**Name : Ruturaj Sandip Sutar Roll No : 59 Div : B**

**Batch : 2 PRN:-12310720**

**Disk Scheduling Algorithms**

1. **FCFS**

**Code:-**

*#include* <iostream>

*#include* <vector>

*#include* <cmath>

using namespace std;

void FCFS(vector<int> &*requests*, int *head*) {

    int seek\_operations = 0;

    cout << "FCFS Order of execution: " << *head*;

*for* (int request : *requests*) {

        cout << " -> " << request;

        seek\_operations += abs(*head* - request);

*head* = request;

    }

    cout << "\nTotal Seek Operations: " << seek\_operations << endl;

}

int main() {

    vector<int> requests = {98, 183, 37, 122, 14, 124, 65, 67};

    int head = 53;

    FCFS(requests, head);

*return* 0;

}

**Output:-**

FCFS Order of execution: 53 -> 98 -> 183 -> 37 -> 122 -> 14 -> 124 -> 65 -> 67

Total Seek Operations: 640

1. **SSTF**

**Code:-**

*#include* <iostream>

*#include* <vector>

*#include* <cmath>

*#include* <algorithm>

*#include* <limits.h>

using namespace std;

void SSTF(vector<int> &*requests*, int *head*) {

    int seek\_operations = 0;

    vector<bool> visited(*requests*.size(), false);

    cout << "SSTF Order of execution: " << *head*;

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

        int min\_distance = INT\_MAX;

        int index = -1;

*for* (int j = 0; j < *requests*.size(); ++j) {

*if* (!visited[j] && abs(*requests*[j] - *head*) < min\_distance) {

                min\_distance = abs(*requests*[j] - *head*);

                index = j;

            }

        }

        visited[index] = true;

        cout << " -> " << *requests*[index];

        seek\_operations += abs(*head* - *requests*[index]);

*head* = *requests*[index];

    }

    cout << "\nTotal Seek Operations: " << seek\_operations << endl;

}

int main() {

    vector<int> requests = {98, 183, 37, 122, 14, 124, 65, 67};

    int head = 53;

    SSTF(requests, head);

*return* 0;

}

**Output:-**

SSTF Order of execution: 53 -> 65 -> 67 -> 37 -> 14 -> 98 -> 122 -> 124 -> 183

Total Seek Operations: 236

1. **SCAN**

**Code:-**

*#include* <iostream>

*#include* <vector>

*#include* <algorithm>

using namespace std;

void SCAN(vector<int> &*requests*, int *head*, int *disk\_size*, string *direction*) {

    int seek\_operations = 0;

*requests*.push\_back(0);

*requests*.push\_back(*disk\_size* - 1);

    sort(*requests*.begin(), *requests*.end());

    int pos = distance(*requests*.begin(), lower\_bound(*requests*.begin(), *requests*.end(), *head*));

    cout << "SCAN Order of execution: " << *head*;

*if* (*direction* == "left") {

*for* (int i = pos; i >= 0; --i) {

            cout << " -> " << *requests*[i];

            seek\_operations += abs(*head* - *requests*[i]);

*head* = *requests*[i];

        }

*for* (int i = pos + 1; i < *requests*.size(); ++i) {

            cout << " -> " << *requests*[i];

            seek\_operations += abs(*head* - *requests*[i]);

*head* = *requests*[i];

        }

    } *else* *if* (*direction* == "right") {

*for* (int i = pos; i < *requests*.size(); ++i) {

            cout << " -> " << *requests*[i];

            seek\_operations += abs(*head* - *requests*[i]);

*head* = *requests*[i];

        }

*for* (int i = pos - 1; i >= 0; --i) {

            cout << " -> " << *requests*[i];

            seek\_operations += abs(*head* - *requests*[i]);

*head* = *requests*[i];

        }

    }

    cout << "\nTotal Seek Operations: " << seek\_operations << endl;

}

int main() {

    vector<int> requests = {98, 183, 37, 122, 14, 124, 65, 67};

    int head = 53;

    int disk\_size = 200;

    string direction = "left";

    SCAN(requests, head, disk\_size, direction);

*return* 0;

}

**Output:-**

SCAN Order of execution: 53 -> 65 -> 37 -> 14 -> 0 -> 67 -> 98 -> 122 -> 124 -> 183 -> 199

Total Seek Operations: 276

1. **C-SCAN**

**Code:-**

*#include* <iostream>

*#include* <vector>

*#include* <algorithm>

using namespace std;

void CSCAN(vector<int> &*requests*, int *head*, int *disk\_size*) {

    int seek\_operations = 0;

*requests*.push\_back(0);

*requests*.push\_back(*disk\_size* - 1);

    sort(*requests*.begin(), *requests*.end());

    int pos = distance(*requests*.begin(), lower\_bound(*requests*.begin(), *requests*.end(), *head*));

    cout << "C-SCAN Order of execution: " << *head*;

*for* (int i = pos; i < *requests*.size(); ++i) {

        cout << " -> " << *requests*[i];

        seek\_operations += abs(*head* - *requests*[i]);

*head* = *requests*[i];

    }

*// Jump to the beginning of the disk*

*head* = 0;

    seek\_operations += *requests*.back();

    cout << " -> " << *head*;

*for* (int i = 0; i < pos; ++i) {

        cout << " -> " << *requests*[i];

        seek\_operations += abs(*head* - *requests*[i]);

*head* = *requests*[i];

    }

    cout << "\nTotal Seek Operations: " << seek\_operations << endl;

}

int main() {

    vector<int> requests = {98, 183, 37, 122, 14, 124, 65, 67};

    int head = 53;

    int disk\_size = 200;

    CSCAN(requests, head, disk\_size);

*return* 0;

}

**Output:-**

C-SCAN Order of execution: 53 -> 65 -> 67 -> 98 -> 122 -> 124 -> 183 -> 199 -> 0 -> 0 -> 14 -> 37

Total Seek Operations: 382