```c
#include <stdio.h>
#define FRAME_SIZE 3
#define PAGE_SIZE 10

void optimalPageReplacement(int pages[], int n) {
    int frames[FRAME_SIZE] = {-1, -1, -1};
    int pageFaults = 0;

    for (int i = 0; i < n; i++) {
        int j, k, flag = 1;
        for (j = 0; j < FRAME_SIZE; j++) {
            if (frames[j] == pages[i]) {
                flag = 0; // Page already in frame
                break;
            }
        }
        if (flag) {
            int l = -1, farthest = -1;
            for (j = 0; j < FRAME_SIZE; j++) {
                for (k = i + 1; k < n; k++) {
                    if (frames[j] == pages[k]) {
                        if (k > farthest) {
                            farthest = k;
                            l = j;
                        }
                        break;
                    }
                }
                if (k == n) {
                    l = j; // Not found in future
                    break;}}
  frames[l] = pages[i]; // Replace page
            pageFaults++;}}

    printf("Total Page Faults: %d\n", pageFaults);}
int main() {
    int pages[PAGE_SIZE] = {0, 1, 2, 0, 3, 0, 4, 2, 3, 0};
    optimalPageReplacement(pages, PAGE_SIZE);
    return 0;
}
```

```
Total Page Faults: 6


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

```c
#include <stdio.h>

#define MAX_RECORDS 100

void readRecords(char records[MAX_RECORDS][50], int count) {
    for (int i = 0; i < count; i++) {
        printf("Record %d: %s\n", i + 1, records[i]);
    }
}

int main() {
    char records[MAX_RECORDS][50] = {
        "Record 1: Data A",
        "Record 2: Data B",
        "Record 3: Data C"
    };
    int count = 3;

    readRecords(records, count);
    return 0;}
```

```
Record 1: Record 1: Data A
Record 2: Record 2: Data B
Record 3: Record 3: Data C


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