EX.NO: 10 OPTIMAL (LFU) PAGE REPLACEMENT ALGORITHMS

Data: 07.10.2024

AIM:

To implement Optimal (The page which is not used for longest time) page replacement ALGORITHMS.

ALGORITHM:

Step1: Read the size of the frame, no. of elements and elements one by one.
Step2: Initialize the frames with value -1.
Step3: Insert each element into frame, if it's already not present.
Step4: If the frame is full and new element is not already present then replace the least
frequently used element by the new element.
Step5: Increment no. of page faults by one while inserting each element into the
frames.
Step6: Display the contents of frames during processing and the total no. of page
faults.

PROGRAM:

```
#include <stdio.h>

int main() {
   int reference_string[25], frames[25], interval[25];
   int pages, total_frames, page_faults = 0;

   printf("Enter Total Number of Pages: ");
   scanf("%d", &pages);

   printf("Enter Values of Reference String:\n");
   for (int m = 0; m < pages; m++) {
      printf("Value No.[%d]: ", m + 1);
      scanf("%d", &reference_string[m]);
   }
   printf("Enter Total Number of Frames: ");
   scanf("%d", &total_frames);

   // Initialize frames
   for (int m = 0; m < total_frames; m++) {
      frames[m] = -1;
   }
```

```
for (int m = 0; m < pages; m++) {
 int flag = 0;
// Check if the page is already in one of the frames
 for (int n = 0; n < total_frames; n++) {
    if (frames[n] == reference_string[m]) {
       flag = 1; // Page is found
       printf("\t"); // Indicate page hit
       break;
    }
 }
// Page fault occurred
 if (flag == 0) {
    int position = -1, maximum_interval = -1;
    // Find the position to replace
    for (int n = 0; n < total_frames; n++) {
       interval[n] = 0; // Reset interval
       for (int temp = m + 1; temp < pages; temp++) {
          if (frames[n] == reference_string[temp]) {
             interval[n] = temp - m; // Calculate the interval
             break;
          }
       }

       // If the page is never referenced again
       if (interval[n] == 0) {
          position = n; // Choose this frame for replacement
          break;
       }

       // Find the frame with the maximum interval
       if (interval[n] > maximum_interval) {
          maximum_interval = interval[n];
          position = n;
       }
    }

    // Replace the page
    frames[position] = reference_string[m];
    printf("FAULT\t");
    page_faults++;
 }

 // Print current frame state
 for (int n = 0; n < total_frames; n++) {
    if (frames[n] != -1) {
       printf("%d\t", frames[n]);
    }
 }
```

```
        printf("\n");
    }

    printf("\nTotal Number of Page Faults: %d\n", page_faults);
    return 0;
}
```

OUTPUT:

```
Enter Total Number of Pages: 7
Enter Values of Reference String:
Value No.[1]: 5
Value No.[2]: 6
Value No.[3]: 3
Value No.[4]: 2
Value No.[5]: 5
Value No.[6]: 1
Value No.[7]: 8
Enter Total Number of Frames: 3

FAULT    5
FAULT    6    5
FAULT    3    6    5
FAULT    2    3    5
FAULT    5    2    3
FAULT    1    2    5
FAULT    8    1    5

Total Number of Page Faults: 6
```

RESULT:

Thus the program to implement optimal page replacement algorithm waswritten and executed successfully.