CSE2008 (Operating Systems) Lab-9

KHAN MOHD OWAIS RAZA
20BCD7138

Q.1 Write a program for following disk scheduling algorithms.

1. First Come First Serve
2. Shortest Seek Time First

(1) First Come First Serve

Java Code :-

```
/* KHAN MOHD OWAIS RAZA */
/* 20BCD7138 */
/* CSE2008 (Operating Systems) Lab Practical-9 */
/* Write program for First Come First Serve
   disk scheduling algorithm
*/
package owaisraza.CSE2008_Lab9;
public class MyClass1{
static int size = 8;
static void FIRST_COME_FIRST_SERVE(int ARRAY[], int HEAD){
int SEEK_COUNT = 0;
int LENGTH, TRACK;
for (int X = 0; X < size; X++){
TRACK = ARRAY[X];
LENGTH = Math.abs(TRACK - HEAD);
SEEK_COUNT += LENGTH;
HEAD = TRACK;
}
System.out.println("** FIRST COME FIRST SERVE DISK SCHEDULING ALGORITHM
**");
System.out.println("  ");
System.out.println("TOTAL SEEK COUNT = " +SEEK_COUNT);
System.out.println("SEEK SEQUENCE IS AS FOLLOWS:");
for (int X = 0; X < size; X++){
System.out.println(ARRAY[X]);
}}
public static void main(String[] args){
int ARRAY[] = {40, 30, 20, 40, 20, 90, 80, 70};
int HEAD = 50;
FIRST_COME_FIRST_SERVE(ARRAY, HEAD);
}}
```

```
<terminated> MyClass1 [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe
** FIRST COME FIRST SERVE DISK SCHEDULING ALGORITHM **

TOTAL SEEK COUNT = 160
SEEK SEQUENCE IS AS FOLLOWS:
40
30
20
40
20
90
80
70
```

C Code :-

```c
/* KHAN MOHD OWAIS RAZA */
/* 20BCD7138 */
/* CSE2008 (Operating Systems) Lab Practical-9 */
/* Write program for First Come First Serve
   disk scheduling algorithm
*/
#include <stdio.h>
void FIRST_COME_FIRST_SERVE(int QUEUE[], int HEAD, int X) {
double SEEK_TIME = 0.0;
int LENGTH = 0;
int A = 0;
printf("\nSTARTING HEAD = %d", HEAD);
printf("\nQUEUE SEQUENCE = ");
for (A = 0; A < X; A++){
printf(" %d ->", QUEUE[A]);
}
for (A = 0; A < X; A++){
LENGTH = QUEUE[A] - HEAD;
if (LENGTH < 0){
LENGTH = -LENGTH;
}
HEAD = QUEUE[A];
SEEK_TIME += LENGTH;
}
printf("\nTOTAL SEEK TIME = %lf", SEEK_TIME);
printf("\nAVERAGE SEEK TIME = %lf\n", SEEK_TIME/X);
}
int main()
{
int QUEUE[] = {40, 30, 20, 40, 20, 90, 80, 70};
int X = sizeof(QUEUE)/sizeof(QUEUE[0]);
int HEAD = 25;
FIRST_COME_FIRST_SERVE(QUEUE, HEAD, X);
return 0;
}
```

C:\Users\Owais\Documents\FCFS.exe

```
STARTING HEAD = 25
QUEUE SEQUENCE =  40 -> 30 -> 20 -> 40 -> 20 -> 90 -> 80 -> 70
TOTAL SEEK TIME = 165.000000
AVERAGE SEEK TIME = 20.625000

--------------------------------
Process exited after 0.1917 seconds with return value 0
Press any key to continue . . .
```

(2) Shortest Seek Time First

Java Code :-

```
/* KHAN MOHD OWAIS RAZA */
/* 20BCD7138 */
/* CSE2008 (Operating Systems) Lab Practical-9 */
/* Write program for Shortest Seek Time First
   disk scheduling algorithm
*/
package owaisraza.CSE2008_Lab9;
public class MyClass2 {
int LENGTH = 0;
boolean ACCESS = false;
public static void Difference(int QUEUE[], int HEAD, MyClass2
DIFFERENCE[]){
for (int X = 0; X < DIFFERENCE.length; X++)
DIFFERENCE[X].LENGTH = Math.abs(QUEUE[X] - HEAD);
}
public static int Minimum(MyClass2 DIFFERENCE[]){
int INDEX = -1, MINIMUM = Integer.MAX_VALUE;
for (int X = 0; X < DIFFERENCE.length; X++) {
if (!DIFFERENCE[X].ACCESS && MINIMUM > DIFFERENCE[X].LENGTH) {
MINIMUM = DIFFERENCE[X].LENGTH;
INDEX = X;
}}
return INDEX;
}
public static void SHORTEST_TIME_SEEK_FIRST(int REQUEST[], int
HEAD){
if (REQUEST.length == 0)
return;
MyClass2 DIFFERENCE[] = new MyClass2[REQUEST.length];
for (int X = 0; X < DIFFERENCE.length; X++)
DIFFERENCE[X] = new MyClass2();
int SEEK_COUNT = 0;
int[] SEQUENCE = new int[REQUEST.length + 1];
for (int X = 0; X < REQUEST.length; X++) {
SEQUENCE[X] = HEAD;
Difference(REQUEST, HEAD, DIFFERENCE);
int INDEX = Minimum(DIFFERENCE);
DIFFERENCE[INDEX].ACCESS = true;
SEEK_COUNT += DIFFERENCE[INDEX].LENGTH;
HEAD = REQUEST[INDEX];
}
SEQUENCE[SEQUENCE.length - 1] = HEAD;
System.out.println("** SHORTEST TIME SEEK FIRST DISK SCHEDULING
ALGORITHM **");
System.out.println("  ");
System.out.println("TOTAL SEEK COUNT = " +SEEK_COUNT);
```

```java
System.out.println("SEEK SEQUENCE IS AS FOLLOWS:");
for (int X = 0; X < SEQUENCE.length; X++)
System.out.println(SEQUENCE[X]);
}
public static void main(String[] args){
int ARRAY[] = {40, 30, 20, 40, 20, 90, 80, 70};
SHORTEST_TIME_SEEK_FIRST(ARRAY, 50);
}}
```

```
<terminated> MyClass2 (1) [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe
** SHORTEST TIME SEEK FIRST DISK SCHEDULING ALGORITHM **

TOTAL SEEK COUNT = 100
SEEK SEQUENCE IS AS FOLLOWS:
50
40
40
30
20
20
70
80
90
```

C++ Code :-

```cpp
#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 NO. OF REQUESTS\n";
cin>>n;
cout<<"ENTER THE REQUESTS\n";
vector <int> a(n),b;
map <int,int> mp;
for(i=0;i<n;i++){
cin>>a[i];
mp[a[i]]++;
}
for(i=0;i<n;i++){
if(a[i]>m){
cout<<"ERROR...UNKNOWN POSITION !!"<<a[i]<<"\n";
return 0;
}}
cout<<"ENTER THE HEAD POSITION\n";
cin>>h;
int temp=h;
int ele;
b.push_back(h);
int count=0;
while(count<n){
```

```
int diff=999999;
for(auto q:mp){
if(abs(q.first-temp)<diff){
ele=q.first;
diff=abs(q.first-temp);
}}
mp[ele]--;
if(mp[ele]==0){
mp.erase(ele);
}
b.push_back(ele);
temp=ele;
count++;
}
cout<<b[0];
temp=b[0];
for(i=1;i<b.size();i++){
cout<<" -> "<<b[i];
sum+=abs(b[i]-temp);
temp=b[i];
}
cout<<'\n';
cout<<"TOTAL HEAD MOVEMENTS = "<< sum<<'\n';
cout<<"AVERAGE HEAD MOVEMENTS = "<<(float)sum/n<<'\n';
return 0;
}
```

Output

```
/tmp/X94wz4bvar.o
ENTER THE SIZE OF DISK
199
ENTER NO. OF REQUESTS
8
ENTER THE REQUESTS
98 183 37 122 14 124 65 67
ENTER THE HEAD POSITION
53
53 -> 65 -> 67 -> 37 -> 14 -> 98 -> 122 -> 124 -> 183
TOTAL HEAD MOVEMENTS = 236
AVERAGE HEAD MOVEMENTS = 29.5
```