CODE >>

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include <limits.h>

void shortestSeekTimeFirst(int request[], int head, int n)

{

int seek\_sequence[n];

int seek\_count = 0;

bool visited[n];

for (int i = 0; i < n; i++)

{

visited[i] = false;

}

for (int i = 0; i < n; i++)

{

int min\_distance = INT\_MAX;

int index = -1;

for (int j = 0; j < n; j++)

{

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

{

index = j;

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

}

}

visited[index] = true;

seek\_sequence[i] = request[index];

seek\_count += min\_distance;

head = request[index];

}

printf("SSTF Disk Scheduling Algorithm:\n");

printf("Total seek time: %d\n", seek\_count);

printf("Seek sequence: ");

for (int i = 0; i < n; i++)

{

printf("%d ", seek\_sequence[i]);

}

printf("\n\n");

}

void scan(int request[], int head, int n, int direction, int disk\_size)

{

int seek\_sequence[n + 1];

int seek\_count = 0;

int distance;

int cur\_track;

for (int i = 0; i < n; i++)

{

for (int j = i + 1; j < n; j++)

{

if (request[i] > request[j])

{

int temp = request[i];

request[i] = request[j];

request[j] = temp;

}

}

}

int index;

for (index = 0; index < n; index++)

{

if (head < request[index])

{

break;

}

}

int left = index - 1;

int right = index;

int seek\_sequence\_index = 0;

// Handle right direction

if(direction==1)

{

while (right < n)

{

cur\_track = request[right];

distance = abs(cur\_track - head);

seek\_count += distance;

seek\_sequence[seek\_sequence\_index++] = cur\_track;

head = cur\_track;

right++;

}

cur\_track = disk\_size-1;

distance = abs(cur\_track - head);

seek\_count += distance;

seek\_sequence[seek\_sequence\_index++] = cur\_track;

head = cur\_track;

while (left >= 0)

{

cur\_track = request[left];

distance = abs(cur\_track - head);

seek\_count += distance;

seek\_sequence[seek\_sequence\_index++] = cur\_track;

head = cur\_track;

left--;

}

}

else if(direction==0)

{

while (left >= 0)

{

cur\_track = request[left];

distance = abs(cur\_track - head);

seek\_count += distance;

seek\_sequence[seek\_sequence\_index++] = cur\_track;

head = cur\_track;

left--;

}

cur\_track = 0;

distance = abs(cur\_track - head);

seek\_count += distance;

seek\_sequence[seek\_sequence\_index++] = cur\_track;

head = cur\_track;

while (right < n)

{

cur\_track = request[right];

distance = abs(cur\_track - head);

seek\_count += distance;

seek\_sequence[seek\_sequence\_index++] = cur\_track;

head = cur\_track;

right++;

}

}

printf("SCAN Disk Scheduling Algorithm:\n");

printf("Total seek time: %d\n", seek\_count);

printf("Seek sequence: ");

for (int i = 0; i <= n; i++)

{

printf("%d ", seek\_sequence[i]);

}

printf("\n\n");

}

void cLook(int request[], int head, int n, int direction, int disk\_size)

{

int seek\_sequence[n];

int seek\_count = 0;

int distance;

int cur\_track;

for (int i = 0; i < n - 1; i++)

{

for (int j = i + 1; j < n; j++)

{

if (request[i] > request[j])

{

int temp = request[i];

request[i] = request[j];

request[j] = temp;

}

}

}

int index;

for (index = 0; index < n; index++)

{

if (head < request[index])

{

break;

}

}

int left = index-1;

int right = index;

for (int i = 0; i < n; i++)

{

if (direction == 0)

{

if (left <= 0)

{

cur\_track = request[0];

left = n-1;

}

else

{

cur\_track = request[left];

left--;

}

}

else if(direction == 1)

{

cur\_track = request[right];

right = (right + 1) % n;

}

distance = abs(cur\_track - head);

seek\_sequence[i] = cur\_track;

head = cur\_track;

seek\_count += distance;

}

printf("C-LOOK Disk Scheduling Algorithm:\n");

printf("Total seek time: %d\n", seek\_count);

printf("Seek sequence: ");

for (int i = 0; i < n; i++)

{

printf("%d ", seek\_sequence[i]);

}

printf("\n\n");

}

int main()

{

int n, head, direction;

int disk\_size = 200;

printf("Enter the number of requests: ");

scanf("%d", &n);

printf("Enter the Disk Size : ");

scanf("%d",&disk\_size);

int proc[n];

printf("Enter the requests: ");

for(int i = 0; i < n; i++)

{

scanf("%d", &proc[i]);

}

printf("Enter the initial head position: ");

scanf("%d", &head);

printf("Enter the direction (0 for left, 1 for right): ");

scanf("%d", &direction);

for (int i = 0; i < n - 1; i++)

{

for (int j = i + 1; j < n; j++)

{

if (proc[i] > proc[j])

{

int temp = proc[i];

proc[i] = proc[j];

proc[j] = temp;

}

}

}

shortestSeekTimeFirst(proc, head, n);

scan(proc, head, n, direction, disk\_size);

cLook(proc, head, n, direction, disk\_size);

return 0;

}

OUTPUT >>

Enter the number of requests: 7

Enter the Disk Size : 200

Enter the requests: 82

170

43

140

24

16

190

Enter the initial head position: 50

Enter the direction (0 for left, 1 for right): 1

SSTF Disk Scheduling Algorithm:

Total seek time: 208

Seek sequence: 43 24 16 82 140 170 190

SCAN Disk Scheduling Algorithm:

Total seek time: 332

Seek sequence: 82 140 170 190 199 43 24 16

C-LOOK Disk Scheduling Algorithm:

Total seek time: 341

Seek sequence: 82 140 170 190 16 24 43

PS P:\VsCode>