Department of
Computer Science and Engineering

Title: Page Replacement Algorithms

Operating System Lab

CSE 310

Green University of Bangladesh

1 Objective(s)

- To gather knowledge of different types of page replacement algorithms.
- To implement first-in first-out, least recently used, least frequently used page replacement algorithm.

2 Problem analysis

In this section, a basic page replacement algorithm, FIFO is discussed.

2.1 First-In First-Out Page Replacement Algorithm

This page replacement algorithm uses the property of queue, to handle page requests. As pages are requested, at first it is checked whether it exists in memory or not. If it exists in memory, then the page is served. But if it does not, then page fault occurs. The page is fetched from the disk and stored in the memory. If the memory is full, then the FIFO algorithm is applied. The page which came first in the memory, is swapped. The oldest page is stored in the disk and the newly requested page is fetched and stored in the memory.

3 Implementation in C

```c
#include<stdio.h>

int main(){
    int pageFaultCount = 0, pages[50], memory[20], memoryIndex = 0,
       numberOfPages, numberOfFrames, i, j, k;

    puts("Enter number of pages:");
    scanf("%d", &numberOfPages);

    puts("Enter the pages:");
    for(i=0; i<numberOfPages; i++){
        scanf("%d", &pages[i]);
    }

    puts("Enter number of frames:");
    scanf("%d", &numberOfFrames);
    for(i=0; i<numberOfFrames; i++){
        memory[i] = -1;
    }

    puts("The Page Replacement Process is -->");
    for(i=0; i<numberOfPages; i++){
        for(j=0; j<numberOfFrames; j++){
            if(memory[j] == pages[i]){
                break;
            }
        }
        if(j == numberOfFrames){
            memory[memoryIndex] = pages[i];
            memoryIndex++;
            pageFaultCount++;
        }
        for(k=0; k<numberOfFrames; k++){
            printf("\t%d", memory[k]);
        }
        if(j == numberOfFrames){
            printf("\tPage Fault No: %d", pageFaultCount);
```

```
37          }
38          puts("");
39          if(memoryIndex == numberOfFrames){
40              memoryIndex = 0;
41          }
42
43      }
44      printf("The number of Page Faults using FIFO is: %d\n", pageFaultCount);
45      return 0;
46  }
```

4 Input/Output

4.1 Input

Input of the program is given below.

```
Enter number of pages:
20
Enter the pages:
7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
Enter number of frames:
3
```

4.2 Output

Output of the program is given below.

```
The Page Replacement Process is –>
7 -1 -1 Page Fault No: 1
7 0 -1 Page Fault No: 2
7 0 1 Page Fault No: 3
2 0 1 Page Fault No: 4
2 0 1
2 3 1 Page Fault No: 5
2 3 0 Page Fault No: 6
4 3 0 Page Fault No: 7
4 2 0 Page Fault No: 8
4 2 3 Page Fault No: 9
0 2 3 Page Fault No: 10
0 2 3
0 2 3
0 1 3 Page Fault No: 11
0 1 2 Page Fault No: 12
0 1 2
0 1 2
7 1 2 Page Fault No: 13
7 0 2 Page Fault No: 14
7 0 1 Page Fault No: 15
The number of Page Faults using FIFO is: 15
```

5 Discussion & Conclusion

Based on the focused objective(s) to understand about first-in, first-out page replacement algorithm, the additional lab exercise made me more confident towards the fulfilment of the objectives(s).

6 Lab Task (Please implement yourself and show the output to the instructor)

1. Implement LRU page replacement algorithm.

6.1 Problem analysis

In this algorithm, pages are replaced in the memory, using their last access time. When the memory is full and a new page is needed to be stored in the memory, the page which was accessed the longest time ago, gets swapped with the new page.

6.2 Input

Input of the program is given below.

```
Enter number of frames: 3
Enter number of pages: 20
Enter reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
```

6.3 Output

Output of the program is given below.

```
The Page Replacement Process is –>

For 7 : 7
For 0 : 7 0
For 1 : 7 0 1
For 2 : 2 0 1
For 0 :No page fault!
For 3 : 2 0 3
For 0 :No page fault!
For 4 : 4 0 3
For 2 : 4 0 2
For 3 : 4 3 2
For 0 : 0 3 2
For 3 :No page fault!
For 2 :No page fault!
For 1 : 1 3 2
For 2 :No page fault!
For 0 : 1 0 2
For 1 :No page fault!
For 7 : 1 0 7
For 0 :No page fault!
For 1 :No page fault!
Total no of page faults using LRU is :12
```

7 Lab Exercise (Submit as a report)

- Implement LFU page replacement algorithm.

7.1 Input

Input of the program is given below.

```
Enter number of frames: 3
Enter number of pages: 20
Enter reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
```

7.2 Output

Output of the program is given below.

```
The Page Replacement Process is –>

For 7 : 7
For 0 : 7 0
For 1 : 7 0 1
For 2 : 2 0 1
For 0 :No page fault!
For 3 : 3 0 1
For 0 :No page fault!
For 4 : 4 0 1
For 2 : 2 0 1
For 3 : 2 0 3
For 0 :No page fault!
For 3 :No page fault!
For 2 :No page fault!
For 1 : 1 0 3
For 2 : 2 0 3
For 0 :No page fault!
For 1 : 2 0 1
For 7 : 2 0 7
For 0 :No page fault!
For 1 : 2 0 1
Total no of page faults using LFU is: 13
```

8 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.