LAB10

Write a C program to simulate page replacement algorithms

a) FIFO

b) LRU

c) Optimal

INPUT:

```c
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
int isPagePresent(int frames[], int frame_count, int page) {
    for (int i = 0; i < frame_count; i++) {
        if (frames[i] == page) {
            return 1;
        }
    }
    return 0;
}
void printFrames(int frames[], int frame_count) {
    for (int i = 0; i < frame_count; i++) {
        if (frames[i] == -1) {
            printf("- ");
        } else {
            printf("%d ", frames[i]);
        }
    }
    printf("\n");
}
void fifo(int pages[], int page_count, int frame_count) {
    int frames[frame_count];
    for (int i = 0; i < frame_count; i++) frames[i] = -1;
    int index = 0, page_faults = 0;
    printf("FIFO Page Replacement Process:\n");
    for (int i = 0; i < page_count; i++) {
        if (!isPagePresent(frames, frame_count, pages[i])) {
            frames[index] = pages[i];
            index = (index + 1) % frame_count;
            page_faults++;
            printFrames(frames, frame_count);
            printf("PF No. %d\n", page_faults);
        }
    }

    printf("The number of Page Faults using FIFO are %d\n\n", page_faults);
}
void lru(int pages[], int page_count, int frame_count) {
    int frames[frame_count];
    int counter[frame_count];
    for (int i = 0; i < frame_count; i++) {
        frames[i] = -1;
        counter[i] = 0;
    }

    int page_faults = 0, time = 0;
    printf("LRU Page Replacement Process:\n");
    for (int i = 0; i < page_count; i++) {
        int least_recently_used = time, lru_index = -1;
```

```
        for (int j = 0; j < frame_count; j++) {
            if (frames[j] == pages[i]) {
                counter[j] = time++;
                lru_index = -1;
                break;
            }
            if (frames[j] == -1) {
                lru_index = j;
                break;
            }
            if (counter[j] < least_recently_used) {
                least_recently_used = counter[j];
                lru_index = j;
            }
        }

        if (lru_index != -1) {
            frames[lru_index] = pages[i];
            counter[lru_index] = time++;
            page_faults++;
            printFrames(frames, frame_count);
            printf("PF No. %d\n", page_faults);
        }
    }

    printf("The number of Page Faults using LRU are %d\n\n", page_faults);
}
void optimal(int pages[], int page_count, int frame_count) {
    int frames[frame_count];
    for (int i = 0; i < frame_count; i++) frames[i] = -1;
    int page_faults = 0;
    printf("Optimal Page Replacement Process:\n");
    for (int i = 0; i < page_count; i++) {
        if (!isPagePresent(frames, frame_count, pages[i])) {
            int index = -1, farthest = i + 1;
            for (int j = 0; j < frame_count; j++) {
                int k;
                for (k = i + 1; k < page_count; k++) {
                    if (frames[j] == pages[k]) {
                        if (k > farthest) {
                            farthest = k;
                            index = j;
                        }
                        break;
                    }
                }
                if (k == page_count) {
                    index = j;
                    break;
                }
```

```
                }
            }
            if (index == -1) index = 0;

            frames[index] = pages[i];
            page_faults++;
            printFrames(frames, frame_count);
            printf("PF No. %d\n", page_faults);
        }
    }

    printf("The number of Page Faults using Optimal are %d\n\n", page_faults);
}

int main() {
    int page_count, frame_count;
    printf("Enter the number of pages: ");
    scanf("%d", &page_count);
    int pages[page_count];
    printf("Enter the page sequence: ");
    for (int i = 0; i < page_count; i++) {
        scanf("%d", &pages[i]);
    }

    printf("Enter the number of frames: ");
    scanf("%d", &frame_count);
    fifo(pages, page_count, frame_count);
    lru(pages, page_count, frame_count);
    optimal(pages, page_count, frame_count);
    return 0;
}
```

OUTPUT:

```
Enter the number of pages: 7
Enter the page sequence: 1
3
0
3
5
6
3
Enter the number of frames: 3
FIFO Page Replacement Process:
1 - -
PF No. 1
1 3 -
PF No. 2
1 3 0
PF No. 3
5 3 0
PF No. 4
5 6 0
PF No. 5
5 6 3
PF No. 6
The number of Page Faults using FIFO are 6

LRU Page Replacement Process:
1 - -
PF No. 1
1 3 -
PF No. 2
1 3 0
PF No. 3
5 3 0
PF No. 4
5 3 6
PF No. 5
The number of Page Faults using LRU are 5

Optimal Page Replacement Process:
1 - -
PF No. 1
3 - -
PF No. 2
3 0 -
PF No. 3
3 5 -
PF No. 4
3 6 -
PF No. 5
The number of Page Faults using Optimal are 5
```