**Write a C program to simulate page replacement algorithms. (Any one) a) FIFO b) LRU c) Optimal**

#include <stdio.h>

#include <limits.h>

int isInFrame(int frames[], int f, int page) {

    for (int i = 0; i < f; i++) {

        if (frames[i] == page)

            return 1;

    }

    return 0;

}

int findLRU(int time[], int f) {

    int min = INT\_MAX, index = -1;

    for (int i = 0; i < f; i++) {

        if (time[i] < min) {

            min = time[i];

            index = i;

        }

    }

    return index;

}

int findOptimal(int pages[], int frames[], int n, int f, int index) {

    int farthest = index;

    int replaceIndex = -1;

    for (int i = 0; i < f; i++) {

        int j;

        for (j = index; j < n; j++) {

            if (frames[i] == pages[j]) {

                if (j > farthest) {

                    farthest = j;

                    replaceIndex = i;

                }

                break;

            }

        }

        if (j == n)

            return i;

    }

    return replaceIndex;

}

void printFrames(int frames[], int f) {

    for (int i = 0; i < f; i++) {

        if (frames[i] == -1)

            printf("- ");

        else

            printf("%d ", frames[i]);

    }

    printf("\n");

}

void fifo(int pages[], int n, int f) {

    int frames[f], faults = 0, index = 0;

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

    printf("\nFIFO Page Replacement:\n");

    for (int i = 0; i < n; i++) {

        if (!isInFrame(frames, f, pages[i])) {

            frames[index] = pages[i];

            index = (index + 1) % f;

            faults++;

        }

        printFrames(frames, f);

    }

    printf("FIFO Page Faults: %d\n", faults);

}

void lru(int pages[], int n, int f) {

    int frames[f], time[f], faults = 0, counter = 0;

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

    printf("\nLRU Page Replacement:\n");

    for (int i = 0; i < n; i++) {

        counter++;

        if (!isInFrame(frames, f, pages[i])) {

            int index = -1;

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

                if (frames[j] == -1) {

                    index = j;

                    break;

                }

            }

            if (index == -1)

                index = findLRU(time, f);

            frames[index] = pages[i];

            time[index] = counter;

            faults++;

        } else {

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

                if (frames[j] == pages[i])

                    time[j] = counter;

            }

        }

        printFrames(frames, f);

    }

    printf("LRU Page Faults: %d\n", faults);

}

void optimal(int pages[], int n, int f) {

    int frames[f], faults = 0;

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

    printf("\nOptimal Page Replacement:\n");

    for (int i = 0; i < n; i++) {

        if (!isInFrame(frames, f, pages[i])) {

            int index = -1;

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

                if (frames[j] == -1) {

                    index = j;

                    break;

                }

            }

            if (index == -1)

                index = findOptimal(pages, frames, n, f, i + 1);

            frames[index] = pages[i];

            faults++;

        }

        printFrames(frames, f);

    }

    printf("Optimal Page Faults: %d\n", faults);

}

 // Reference String: 4, 1, 0, 2, 4, 3

int main() {

    int n, f;

    printf("Enter number of frames: ");

    scanf("%d", &f);

    printf("Enter length of reference string: ");

    scanf("%d", &n);

    int pages[n];

    printf("Enter page reference string:\n");

    for (int i = 0; i < n; i++)

        scanf("%d", &pages[i]);

    fifo(pages, n, f);

    lru(pages, n, f);

    optimal(pages, n, f);

    return 0;

}

