cout << "Total number of seek operations = "

         << seekcount << endl;

    cout << "Seek sequence is : " << "\n";

    for(int i = 0; i <= n; i++)

    {

        cout << seeksequence[i] << "\n";

    }

}

int main()

{

    int n = 8;

    int proc[n] = { 176, 79, 34, 60, 92, 11, 41, 114 };

    shortestSeekTimeFirst(proc, 50, n);

    return 0;

}

**OUTPUT:**

Total number of seek operations = 204

Seek Sequence is

50

41

34

11

60

79

92

114

176

(4) C-SCAN

#include <bits/stdc++.h>

using namespace std;

int size = 8;

int disk\_size = 200;

void CSCAN(int arr[], int head)

{

    int seek\_count = 0;

    int distance, cur\_track;

    vector<int> left, right;

    vector<int> seek\_sequence;

    left.push\_back(0);

    right.push\_back(disk\_size - 1);

    for (int i = 0; i < size; i++) {

        if (arr[i] < head)

            left.push\_back(arr[i]);

        if (arr[i] > head)

            right.push\_back(arr[i]);

    }

    sort(left.begin(), left.end());

    sort(right.begin(), right.end());

    for (int i = 0; i < right.size(); i++) {

        cur\_track = right[i];

        seek\_sequence.push\_back(cur\_track);

        distance = abs(cur\_track - head);

        seek\_count += distance;

        head = cur\_track;

    }

    head = 0;

    seek\_count += (disk\_size - 1);

    for (int i = 0; i < left.size(); i++) {

        cur\_track = left[i];

        seek\_sequence.push\_back(cur\_track);

        distance = abs(cur\_track - head);

        seek\_count += distance;

        head = cur\_track;

    }

    cout << "Total number of seek operations = "

         << seek\_count << endl;

    cout << "Seek Sequence is" << endl;

    for (int i = 0; i < seek\_sequence.size(); i++) {

        cout << seek\_sequence[i] << endl;

    }

}

int main()

{

    int arr[size] = { 176, 79, 34, 60, 92, 11, 41, 114 };

    int head = 50;

    cout << "Initial position of head: " << head << endl;

    CSCAN(arr, head);

    return 0;

}

**OUTPUT:**

Initial position of head: 50

Total number of seek operations = 389

Seek Sequence is

60

79

92

114

176

199

0

11

34

41

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_