**Page Replacement**

#include<iostream>

using namespace std;

int page\_str[30], n\_pages;

void fifo();

void lru();

void optimal();

int main() {

int ch, i;

char ans;

cout << "Total= 20\n\t\t7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1\n";

cout << "\n\t\tENTER NO PAGES IN REFERENCE STRING::->";

cin >> n\_pages;

cout << "\n\tENTER " << n\_pages << " PAGE NO.S::->";

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

cin >> page\_str[i];

do {

cout << "\n\t\tMENU\n";

cout << "\n\t\t1.FIFO PAGE REPLACEMENT";

cout << "\n\t\t2.LRU PAGE REPLACEMENT";

cout << "\n\t\t3.OPTIMAL";

cout << "\n\t\t4.EXIT";

cout << "\n\t\tENTER YOUR CHOICE::->";

cin >> ch;

switch (ch) {

case 1:

fifo();

break;

case 2:

lru();

break;

case 3:

optimal();

break;

}

cout << "\nDo u want to cont..";

cin >> ans;

} while (ans == 'y' || ans == 'Y');

return 0;

}

void fifo() {

int col = 0, size\_frame, flag, fault = 0, page[3] = { -1,-1,-1 };

int i = 0, j = 0, k = 0;

cout << "\nENTER SIZE OF PAGE FRAME:";

cin >> size\_frame;

cout << "\n Page String:\n";

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

cout << page\_str[i] << " ";

cout << "\nFIFO PAGE REPLACEMENT ALGO:\n\n";

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

for (j = 0; j < size\_frame; j++) {

flag = 1;

if (page[j] == page\_str[i]) {

flag = 0;

break;

}

}

if (flag == 1) {

fault++;

if (col >= size\_frame)

col = 0;

page[col++] = page\_str[i];

}

cout << "\n";

for (k = 0; k < size\_frame; k++) {

cout << page[k] << " ";

}

}

cout << "\nFault:-" << fault;

}

void lru() {

int flag1 = 0, col = 0, i, j, k, min = 0, cflag = 0, page[3] = { -1,-1,-1 };

int cnt[3] = { -1,-1,-1, }, fault = 0, size\_frame;

cout << "\nENTER SIZE OF PAGE FRAME:";

cin >> size\_frame;

cout << "\n Page Frame:\n";

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

cout << page\_str[i] << " ";

cout << "\nLRU PAGE REPLACEMENT ALGO:\n\n";

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

for (j = 0; j < size\_frame; j++) {

flag1 = 1;

if (page[j] == page\_str[i]) {

flag1 = 0;

cnt[j] = i;

break;

}

}

if (flag1 == 1) {

fault++;

if (col >= size\_frame) {

col = 0;

cflag = 1;

}

min = cnt[0];

for (j = 0; j < size\_frame; j++) {

if (min > cnt[j] && cnt[j] != -1) {

min = cnt[j];

col = j;

}

}

if (min == cnt[0] && cflag == 1) {

col = 0;

}

cnt[col] = i;

page[col++] = page\_str[i];

}

for (k = 0; k < size\_frame; k++) {

cout << " " << page[k] << " ";

}

cout << "\n\n";

}

cout << "\nNo of Faults :" << fault;

}

void optimal() {

int p[30], i, j, no\_of\_pages, size\_frame, max, found = 0, index, k, l, flag1 = 0, flag2 = 0, pf = 0;

int fr[10] = { -1,-1,-1 };

int lg[3], fs[3];

cout << "20\n7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1\n";

cout << "\nENTER NO PAGES IN REFERENCE STRING::->";

cin >> no\_of\_pages;

cout << "\nENTER " << no\_of\_pages << " PAGE NO.S::->";

for (i = 0; i < no\_of\_pages; i++)

cin >> p[i];

cout << "\nENTER SIZE OF PAGE FRAME:";

cin >> size\_frame;

cout << "\n";

for (j = 0; j < no\_of\_pages; j++) {

flag1 = 0;

flag2 = 0;

for (i = 0; i < size\_frame; i++) {

if (fr[i] == p[j]) {

flag1 = 1;

flag2 = 1;

break;

}

}

if (flag1 == 0) {

for (i = 0; i < size\_frame; i++) {

if (fr[i] == -1) {

fr[i] = p[j];

flag2 = 1;

pf++;

break;

}

}

}

if (flag2 == 0) {

for (i = 0; i < size\_frame; i++)

lg[i] = 0;

for (i = 0; i < size\_frame; i++) {

for (k = j + 1; k < no\_of\_pages; k++) {

if (fr[i] == p[k]) {

lg[i] = k - j;

break;

}

}

}

found = 0;

for (i = 0; i < size\_frame; i++) {

if (lg[i] == 0) {

index = i;

found = 1;

break;

}

}

if (found == 0) {

max = lg[0];

index = 0;

for (i = 1; i < size\_frame; i++) {

if (max < lg[i]) {

max = lg[i];

index = i;

}

}

}

fr[index] = p[j];

pf++;

}

for (k = 0; k < size\_frame; k++)

cout << " " << fr[k] << " ";

cout << "\n";

}

cout << "\nNO.OF PAGE FAULTS:- " << pf;

}



