Ex no:11 c)	OPTIMAL

Aim:

To write a c program to implement Optimal page replacement

Algorithm:

1.Start the process

2.Declare the size

3.Get the number of pages to be inserted

4.Get the value

5.Declare counter and stack

6.Select the least frequently used page by counter

value 7.Stack them according the selection.

8.Display the values

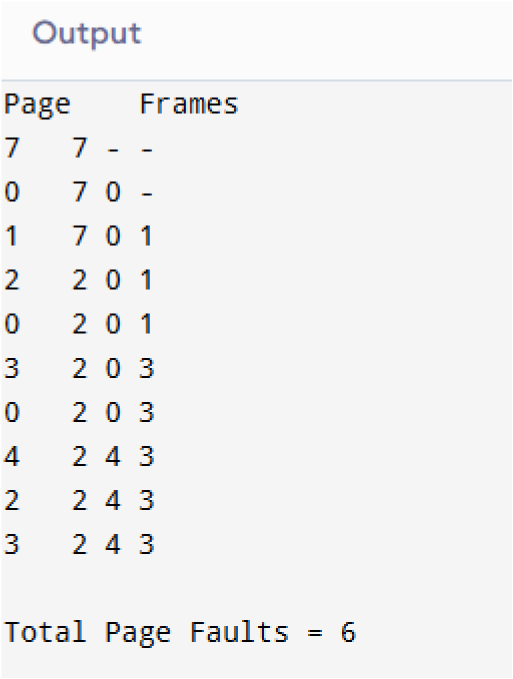
9.Stop the process

Program:

```
#include <stdio.h>

int main() {
   int pages[] = {7, 0, 1, 2, 0, 3, 0, 4, 2, 3};  // Reference String
   int n = 10;                              // Number of pages
   int frames = 3;                            // Number of frames
   int frame[10];
   int pageFaults = 0;

   // Initialize all frames as empty (-1)
   for (int i = 0; i < frames; i++) {
      frame[i] = -1;
   }
```

```
printf("Page\tFrames\n");

for (int i = 0; i < n; i++) {
    int found = 0;

    // Check if page is already in frame
    for (int j = 0; j < frames; j++) {
        if (frame[j] == pages[i]) {
            found = 1;
            break;
        }
    }

    // If page not found - Page Fault
    if (found == 0) {
        pageFaults++;

        // Check for empty frame
        int empty = -1;
        for (int j = 0; j < frames; j++) {
            if (frame[j] == -1) {
                empty = j;
                break;
            }
        }

        // If empty frame available
        if (empty != -1) {
            frame[empty] = pages[i];
        } else {
            int farthest = -1, pos = -1;

            // Find the page not used for longest time
            for (int j = 0; j < frames; j++) {
                int k;
                for (k = i + 1; k < n; k++) {
                    if (frame[j] == pages[k])
                        break;
                }

                if (k > farthest) {
                    farthest = k;
                    pos = j;
                }
        }

        frame[pos] = pages[i];  // Replace page
      }
    }

    // Display current frame status
    printf("%d\t", pages[i]);
    for (int j = 0; j < frames; j++) {
      if (frame[j] != -1)
        printf("%d ", frame[j]);
      else
        printf("- ");
    }
    printf("\n");
  }

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

  return 0;
}
```

Output:



Output

```
Page    Frames
7   7 - -
0   7 0 -
1   7 0 1
2   2 0 1
0   2 0 1
3   2 0 3
0   2 0 3
4   2 4 3
2   2 4 3
3   2 4 3

Total Page Faults = 6
```

Result:

Thus the optimal page replacement program was executed successfully.