

OS LAB-8

Q) a) Write a C program to simulate page replacement algorithms a) FIFO b) LRU c) Optimal

b) Write a C program to simulate disk scheduling algorithms a) FCFS b) SCAN c) C-SCAN

CODE-

```
Write a program to simulate page replacement algorithms
a) FIFO
b) LRU
c) Optimal

Code:-

#include <stdio.h>
int n, m, i, j, k, p, q, fault = 0;

void input() {
    printf("\nEnter length of page reference sequence:");
    scanf("%d", &n);
    printf("\nEnter the page reference sequence:");
    for (i = 0; i < n; i++)
        scanf("%d", &arr[i]);
    printf("\nEnter no of frames:");
    scanf("%d", &m);
}

void simulate_fifo() {
    int fault = 0;
    for (i = 0; i < n; i++)
        if (arr[i] != -1)
            fault++;
    printf("\nFault count: %d", fault);
}
```

```

    ind = 1;
    break;
}
return ind; }

```

und getaktieren (auch delete) f

```

    und inding;
    for (k=0; k<nf; k++)
    {
        if (p[k] == delete)
        {
            inding = k;
            break;
        }
    }
    return inding;
}

```

void display ()

```

{
    for (k=0; k<nf; k++)
    {
        if (p[k] != 9999)
            printf("%d ", p[k]);
    }
}

```

void displayFault ()

```

{
    printf("In total no. of page fault is: %d, (pfaults)");
}

```

void fgo (2

{

unshelg(c);

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

{

printf("In fgo - i.d : ", ans[i]);

if (ashut(ans[i]) == 0)

{

for (k=0; k<nf-n; k++)

ps[k] = ps[k+1];

ps[k] = ans[i];

psfaulter++;

dispgsnc();

}

else

printf("No pgz faulter");

dispgfaulter();

}

void optmnd(c)

{

unshelg(c);

unk nca(ssa);

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

{

printf("In fgo - i.d : ", ans[i]);

if (ashut(ans[i]) == 0)

{

for (j=0; j<nf-j; j++)

unk ps = ps[j];

```
unt found = 0;
```

```
for (r = 0; r < n; r++)
```

```
{  
    if (s[r] == '\n')
```

```
    {  
        near[r] = r;
```

```
        found = 1;
```

```
        break;
```

```
    }
```

```
    else
```

```
        found = 0;
```

```
}
```

```
if (!found)
```

```
    near[r] = 9999;
```

```
}
```

```
unt max = -9999;
```

```
unt repindex;
```

```
for (y = 0; y < n; y++)
```

```
{
```

```
    if (near[y] > max)
```

```
    {
```

```
        max = near[y];
```

```
        repindex = y;
```

```
    }
```

```
p[repindex] = '\n';
```

```
p[faulter]++;
```

```
clearPage();
```

```
}
```

```
else
```

```
    printf("No page fault");
```

```
}
```


displayFaultEnter();

void main()

{

initialize();

and clear();

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

{

printf("Enter id: ", arr[i]);

if (isHit(arr[i]) == 0)

{

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

if (least[j] < min)

{ min = least[j];

index = j;

}

}

p[index] = arr[i];

offset++;

display();

}

else

printf("No page fault!");

}

displayFaultEnter();

int main()

{

int choice;

while(1)

{

printf("\n Page Replacement Algorithms\n 1. FNRs algo
2. FIFO\n 3. Optimal\n 4. LRU\n 5. Exit\n Enter your choice:");

scanf("%d", &choice);

switch(choice)

{

case 1: printf("FNRs algo\n");
break;

case 2: printf("FIFO\n");
break;

case 3: printf("Optimal\n");
break;

case 4: printf("LRU\n");
break;

}

}

}

Output

Page Replacement Algorithms

1. FNRs algo

2. FIFO

3. Optimal

4. LRU

5. Exit

Write a C Program to simulate disk scheduling algorithms.

a) FCFS

b) SCAN

c) C-SCAN

Program

a) FCFS

#include <stdio.h>

#include <stdlib.h>

int main()

{

int RBT[100], i, n, TotalHeadMovement = 0, unvisited;

printf("Enter no of requests: ");

scanf("%d", &n);

printf("Enter the Request sequence: ");

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

printf("Enter unvisited head position: ");

scanf("%d", &unvisited);

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

{

TotalHeadMovement = TotalHeadMovement + abs(RBT[i] - unvisited);

unvisited = RBT[i];

}

printf("Total head movement is: %d", TotalHeadMovement);

return 0;

}

void insert()

for ($j = 0, j < n, j++$)

{

if (united == POS)

{

index = 1;

break;

}

if (move == 0)

{

for ($j = index, j < n, j++$)

{

TotalHeadMovement = TotalHeadMovement + $abs(POS - united)$;

united = POS;

}

TotalHeadMovement = TotalHeadMovement + $abs(POS - POS - j - 1)$

united = key - 1;

for ($j = index - 1, j >= 0, j--$)

{

TotalHeadMovement = TotalHeadMovement + $abs(POS - united)$

united = POS;

}

else

{

for ($j = index + 1, j < n, j++$)

{

TotalHeadMovement = TotalHeadMovement + $abs(POS - united)$

united = POS;

}

TotalHeadMovement = TotalHeadMovement + abs(R[j+1] - 0);
 unikal = 0;

for (i = unikal; i < n; i++)

{

TotalHeadMovement = TotalHeadMovement + abs(R[i] - unikal);

unikal = R[i];

}

printf("Total head movement is: %d", TotalHeadMovement);

return 0;

}

in C SCAN.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int R[100], i, j, n, TotalHeadMovement = 0, unikal, size, mono;

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

scanf("%d", &n);

void CSCAN()

{

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

{

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

{

if (R[j] > R[j+1])

{

int temp;

temp = R[j];

$RB[j] = RB[j+1]$

$RB[j+1] = temp$

}

unt under;

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

{

if ($under < RB[i]$)

{

$under = i$

break;

}

if ($move == 1$)

{

for ($i = under$; $i < n$; $i++$)

{

$TotalHeadMovement = TotalHeadMovement + abs(RB[i] - under)$

$under = RB[i]$

}

else

{

for ($i = under - 1$; $i > 0$; $i--$)

{

$TotalHeadMovement = TotalHeadMovement + abs(RB[i] - under)$

$under = RB[i]$

}

print C^u Total head movement for CSAN as $\cdot \cdot d$, Total Head Round)

Output

7. FCFS

Enter number of requests

3

Enter initial head position

20

Enter the Request Sequence

35 12 37 22

Total head movement is 118

8) ~~FCFS~~ SCAN & CSAN

Enter number of Requests 5

Enter the request sequence.

35 12 37 22 60

Enter initial head position

20

Enter total disk size

100

Enter the head movement direction for high 1 and for low 0

1

Total head movement for SCAN is 93

Total head movement for CSAN is 17

10/10

25/8/23

OUTPUT-

Page replacement algorithm

Page Replacement Algorithms

1.Enter data

2.FIFO

3.Optimal

4.LRU

5.Exit

Enter your choice:1

Enter length of page reference sequence:13

Enter the page reference sequence:1 2 3 4 5 2 1 6 7 8 7 8 9

Enter no of frames:3

Page Replacement Algorithms

1.Enter data

2.FIFO

3.Optimal

4.LRU

5.Exit

Enter your choice:2

For 1 : 1

For 2 : 1 2

For 3 : 1 2 3

For 4 : 2 3 4

For 5 : 3 4 5

For 2 : 4 5 2

For 1 : 5 2 1

For 6 : 2 1 6

For 7 : 1 6 7

```
For 7 :No page fault
For 8 :No page fault
For 9 : 7 8 9
Total no of page faults:11
Page Replacement Algorithms
1.Enter data
2.FIFO
3.Optimal
4.LRU
5.Exit
Enter your choice:3
```

```
For 1 : 1
For 2 : 1 2
For 3 : 1 2 3
For 4 : 1 2 4
For 5 : 1 2 5
For 2 :No page fault
For 1 :No page fault
For 6 : 6 2 5
For 7 : 7 2 5
For 8 : 7 8 5
For 7 :No page fault
For 8 :No page fault
For 9 : 9 8 5
Total no of page faults:9
Page Replacement Algorithms
1.Enter data
2.FIFO
3.Optimal
4.LRU
5.Exit
```

```
Enter your choice:4
```

```
For 1 : 1
For 2 : 1 2
For 3 : 1 2 3
For 4 : 4 2 3
For 5 : 4 5 3
For 2 : 4 5 2
For 1 : 1 5 2
For 6 : 1 6 2
For 7 : 1 6 7
For 8 : 8 6 7
For 7 :No page fault!
For 8 :No page fault!
For 9 : 8 9 7
Total no of page faults:11
```

Disk Scheduling

```
Enter the number of Requests
4
Enter the Requests sequence
55 12 37 22
Enter initial head position
20
Total head moment of FCFS is 118
```

```
Enter the number of Requests
5
Enter the Requests sequence
55 12 37 22 40
Enter initial head position
20
Enter total disk size
100
Enter the head movement direction for high 1 and for low 0
1
Total head movement of SCAN is 166
Process returned 0 (0x0) execution time : 18.170 s
Press any key to continue.
```

```
Enter the number of Requests
5
Enter the Requests sequence
55 12 37 22 40
Enter initial head position
20
Enter total disk size
100
Enter the head movement direction for high 1 and for low 0
1
Total head movement CSCAN is 190
Process returned 0 (0x0) execution time : 14.896 s
Press any key to continue.
```