Name : Sarthak Pagar

Roll No. : 34

Classs : TE (IT)

Practical : 6

Statement : [Implement the C program for Page Replacement Algorithms: FCFS, LRU, andOptimal for frame size as minimum three.](https://drive.google.com/file/d/1iB37o35i4KBF8faB9AqC66TNq8cHN3UV/view?usp=sharing)

#include<stdio.h>

typedef struct {

char data[20][2];

int end;

}queue;

void enqueue(queue \*q,char data,int position);

char dequeue(queue \*q,int position);

void fifo(char string[],int frameSize,int count);

void optimal(char string[],int frameSize,int count);

void lru(char string[],int frameSize,int count);

void main() {

int frameSize,count,cnt,ch;

char string[50];

printf("Enter the string: ");

count=0;

do{

scanf("%c",&string[count]);

count++;

}

while(string[count-1]!='\n');

count--;

printf("\nEnter the size of the frame: ");

scanf("%d",&frameSize);

do{

printf("\nMENU\n====\n1.FIFO\n2.Least Recently Used (LRU)\n3.Optimal\n4.Exit\n\nYour Choice:");

scanf("%d",&ch);

switch(ch){

case 1:

fifo(string,frameSize,count);

break;

case 2:

lru(string,frameSize,count);

break;

case 3:

optimal(string,frameSize,count);

break;

case 4:

//exit(0);

break;

default:

printf("\nInvalid choice! Please try again!");

continue;

}

}

while(ch!=4);

}

void enqueue(queue \*q,char data,int position) {

q->data[position][0]=data;

}

char dequeue(queue \*q,int position){

char value;

value=q->data[position][0];

return(value);

}

void fifo(char string[],int frameSize,int count) {

int cnt,cnt2,flag,faults=0;

queue q;

int firstin=-1;

q.end=0;

printf("\nData Requested\tFrame contents\t Page Fault\n==============================================");

for(cnt=0;cnt<count;cnt+=1) {

printf("\n\n\t%c",string[cnt]);

flag=0;

for(cnt2=0;cnt2<q.end;cnt2++) {

if(string[cnt]==q.data[cnt2][0]){

flag=1;

break;

}

}

if(flag==0){

faults++;

if(q.end<frameSize) {

enqueue(&q,string[cnt],q.end);

q.end++;

}

else{

dequeue(&q,firstin);

firstin=(firstin+1)%(q.end);

enqueue(&q,string[cnt],firstin);

}

printf("\t ");

for(cnt2=0;cnt2<q.end;cnt2++) {

printf("%c ",q.data[cnt2][0]);

}

printf("\t\tY");

}

else{

printf("\t ");

for(cnt2=0;cnt2<q.end;cnt2++) {

printf("%c ",q.data[cnt2][0]);

}

printf("\t\tN");

}

}

printf("\n\n==============================================\n");

printf("\nTotal no. of Page Faults: %d\n\n",faults);

}

void optimal(char string[], int frameSize, int count) {

int cnt, cnt2, selector, flag, max, faults = 0;

int distance[20];

queue q;

q.end = 0;

printf("\nData Requested\tFrame contents\t Page Fault\n==============================================");

for (cnt = 0; cnt < count; cnt += 1) {

printf("\n\n\t%c", string[cnt]);

flag = 0;

// Check if the page is already in the frame

for (cnt2 = 0; cnt2 < q.end; cnt2++) {

if (string[cnt] == q.data[cnt2][0]) {

flag = 1;

break;

}

}

if (flag == 0) { // Page fault

faults++;

if (q.end < frameSize) { // If there is room in the frame

enqueue(&q, string[cnt], q.end);

q.end++;

} else { // If the frame is full, find the page to replace

// Calculate distances for each page in the frame

for (cnt2 = 0; cnt2 < q.end; cnt2++) {

distance[cnt2] = 99; // Initialize with a large value

for (int future = cnt + 1; future < count; future++) {

if (string[future] == q.data[cnt2][0]) {

distance[cnt2] = future; // When the page is next used

break;

}

}

}

// Find the page that is used the farthest in the future or not at all

max = 0;

selector = 0;

for (cnt2 = 0; cnt2 < q.end; cnt2++) {

if (distance[cnt2] > max) {

max = distance[cnt2];

selector = cnt2;

}

}

// Replace the page

dequeue(&q, selector);

enqueue(&q, string[cnt], selector);

}

// Print the frame contents

printf("\t ");

for (cnt2 = 0; cnt2 < q.end; cnt2++) {

printf("%c ", q.data[cnt2][0]);

}

printf("\t\tY");

} else { // No page fault

printf("\t ");

for (cnt2 = 0; cnt2 < q.end; cnt2++) {

printf("%c ", q.data[cnt2][0]);

}

printf("\t\tN");

}

}

printf("\n\n==============================================\n");

printf("\nTotal no. of Page Faults: %d\n\n", faults);

}

void lru(char string[], int frameSize, int count) {

int cnt, cnt2, selector, flag, faults = 0;

queue q;

q.end = 0;

int lastUsed[20] = {0}; // Array to keep track of last used times

printf("\nData Requested\tFrame contents\t Page Fault\n==============================================");

for(cnt = 0; cnt < count; cnt += 1) {

printf("\n\n\t%c", string[cnt]);

flag = 0;

// Check if the page is already in the frame

for(cnt2 = 0; cnt2 < q.end; cnt2++) {

if(string[cnt] == q.data[cnt2][0]) {

flag = 1;

break;

}

}

if(flag == 0) { // Page fault

faults++;

if(q.end < frameSize) {

enqueue(&q, string[cnt], q.end);

q.end++;

} else {

// Find the least recently used page

int min = 0;

for(cnt2 = 1; cnt2 < q.end; cnt2++) {

if(lastUsed[q.data[min][0] - '0'] > lastUsed[q.data[cnt2][0] - '0']) {

min = cnt2;

}

}

dequeue(&q, min);

enqueue(&q, string[cnt], min);

}

// Update last used time

lastUsed[string[cnt] - '0'] = cnt;

printf("\t ");

for(cnt2 = 0; cnt2 < q.end; cnt2++) {

printf("%c ", q.data[cnt2][0]);

}

printf("\t\tY");

} else { // No page fault

// Update last used time

lastUsed[string[cnt] - '0'] = cnt;

printf("\t ");

for(cnt2 = 0; cnt2 < q.end; cnt2++) {

printf("%c ", q.data[cnt2][0]);

}

printf("\t\tN");

}

}

printf("\n\n==============================================\n");

printf("\nTotal no. of Page Faults: %d\n\n", faults);

}

Output :-

[Saru1594@localhost 6]$ gcc pagereplacement.c

[Saru1594@localhost 6]$ ./a.out

Enter the string: 70120304230321201701

Enter the size of the frame: 3

MENU

====

1.FIFO

2.Least Recently Used (LRU)

3.Optimal

4.Exit

Your Choice:1

Data Requested Frame contents Page Fault

==============================================

7 7 Y

0 7 0 Y

1 7 0 1 Y

2 2 0 1 Y

0 2 0 1 N

3 2 3 1 Y

0 2 3 0 Y

4 4 3 0 Y

2 4 2 0 Y

3 4 2 3 Y

0 0 2 3 Y

3 0 2 3 N

2 0 2 3 N

1 0 1 3 Y

2 0 1 2 Y

0 0 1 2 N

1 0 1 2 N

7 7 1 2 Y

0 7 0 2 Y

1 7 0 1 Y

==============================================

Total no. of Page Faults: 15

MENU

====

1.FIFO

2.Least Recently Used (LRU)

3.Optimal

4.Exit

Your Choice:2

Data Requested Frame contents Page Fault

==============================================

7 7 Y

0 7 0 Y

1 7 0 1 Y

2 2 0 1 Y

0 2 0 1 N

3 2 0 3 Y

0 2 0 3 N

4 4 0 3 Y

2 4 0 2 Y

3 4 3 2 Y

0 0 3 2 Y

3 0 3 2 N

2 0 3 2 N

1 1 3 2 Y

2 1 3 2 N

0 1 0 2 Y

1 1 0 2 N

7 1 0 7 Y

0 1 0 7 N

1 1 0 7 N

==============================================

Total no. of Page Faults: 12

MENU

====

1.FIFO

2.Least Recently Used (LRU)

3.Optimal

4.Exit

Your Choice:3

Data Requested Frame contents Page Fault

==============================================

7 7 Y

0 7 0 Y

1 7 0 1 Y

2 2 0 1 Y

0 2 0 1 N

3 2 0 3 Y

0 2 0 3 N

4 2 4 3 Y

2 2 4 3 N

3 2 4 3 N

0 2 0 3 Y

3 2 0 3 N

2 2 0 3 N

1 2 0 1 Y

2 2 0 1 N

0 2 0 1 N

1 2 0 1 N

7 7 0 1 Y

0 7 0 1 N

1 7 0 1 N

==============================================

Total no. of Page Faults: 9

MENU

====

1.FIFO

2.Least Recently Used (LRU)

3.Optimal

4.Exit

Your Choice:4