

32. Construct a C program to simulate the Least Recently Used paging technique of memory management.

```
#include <stdio.h>

int findLRU(int time[], int n) {
    int i, minimum = time[0], pos = 0;
    for (i = 1; i < n; ++i) {
        if (time[i] < minimum) {
            minimum = time[i];
            pos = i;
        }
    }
    return pos;
}

int main() {
    int no_of_frames, no_of_pages, frames[10], pages[30];
    int counter = 0, time[10], flag1, flag2, i, j, pos, faults = 0;
    printf("Enter number of frames: ");
    scanf("%d", &no_of_frames);
    printf("Enter number of pages: ");
    scanf("%d", &no_of_pages);
    printf("Enter the page reference string:\n");
    for (i = 0; i < no_of_pages; ++i)
        scanf("%d", &pages[i]);
    for (i = 0; i < no_of_frames; ++i)
        frames[i] = -1;
    printf("\nPage Reference | Frame Content | Page Fault\n");
    printf("-----\n");
    for (i = 0; i < no_of_pages; ++i) {
        flag1 = flag2 = 0;
```

```
// Check if page already exists in frame
```

```
for (j = 0; j < no_of_frames; ++j) {
```

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

```
        counter++;
```

```
        time[j] = counter;
```

```
        flag1 = flag2 = 1;
```

```
        break;
```

```
    }
```

```
}
```

```
// If page not found in frame
```

```
if (flag1 == 0) {
```

```
    for (j = 0; j < no_of_frames; ++j) {
```

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

```
            counter++;
```

```
            faults++;
```

```
            frames[j] = pages[i];
```

```
            time[j] = counter;
```

```
            flag2 = 1;
```

```
            break;
```

```
        }
```

```
    }
```

```
}
```

```
// Replace least recently used page
```

```
if (flag2 == 0) {
```

```
    pos = findLRU(time, no_of_frames);
```

```
    counter++;
```

```
    faults++;
```

```
    frames[pos] = pages[i];
```

```
    time[pos] = counter;
```

```
}
```

```

printf("%10d | ", pages[i]);

for (j = 0; j < no_of_frames; ++j) {
    if (frames[j] != -1)
        printf("%d ", frames[j]);

    else
        printf("- ");
}

if (flag1 == 0)
    printf(" | Yes\n");

else
    printf(" | No\n");
}

printf("\nTotal Page Faults = %d\n", faults);

printf("Page Fault Ratio = %.2f\n", (float)faults / no_of_pages);

return 0;
}

```

## OUTPUT:

```

#include <stdio.h>
int findLRU(int time[], int n) {
    int i, minimum = time[0], pos = 0;
    for (i = 1; i < n; ++i) {
        if (time[i] < minimum) {
            minimum = time[i];
            pos = i;
        }
    }
    return pos;
}

int main() {
    int no_of_frames, no_of_pages, frames[10], pages[30];
    int counter = 0, time[10], flag1, flag2, i, j, pos, faults = 0;
    printf("Enter number of frames: ");
    scanf("%d", &no_of_frames);
    printf("Enter number of pages: ");
    scanf("%d", &no_of_pages);
    printf("Enter the page reference string:\n");
    for (i = 0; i < no_of_pages; ++i)
        scanf("%d", &pages[i]);
    for (i = 0; i < no_of_frames; ++i)
        frames[i] = -1;
    printf("\nPage Reference | Frame Content | Page Fault\n");

```

Enter number of frames: 4  
 Enter number of pages: 5  
 Enter the page reference string:  
 1 2 3 5 2

Page Reference	Frame Content	Page Fault
1	1 - - -	Yes
2	1 2 - -	Yes
3	1 2 3 -	Yes
5	1 2 3 5	Yes
2	1 2 3 5	No

Total Page Faults = 4  
 Page Fault Ratio = 0.80

-----  
 Process exited after 13.9 seconds with return