1. **PROGRAM FOR FIF0 PAGE REPLACEMENT**

#include <stdio.h>

void main() {

int a[] = {4,1,2,4,5}, f[3] = {-1,-1,-1}, i, j, k = 0, pf = 0;

printf("Page\tF1\tF2\tF3\n");

for (i = 0; i < 5; i++) {

int hit = 0;

for (j = 0; j < 3; j++) if (f[j] == a[i]) hit = 1;

if (!hit) { f[k] = a[i]; k = (k + 1) % 3; pf++; }

printf("%d\t", a[i]);

for (j = 0; j < 3; j++) ( f[j] == -1 ) ? printf("-\t") : printf("%d\t", f[j]);

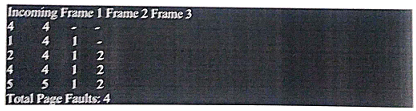
printf("\n");

}

printf("Total Page Faults: %d\n", pf);

}

**Output:**



1. **PROGRAM FOR LRU PAGE REPLACEMENT**

#include <stdio.h>

#include <limits.h>

int checkHit(int p, int q[], int occ) {

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

if (p == q[i]) return 1;

return 0;

}

void printFrame(int q[], int occ) {

for (int i = 0; i < 3; i++)

i < occ ? printf("%d\t", q[i]) : printf("-\t");

printf("\n");

}

void main() {

int in[] = {1,2,3,2,1,5,2,1,6,2,5,6,3,1,3}, q[3], d[3], occ = 0, pf = 0;

int n = sizeof(in)/sizeof(in[0]);

printf("Page\tF1\tF2\tF3\n");

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

printf("%d\t", in[i]);

if (checkHit(in[i], q, occ)) { printFrame(q, occ); continue; }

if (occ < 3) {

q[occ++] = in[i]; pf++;

printFrame(q, occ); continue;

}

for (int j = 0; j < 3; j++) {

d[j] = 0;

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

d[j]++;

if (q[j] == in[k]) break;

}

}

int max = 0;

for (int j = 1; j < 3; j++)

if (d[j] > d[max]) max = j;

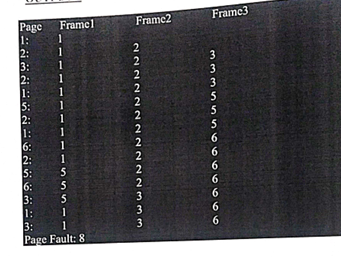
q[max] = in[i]; pf++;

printFrame(q, occ);

} printf("Page Faults: %d\n", pf);

}

**Output:**

****

1. **PROGRAM FOR OPTIMAL PAGE REPLACEMENT**

#include <stdio.h>

void main() {

int in[] = {7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1};

int f[3], t[3], n = 20, faults = 0, i, j, k, pos, max;

for (i = 0; i < 3; i++) f[i] = -1;

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

int hit = 0;

for (j = 0; j < 3; j++)

if (f[j] == in[i]) { hit = 1; break; }

if (!hit) {

int empty = -1;

for (j = 0; j < 3; j++)

if (f[j] == -1) { f[j] = in[i]; faults++; empty = 1; break; }

if (!empty) {

for (j = 0; j < 3; j++) {

t[j] = -1;

for (k = i + 1; k < n; k++)

if (f[j] == in[k]) { t[j] = k; break; }

}

for (j = 0; j < 3; j++)

if (t[j] == -1) { pos = j; break; }

else {

max = t[0]; pos = 0;

for (j = 1; j < 3; j++)

if (t[j] > max) { max = t[j]; pos = j; }

break;

}

f[pos] = in[i]; faults++;

}

}

for (j = 0; j < 3; j++) f[j] == -1 ? printf("-\t") : printf("%d\t", f[j]);

printf("\n");

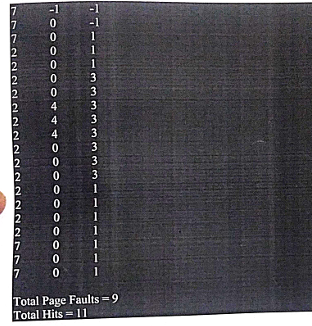
}

printf("\nTotal Page Faults = %d", faults);

printf("\nTotal Hits = %d\n", n - faults);

}

**Output:**



1. **PROGRAM OF FCFS SHEDULING:**

#include <stdio.h>

#include <conio.h>

void main() {

int i, sum = 0, n, disk;

int ar[20], tm[20];

printf("Enter number of requests:\t");

scanf("%d", &n);

printf("Enter initial head position:\t");

scanf("%d", &disk);

printf("Enter the request sequence :\n");

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

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

}

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

tm[i] = disk - ar[i];

if (tm[i] < 0) tm[i] = -tm[i];

sum += tm[i];

disk = ar[i];

}

printf("\nMovement of cylinders = %d", sum);

getch();

}

**Output:**

Enter number of requests: **8**

Enter initial head position: **53**

Enter the request sequence**: 95 180 34 119 11 123 62 64**

Movement of total cylinders : **641**

1. **PROGRAM OF SCAN SCHEDULING**

#include<stdio.h>

#include<stdlib.h>

void main() {

int RQ[100],i,j,n,TotalHeadMoment=0,initial,size,move;

printf("Enter the number of Requests\n");

scanf("%d",&n);

printf("Enter the Requests sequence\n");

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

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

printf("Enter initial head position\n");

scanf("%d",&initial);

printf("Enter total disk size\n");

scanf("%d",&size);

printf("Enter the head movement direction for high 1 and for low 0\n");

scanf("%d",&move);

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

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

{

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

{

int temp;

temp = RQ[j];

RQ[j] = RQ[j+1];

RQ[j+1] = temp;

} } }

int index;

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

if(initial<RQ[i]) {

index=i;

break;

} }

if(move==1){

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

TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);

initial=RQ[i];

}

TotalHeadMoment=TotalHeadMoment+abs(size-RQ[i-1]-1);

initial=size-1;

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

TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);

initial=RQ[i];

} }

else{

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

TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);

initial=RQ[i];

}

TotalHeadMoment=TotalHeadMoment+abs(RQ[i+1]-0);

initial=0;

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

TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);

initial=RQ[i];

} }

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

}

**Output:**

Enter number of requests: **8**

Enter the request sequence: **95 180 34 119 11 123 62 64**

Enter initial head position: **50**

Enter total disk size: **200**

Enter movement direction for high 1 and for low 0: **1**

Total head movement is: **337**

1. **PROGRAM OF Circular SCAN (C-SCAN)**

#include <stdio.h>

#include <stdlib.h>

void main() {

int RQ[] = {95,180,34,119,11,123,62,64}, n = 8;

int head = 50, size = 200, move = 1, total = 0;

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

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

if (RQ[j] > RQ[j+1]) {

int t = RQ[j]; RQ[j] = RQ[j+1]; RQ[j+1] = t;

}

int i, idx;

for (i = 0; i < n; i++) if (head < RQ[i]) { idx = i; break; }

if (move) {

for (i = idx; i < n; i++) total += abs(RQ[i] - head), head = RQ[i];

total += abs(size - head - 1) + (size - 1);

head = 0;

for (i = 0; i < idx; i++) total += abs(RQ[i] - head), head = RQ[i];

} else {

for (i = idx-1; i >= 0; i--) total += abs(RQ[i] - head), head = RQ[i];

total += abs(head - 0) + (size - 1);

head = size - 1;

for (i = n-1; i >= idx; i--) total += abs(RQ[i] - head), head = RQ[i];

}

printf("Total head movement: %d\n", total);

}

**Output:**

Total head movement: 382