```c
#include <stdio.h>

#define FRAME_SIZE 3
#define PAGE_COUNT 5

void fifo(int pages[], int n) {
    int frame[FRAME_SIZE] = {-1, -1, -1};
    int pageFaults = 0, index = 0;

    for (int i = 0; i < n; i++) {
        int found = 0;
        for (int j = 0; j < FRAME_SIZE; j++) {
            if (frame[j] == pages[i]) {
                found = 1;
                break;
            }
        }
        if (!found) {
            frame[index] = pages[i];
            index = (index + 1) % FRAME_SIZE;
            pageFaults++;
        }
    }
    printf("Total Page Faults: %d\n", pageFaults);
}

int main() {
    int pages[PAGE_COUNT] = {0, 1, 2, 0, 1};
    fifo(pages, PAGE_COUNT);
    return 0;
}
```

```
Total Page Faults: 3


=== Code Execution Successful ===
```

```c
#include <stdio.h>
#include <stdlib.h>

#define FRAME_SIZE 3

void lru(int pages[], int n) {
    int frame[FRAME_SIZE], counter[FRAME_SIZE] = {0}, pageFaults = 0;
    for (int i = 0; i < FRAME_SIZE; i++) frame[i] = -1;

    for (int i = 0; i < n; i++) {
        int j, found = 0;
        for (j = 0; j < FRAME_SIZE; j++) {
            if (frame[j] == pages[i]) {
                found = 1;
                counter[j] = i; // Update the counter
                break;
            }
        }
        if (!found) {
            int lruIndex = 0;
            for (j = 1; j < FRAME_SIZE; j++)
                if (counter[j] < counter[lruIndex]) lruIndex = j;
            frame[lruIndex] = pages[i];
            counter[lruIndex] = i; // Update the counter
            pageFaults++;
        }
    }
    printf("Total Page Faults: %d\n", pageFaults);
}

int main() {
    int pages[] = {7, 0, 1, 2, 0, 3, 0, 4, 2, 3};
    int n = sizeof(pages) / sizeof(pages[0]);
    lru(pages, n);
    return 0;
}
```

```
Total Page Faults: 8


=== Code Execution Successful ===
```