ASSIGNMENT 10
28/11/23

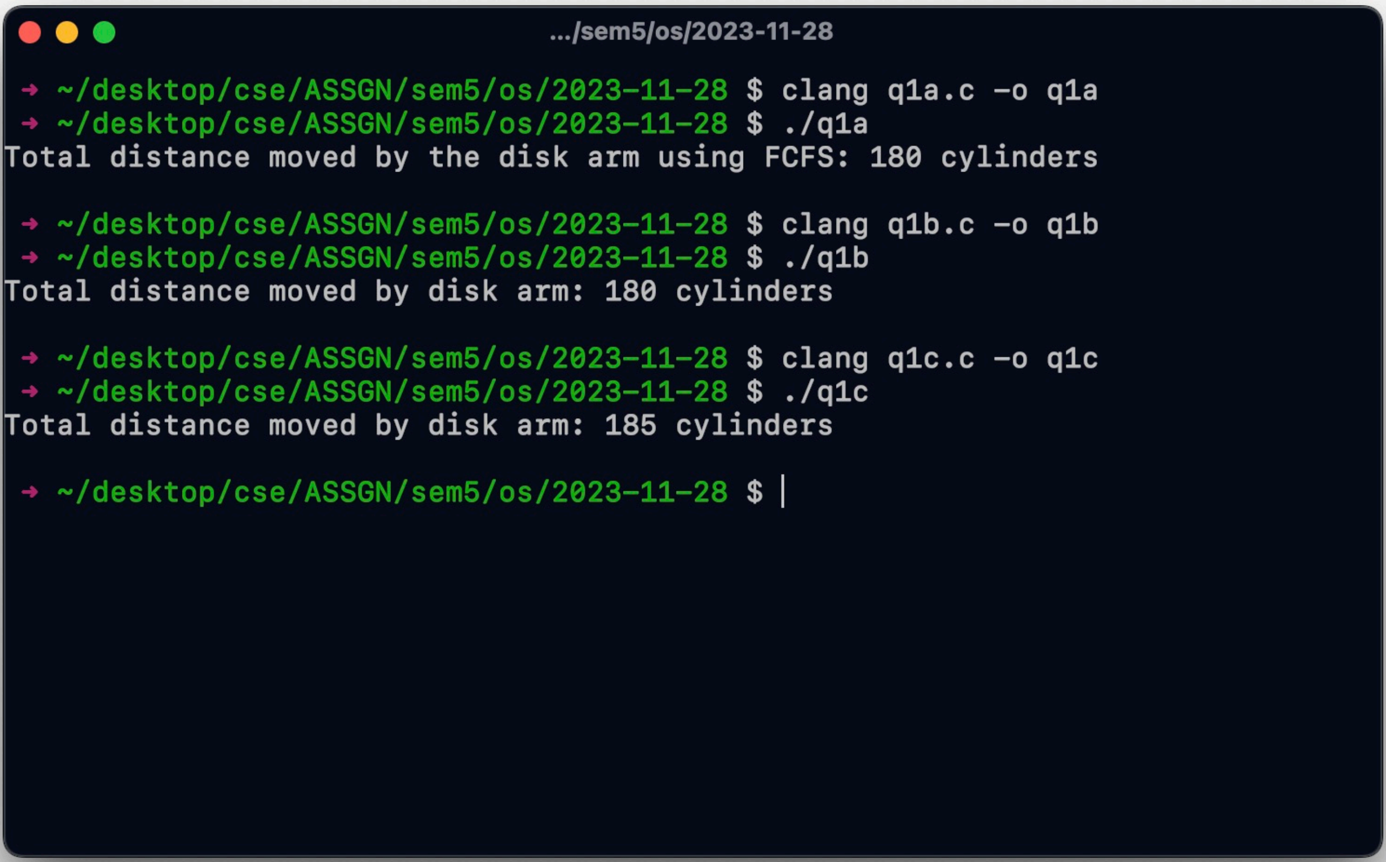
NAME : SHRESTH SONKAR
REGNO : 20214272
GROUP : CS5D
TOPIC : OS LAB
CODE : CS-15203

```c
// Q1a : Write C programs to implement FCFS disk
scheduling algorithms

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

int calculateTotalMovement(int initialPosition, int
requestQueue[], int numOfRequests) {
    int totalMovement = 0;
    int currentPosition = initialPosition;

    for (int i = 0; i < numOfRequests; ++i) {
        int distance = abs(requestQueue[i] -
currentPosition);
        totalMovement += distance;
        currentPosition = requestQueue[i];
    }

    return totalMovement;
}

int main() {
    int diskSize = 201;
    int initialPosition = 95;
    int requestQueue[] = {30, 85, 90, 100, 105, 110,
135, 145};

    int numOfRequests = sizeof(requestQueue) /
sizeof(requestQueue[0]);
    int totalMovement =
calculateTotalMovement(initialPosition, requestQueue,
numOfRequests);

    printf("Total distance moved by the disk arm using
FCFS: %d cylinders\n\n", totalMovement);
    return 0;
}
```

```c
// Q1b : Write C programs to implement SCAN disk
scheduling algorithms

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

#define TOTAL_CYLINDERS 201
#define DIRECTION_RIGHT 1
#define DIRECTION_LEFT 0

void sortRequests(int req[], int n) {
    int i, j, temp;
    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (req[i] > req[j]) {
                temp = req[i];
                req[i] = req[j];
                req[j] = temp;
            }
        }
    }
}

int findNextIndex(int req[], int n, int start, int
direction) {
    int index = -1;
    if (direction == DIRECTION_RIGHT) {
        for (int i = start; i < n; i++) {
            if (req[i] >= 0) {
                index = i;
                break;
            }
        }
    } else {
        for (int i = start; i >= 0; i--) {
            if (req[i] >= 0) {
                index = i;
                break;
            }
        }
    }
    return index;
}
```

```
int calculateTotalDistance(int req[], int n, int start)
{
    int totalDistance = 0;
    int direction = DIRECTION_RIGHT;
    int currentPos = start;

    sortRequests(req, n);
    int nextIndex = findNextIndex(req, n, 0,
direction);

    while (nextIndex != -1) {
        totalDistance += abs(req[nextIndex] -
currentPos);
        currentPos = req[nextIndex];
        req[nextIndex] = -1;

        if (currentPos == 0 || currentPos ==
TOTAL_CYLINDERS - 1)
            direction = !direction;

        if (direction == DIRECTION_RIGHT)
            nextIndex = findNextIndex(req, n, nextIndex
+ 1, direction);
        else
            nextIndex = findNextIndex(req, n, nextIndex
- 1, direction);
    }

    return totalDistance;
}

int main() {
    int diskRequests[] = {30, 85, 90, 100, 105, 110,
135, 145};
    int numOfRequests = sizeof(diskRequests) /
sizeof(diskRequests[0]);
    int currentHeadPos = 95;

    int totalDistance =
calculateTotalDistance(diskRequests, numOfRequests,
currentHeadPos);
    printf("Total distance moved by disk arm: %d
cylinders\n\n", totalDistance);
    return 0;
}
```

```c
// Q1c : Write C programs to implement SSTF disk
scheduling algorithms
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>

#define TOTAL_CYLINDERS 201

int findShortestSeekTime(int req[], int n, int
currentPos, int visited[]) {
    int minDistance = INT_MAX;
    int nextIndex = -1;

    for (int i = 0; i < n; i++) {
        if (!visited[i]) {
            int distance = abs(req[i] - currentPos);
            if (distance < minDistance) {
                minDistance = distance;
                nextIndex = i;
            }
        }
    }

    return nextIndex;
}

int calculateTotalDistance(int req[], int n, int start)
{
    int totalDistance = 0;
    int currentPos = start;
    int visited[n];

    for (int i = 0; i < n; i++)
        visited[i] = 0;

    for (int i = 0; i < n; i++) {
        int nextIndex = findShortestSeekTime(req, n,
currentPos, visited);
        totalDistance += abs(req[nextIndex] -
currentPos);
        currentPos = req[nextIndex];
        visited[nextIndex] = 1;
    }

    return totalDistance;
}

int main() {
    int diskRequests[] = {30, 85, 90, 100, 105, 110, 
135, 145};
    int numOfRequests = sizeof(diskRequests) / 
sizeof(diskRequests[0]);
    int currentHeadPos = 95;

    int totalDistance = 
calculateTotalDistance(diskRequests, numOfRequests, 
currentHeadPos);
    printf("Total distance moved by disk arm: %d 
cylinders\n\n", totalDistance);
    return 0;
}
```



...../sem5/os/2023-11-28

```
→ ~/desktop/cse/ASSGN/sem5/os/2023-11-28 $ clang q1a.c -o q1a
→ ~/desktop/cse/ASSGN/sem5/os/2023-11-28 $ ./q1a
Total distance moved by the disk arm using FCFS: 180 cylinders

→ ~/desktop/cse/ASSGN/sem5/os/2023-11-28 $ clang q1b.c -o q1b
→ ~/desktop/cse/ASSGN/sem5/os/2023-11-28 $ ./q1b
Total distance moved by disk arm: 180 cylinders

→ ~/desktop/cse/ASSGN/sem5/os/2023-11-28 $ clang q1c.c -o q1c
→ ~/desktop/cse/ASSGN/sem5/os/2023-11-28 $ ./q1c
Total distance moved by disk arm: 185 cylinders

→ ~/desktop/cse/ASSGN/sem5/os/2023-11-28 $ |
```