#include<stdio.h> #include<stdlib.h> #include<string.h>

int referenceString[]={1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5};

int lengthOfReferenceString = sizeof(referenceString) / sizeof(int);

int pagePresentInFrames; // Flag to indicate presence of particular page in frame int pageFaults; // To count page faults

int i; // Index , for page in reference string int j;

int k; // Index , for frames

void printPagesInFrames( int frame[], int numberOfFrames )

{

// Function prints current pages in frames, -1 if no page in frame for( k=0; k < numberOfFrames; k++) printf("\t %d",frame[k] );

}

int findIndexOfLeastRecentlyUsed( int lruCounter[], int numberOfFrames )

{

// Function return index of Least Recently Used page int lruValue = -1;

int indexOfLRU = 0;

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

if( lruCounter[i] == -1 ) // Return index of first occurance of free frame return i;

else if( lruCounter[i] > lruValue ) // else find frame with highest counter

{

// hence least recently used lruValue = lruCounter[i]; indexOfLRU = i;

}

return indexOfLRU;

}

void fifoPageReplacement( int numberOfFrames )

{

int frame[5] = {-1, -1, -1, -1, -1};// To remember pages in frames, initialize as unallocated pageFaults = 0; // Initialize page faults = 0

j=0; // Which frame is going to be filled next with page

printf("\n FIFO Page replacement using %d frames, initial frames = ", numberOfFrames); printPagesInFrames( frame, numberOfFrames );

printf("\n Page in reference string\t\t Pages in Frames");

for( i=0; i<lengthOfReferenceString; i++ ) // For each page in reference string

{

printf("\n\t\t %d\t\t",referenceString[i]);//print page that will be allocated frame pagePresentInFrames=0; // Assume page is not present in frames

for(k=0; k < numberOfFrames; k++) // For each frame

if( frame[k] == referenceString[i] )// Check if page is present in frames pagePresentInFrames=1; // Page exists in frames

if ( pagePresentInFrames == 0 ) // If page was not present in frames

{

frame[j]=referenceString[i]; // allocate j th frame to page referenceString[i]

j=(j + 1) % numberOfFrames; // increment j, modulo division for circular queue pageFaults++; // Increment page faults

printPagesInFrames( frame, numberOfFrames );

}

}

printf("\n Page Faults are = %d\n", pageFaults);

}

void lruPageReplacement( int numberOfFrames )

{

// counter for least recently used , -1 for unused frame, 0 is recently used,

int lruCounter[5] = {-1, -1, -1, -1, -1};//highest value will be least recently used

int frame[5] = {-1, -1, -1, -1, -1};// To remember page in frame, initialized as unallocated pageFaults = 0; // Initialize page faults = 0

printf("\n LRU Page replacement using %d frames, initial frames = ", numberOfFrames); printPagesInFrames( frame, numberOfFrames );

printf("\n Page in reference string\t\t Pages Frames");

for( i=0; i<lengthOfReferenceString; i++ ) // For each page in reference string

{

printf("\n\t\t %d\t\t",referenceString[i]);//print page that will be allocated frame pagePresentInFrames=0; // Assume page is not present in frames

for(k=0; k < numberOfFrames; k++) // For each frame

if( frame[k] == referenceString[i] )// Check if page is present in frames

{

pagePresentInFrames=1; // Page exists in frames

lruCounter[k] = 0; // page used, hence reinitialize counter as recently used

}

else if( lruCounter[k] != -1 )

lruCounter[k]++; // It is different page, update recently used counter if ( pagePresentInFrames == 0 ) // If page was not present in frames

{

j = findIndexOfLeastRecentlyUsed( lruCounter, numberOfFrames ); frame[j]=referenceString[i]; // allocate j th frame to page referenceString[i] lruCounter[j] = 0; // initialize j th counter as recently used

pageFaults++; // Increment page faults printPagesInFrames( frame, numberOfFrames );

}

}

printf("\n Page Faults are = %d\n",pageFaults);

}

int main() // read the number of pages, frames and reference string from user

{

// print reference string printf("\n Reference string = ");

for( i=0; i < lengthOfReferenceString; i++) printf(" %d",referenceString[i]); fifoPageReplacement( 2 ); // Try with 3, 4 and 5 frames

lruPageReplacement( 2 ); // Try with 3, 4 and 5 frames

return 0; // Try 3, 2, 1, 0, 3, 2, 4, 3, 2, 1, 0, 4, 3

}/\* also 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0 , 1, 7, 0, 1 \*/