|  |  |
| --- | --- |
| Exp no: 05 | DISK SCHEDULING |
| Date: |

# AIM:

# DESCRIPTION:

|  |  |
| --- | --- |
| EXP NO: 5(A) | FIRST COME FIRST SERVE |
| DATE: |

# AIM:

# FIRST COME FIRST SERVE DISK SCHEDULING:

# ALGORITHM:

# SOURCE CODE:

#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 };

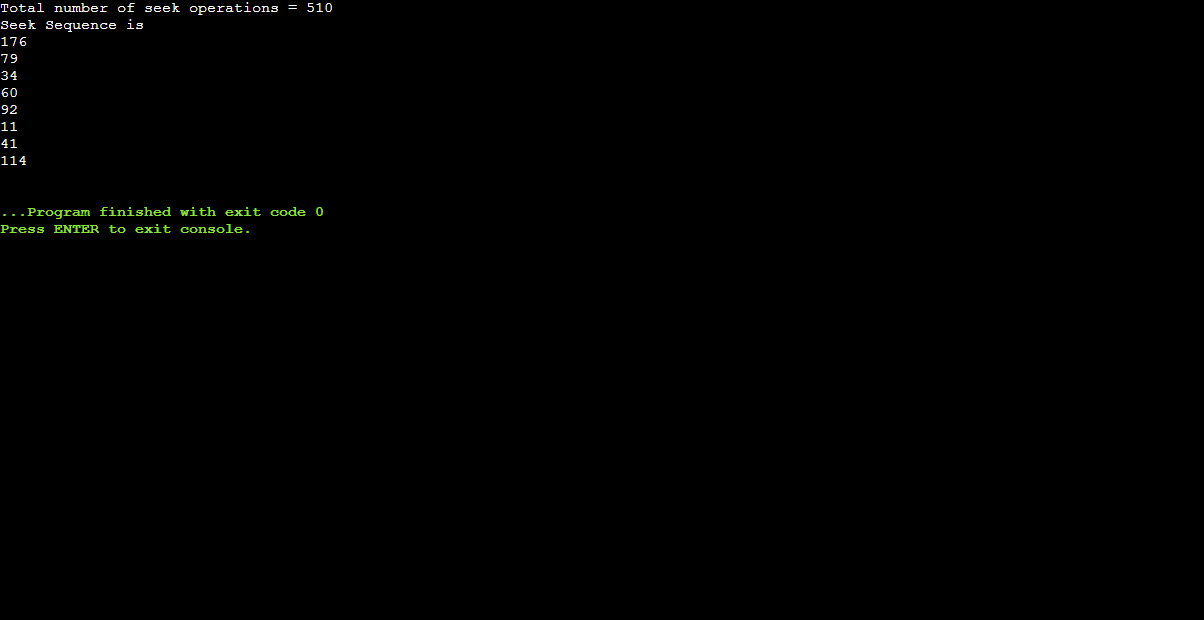
int head = 50;

FCFS(arr, head);

return 0;

}

# Output:



# RESULT:

|  |  |
| --- | --- |
| Exp no: 5(B) | SHORTEST SEEK TIME FIRST |
| Date: |

# AIM:

# SHORTEST SEEK TIME FIRST:

# ALGORITHM:

# SOURCE CODE:

#include <bits/stdc++.h>

using namespace std;

const int N=100005;

int n;

int head;

int done[N];

int positions[N];

void sstf(void)

{

int movement,shortest,index;

movement = 0;

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

{

index = 0;

shortest = INT\_MAX;

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

{

if(abs(head - positions[k]) < shortest && !done[k])

{

index = k;

shortest = abs(head - positions[k]);

}

}

done[index] = true;

movement += shortest;

head = positions[index];

}

printf("Total Head Movement %d Cylinders\n",movement);

return;

}

int main()

{

cout<<"Initial Head Position: ";

cin>>head;

cout<<"Queue Size: ";

cin>>n;

cout<<"Queue:\n";

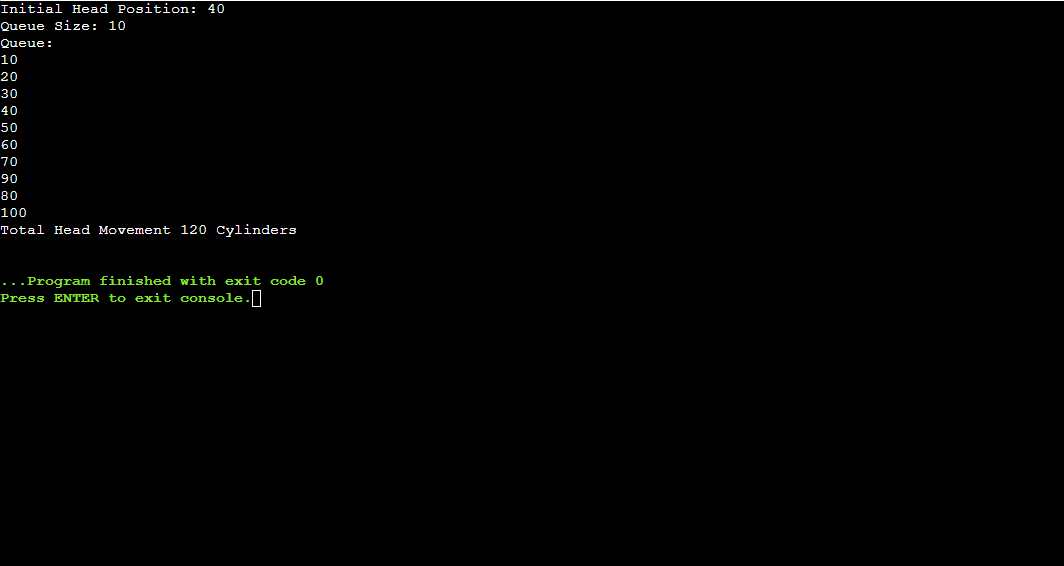
for(int i=0; i<n; i++) cin>>positions[i];

sstf();

return 0;

}

# OUTPUT:



# RESULT:

|  |  |
| --- | --- |
| EXP NO: 5(C) | SCAN DISK SCHEDULING |
| DATE: |

# AIM:

# SCAN DISK SCHEDULING

# ALGORITHM:

# SOURCE CODE:

#include <bits/stdc++.h>

using namespace std;

const int N=100005;

int n;

int head;

int direction;

int done[N];

int positions[N];

void scan(void)

{

int movement,best,index,complete;

movement = 0;

complete = 0;

while(complete < n)

{

index = -1;

if(direction == 0) best = INT\_MIN;

if(direction == 1) best = INT\_MAX;

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

{

if(!done[i])

{

if(direction == 0 && positions[i] > best && positions[i] < head)

{

index = i;

best = positions[i];

}

if(direction == 1 && positions[i] < best && positions[i] > head)

{

index = i;

best = positions[i];

}

}

}

if(index >= 0)

{

complete++;

done[index] = true;

movement += abs(head - best);

head = positions[index];

}

else

{

direction = 1 - direction;

}

}

cout<<"Total Head Movement "<<movement<<" Cylinders\n";

return;

}

int main()

{

cout<<"Initial Head Position: ";

cin>>head;

cout<<"Initial Direction [0/1]: ";

cin>>direction;

cout<<"Queue Size: ";

cin>>n;

cout<<"Queue:\n";

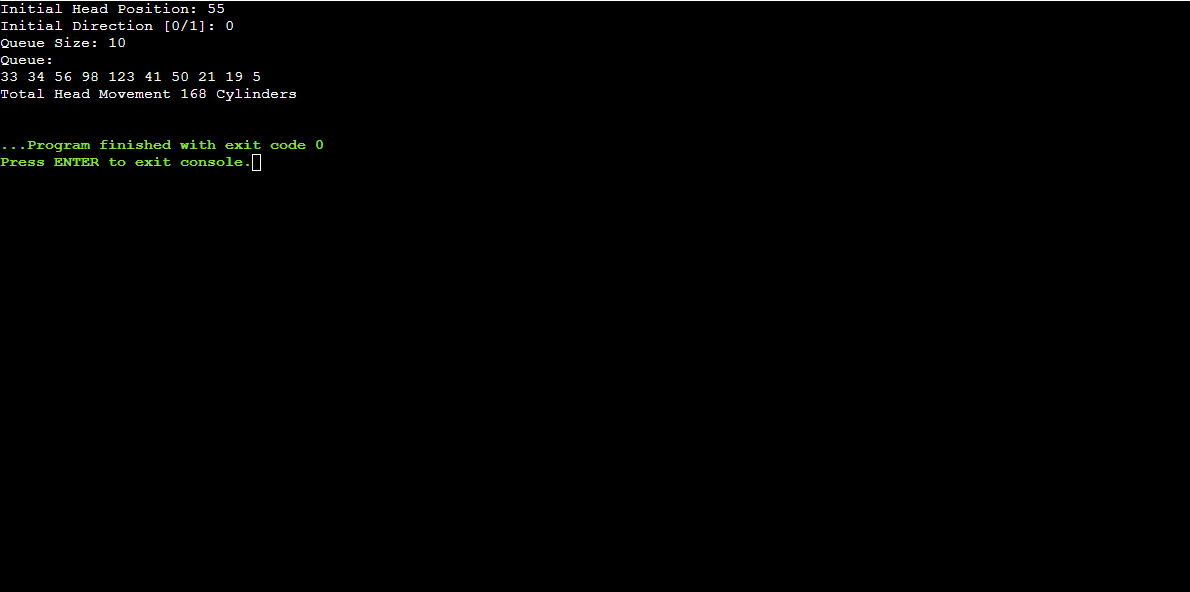
for(int i=0; i<n; i++) cin>>positions[i];

scan();

return 0;

}

# OUTPUT:



# RESULT:

|  |  |
| --- | --- |
| EXP NO :5(D) | LOOK DISK SCHEDULING |
| DATE: |

# AIM:

# LOOK DISK SCHEDULING

# ALGORITHM:

# SOURCE CODE:

#include<bits/stdc++.h>

using namespace std;

int main(){

int i,j,k,n,m,sum=0,x,y,h;

cout<<"Enter the size of disk\n";

cin>>m;

cout<<"Enter number of requests\n";

cin>>n;

cout<<"Enter the requests\n";

vector <int> a(n),l;

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

cin>>a[i];

}

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

if(a[i]>m){

cout<<"Error, Unknown position\n";

return 0;

}

}

cout<<"Enter the head position\n";

cin>>h;

a.push\_back(h);

sort(a.begin(),a.end());

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

if(h==a[i])

break;

}

k=i;

if(k<n/2){

for(i=k;i<a.size();i++){

l.push\_back(a[i]);

}

for(i=k-1;i>=0;i--){

l.push\_back(a[i]);

}

}

else{

for(i=k;i>=0;i--){

l.push\_back(a[i]);

}

for(i=k+1;i<a.size();i++){

l.push\_back(a[i]);

}

}

int temp=l[0];

cout<<temp;

for(i=1;i<l.size();i++){

cout<<" -> "<<l[i]<<' ';

sum+=abs(l[i]-temp);

temp=a[i];

}

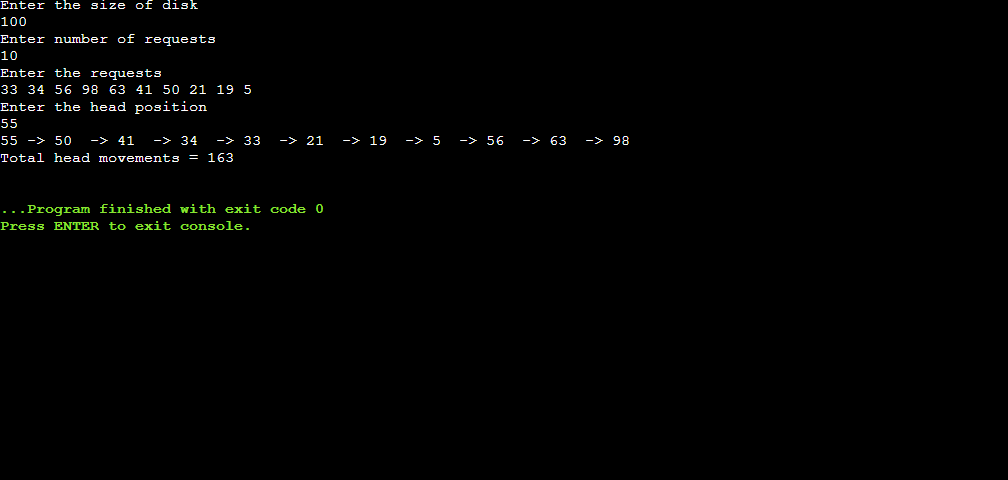
cout<<'\n';

cout<<"Total head movements = "<< sum<<'\n';

return 0;

}

# OUTPUT:



# RESULT: