**Name**: Akshat Rajesh Andhale **Class**: TEIT

**Roll No.:** 03

**Assignment no: 06** Implement C program for page replacement algorithms for page replacement algorithms: FCFS, LRU, and Optimal for frame size as minimum three.

1. **FCFS:**

**CODE:**

#include<stdio.h>

int present(int table\_frame[], int nf, int page)

{

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

if(page == table\_frame[i])

return 1;

return 0;

}

void printtable(int table\_frame[], int nf)

{

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

{

if(table\_frame[i] == -1)

printf("-- ");

else

printf("%2d ", table\_frame[i]);

}

printf("||");

}

int main()

{

//nf-number of frames

int n,nf,i,pos=0;

printf("enter number of frames\n");

scanf("%d",&nf);

int table\_frame[nf];

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

{

table\_frame[i]=-1;

}

printf("enter total number of page requests\n");

scanf("%d",&n);

int pages[n];

printf("enter reference string\n");

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

{

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

}

int count1=0;

printf("position of frame table after each request\n");

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

{

printf("page table after request from %2d || ",pages[i]);

if(!present(table\_frame,nf,pages[i]))

{

table\_frame[pos] = pages[i];

pos = (pos+1)%nf ;//considering it as a queue

printtable(table\_frame,nf);

printf("page fault\n");

count1++;

continue;

}

printtable(table\_frame,nf);

printf("\n");

}

printf("\nNumber of page faults : %d\n\n", count1);

}

**OUTPUT:**

enter number of frames

4

enter total number of page requests

5

enter reference string

33 12 11 24 10

position of frame table after each request

page table after request from 33 || 33 -- -- -- ||page fault

page table after request from 12 || 33 12 -- -- ||page fault

page table after request from 11 || 33 12 11 -- ||page fault

page table after request from 24 || 33 12 11 24 ||page fault

page table after request from 10 || 10 12 11 24 ||page fault

Number of page faults : 5

1. **LRU:**

**CODE:**

#include<stdio.h>

int present(int table\_frame[], int nf, int page)

{

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

if(page == table\_frame[i])

return 1;

return 0;

}

void printtable(int table\_frame[], int nf)

{

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

{

if(table\_frame[i] == -1)

printf("-- ");

else

printf("%2d ", table\_frame[i]);

}

printf("||");

}

int findpos(int table\_frame[], int nf, int pages[], int curr, int np)

{

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

if(table\_frame[i] == -1)

return i;

int pos[nf];

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

{

pos[i] = -1e9;

for(int j=curr-1; j>=0; j--)

if(pages[j] == table\_frame[i])

{

pos[i] = j;

break;

}

}

int min1 = 1000000, retPos = -1;

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

if(min1 > pos[i])

{

min1 = pos[i];

retPos = i;

}

return retPos;

}

int main()

{

//nf-number of frames

int n,nf,i,pos=0;

printf("enter number of frames\n");

scanf("%d",&nf);

int table\_frame[nf];

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

{

table\_frame[i]=-1;

}

printf("enter total number of page requests\n");

scanf("%d",&n);

int pages[n];

printf("enter pages\n");

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

{

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

}

int count1=0;

printf("position of frame table after each request\n");

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

{

printf("page table after request from %2d || ",pages[i]);

if(!present(table\_frame,nf,pages[i]))

{

int pos = findpos(table\_frame,nf,pages,i,n);

table\_frame[pos]=pages[i];

printtable(table\_frame,nf);

printf("page fault\n");

count1++;

continue;

}

printtable(table\_frame,nf);

printf("\n");

}

printf("\nNumber of page faults : %d\n\n", count1);

}

**OUTPUT**:

enter number of frames

4

enter total number of page requests

5

enter pages

12 23 17 22 10

position of frame table after each request

page table after request from 12 || 12 -- -- -- ||page fault

page table after request from 23 || 12 23 -- -- ||page fault

page table after request from 17 || 12 23 17 -- ||page fault

page table after request from 22 || 12 23 17 22 ||page fault

page table after request from 10 || 10 23 17 22 ||page fault

Number of page faults : 5

1. **Optimal for frame size as minimum three:**

**CODE:**

#include<stdio.h>

int present(int table\_frame[], int nf, int page){

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

if(page == table\_frame[i])

return 1;

return 0;

}

void printtable(int table\_frame[], int nf)

{

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

{

if(table\_frame[i] == -1)

printf("-- ");

else

printf("%2d ", table\_frame[i]);

}

printf("||");

}

int findpos(int table\_frame[],int nf,int pages[],int curr,int np)

{

int i,j;

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

{

if(table\_frame[i] == -1)

return i;

}

int pos[nf];

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

{

pos[i]=1e9;

for(j=curr+1;j<np;j++)

{

if(pages[j]==table\_frame[i])

{

pos[i]=j;

break;

}

}

}

int max1=-1;

int returnpos=-1;

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

if(pos[i]>max1)

{

max1=pos[i];

returnpos=i;

}

}

return returnpos;

}

int main()

{

//nf-number of frames

int n,nf,i,pos=0;

printf("enter number of frames\n");

scanf("%d",&nf);

int table\_frame[nf];

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

{

table\_frame[i]=-1;

}

printf("enter total number of page requests\n");

scanf("%d",&n);

int pages[n];

printf("enter pages\n");

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

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

}

int count1=0;

printf("position of frame table after each request\n");

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

printf("page table after request from %2d || ",pages[i]);

if(!present(table\_frame,nf,pages[i])){

int pos = findpos(table\_frame,nf,pages,i,n);

table\_frame[pos]=pages[i];

printtable(table\_frame,nf);

printf("page fault\n");

count1++;

continue;

}

printtable(table\_frame,nf);

printf("\n");

}

printf("\nNumber of page faults : %d\n\n", count1);

}

**OUTPUT:**

enter number of frames

5

enter total number of page requests

4

enter pages

12 23 17 34 28

position of frame table after each request

page table after request from 12 || 12 -- -- -- -- ||page fault

page table after request from 23 || 12 23 -- -- -- ||page fault

page table after request from 17 || 12 23 17 -- -- ||page fault

page table after request from 34 || 12 23 17 34 -- ||page fault

Number of page faults : 4