Vidyavardhini's College of Engineering & Technology
Department of Computer Engineering
Academic Year : 2023-24

Experiment No 9: Page Replacement Policy FIFO
Date of Performance:05/04/24
Date of Submission:05/04/24
Name:JENIL KOTADIA
Roll No : 19

AIM: To Study and Implement page replacement Policy FIFO

OBJECTIVE:Page replacement algorithms are an Important part of virtual memory management and it help the os to decide which memory page can be moved out,making spaces for the currently needed page .however the ultimate objective of all page replacement algorithm is to reduce the number of page faults

THEORY :

Page replacement policies are essential in operating systems, particularly in managing memory. When the system's physical memory (RAM) is full, and a new page needs to be brought in, the operating system must decide which page to evict to make room for the new one. One of the simplest page replacement algorithms is the First-In-First-Out (FIFO) policy.

ADVANTAGES :

Simplicity: FIFO is one of the simplest page replacement algorithms to implement. It only requires a queue data structure to track the order in which pages were brought into memory.

Low Overhead: FIFO has minimal overhead in terms of computational complexity and memory usage. It does not involve complex calculations or data structures, making it efficient in terms of both time and space.

Predictability: FIFO's behavior is predictable and deterministic. It always selects the oldest page in memory for replacement, regardless of the access patterns or frequency of page references. This predictability makes it easy to analyze and understand its performance characteristics.

DISADVANTAGE:

Simplicity: FIFO is straightforward to implement and understand. It operates on the principle of replacing the oldest page in memory, which makes its logic easy to grasp for developers and system administrators.

Low Overhead: FIFO requires minimal overhead in terms of memory and processing resources. It only needs to maintain a queue of pages in memory, making it suitable for systems with limited computational resources.

No Complexity: FIFO does not involve complex calculations or heuristics. It is a non-adaptive algorithm, meaning it does not dynamically adjust its behavior based on page access patterns. This simplicity makes it efficient and reliable in certain scenarios.

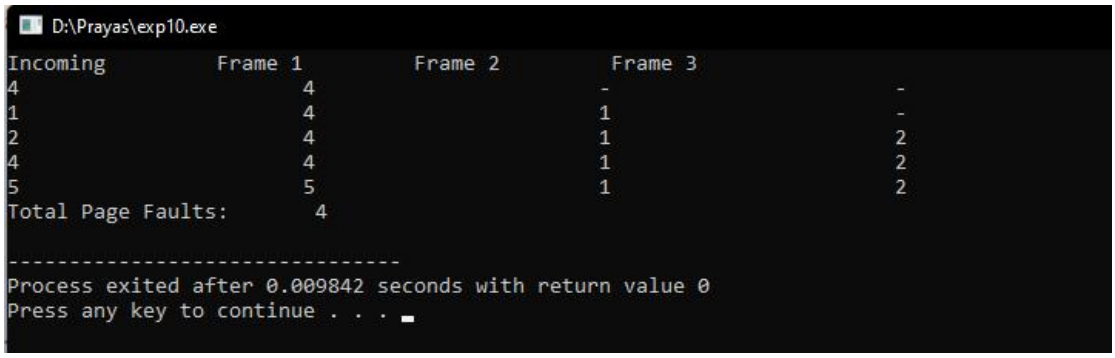
CODE:

```
#include <stdio.h>

int main() {
    int incomingStream[] = {4, 1, 2, 4, 5};
    int pageFaults = 0;
    int frames = 3;
    int m, n, s, pages;

    pages = sizeof(incomingStream) / sizeof(incomingStream[0]);

    printf("Incoming \t Frame 1 \t Frame 2 \t Frame 3\n");
```

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

for (m = 0; m < pages; m++) {
   s = 0;
   for (n = 0; n < frames; n++) {
      if (incomingStream[m] == temp[n]) {
         s++;
         pageFaults--;
      }
   }

   pageFaults++;

   if (pageFaults <= frames && s == 0) {
      temp[m] = incomingStream[m];
   } else if (s == 0) {
      temp[(pageFaults - 1) % frames] = incomingStream[m];
   }

   printf("%d\t\t\t", incomingStream[m]);
   for (n = 0; n < frames; n++) {
      if (temp[n] != -1) {
         printf("%d\t\t\t", temp[n]);
      } else {
         printf("-\t\t\t");
      }
```

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

    printf("Total Page Faults:\t %d\n", pageFaults);
    return 0;
}
```

OUTPUT:



CONCLUSION :

In conclusion, the FIFO (First-In-First-Out) page replacement policy offers several advantages that make it a practical choice in certain computing environments. Its simplicity, low overhead, fairness, and deterministic behavior make it suitable for various scenarios, particularly in systems with limited computational resources or where predictability and fairness are prioritized over fine-grained performance optimization