

Student name: Ajay Kumar Natwariya

Student ID: 11712132

Email Address: kumarajayap172@gmail.com

Roll No.:30

Github link: https://github.com/AjayKumarNatwariya/OS_Project

Question

Consider a scenario of demand paged memory. Page table is held in registers. It takes 8 milliseconds to service a page fault if an empty page is available or the replaced page is not modified and 20 milliseconds if the replaced page is modified. Memory access time is 100 nanoseconds. Assume that the page to be replaced is modified 70 percent of the time. Generate a solution to find maximum acceptable page-fault rate for access time that is not more than 200 nanoseconds.

Definition of Terms:

- **Demand Page:** In virtual memory systems, demand paging is a type of swapping in which pages of data are not copied from disk to RAM until they are needed. In contrast, some virtual memory systems use anticipatory paging, in which the operating system attempts to anticipate which data will be needed next and copies it to RAM before it is actually required.
- **Page Service:** A Service Page is All About What You Offer – The Who, What, When, and How. The central focus of this page is to communicate your services, the process, and how someone can contact you or sign up for those services. You need to both appeal to potential clients as well as inform them.
- **Modified Page:** Modified means sometimes you might try to write something on to the page. If a page is modified, then whenever you should replace that page with some other page, then the modified information should be kept on the hard disk or it has to be written back or it has to be saved back.
- **Memory Access Time:** A transfer of pages between main memory and an auxiliary store, such as a hard disk drive, is referred to as paging or swapping.

- **Page Replacement:** In a computer operating system that uses paging for virtual memory management, page replacement algorithms decide which memory pages to page out, sometimes called swap out, or write to disk, when a page of memory needs to be allocated.
- **Page Fault:** A page fault (sometimes called #PF, PF or hard fault) is a type of exception raised by computer hardware when a running program accesses a memory page that is not currently mapped by the memory management unit (MMU) into the virtual address space of a process.
- **Maximum Access Time:** Maximum access time is the maximum time from the start of one storage device access to the time when the next access can be started

In Given Question:

To service a page fault = 8 milliseconds.

(if an empty page is available or the replaced page is not modified)

To service a page fault = 20 milliseconds

(if the replaced page is modified)

Memory access time = 100 nanoseconds.

The page to be replaced is modified percent of the time = 70%

Maximum access time = 200 nanoseconds

Maximum acceptable page-fault rate = ?

Algorithm –

- **First in First Out (FIFO):** This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.
- **Optimal Page replacement:** In this algorithm, pages are replaced which would not be used for the longest duration of time in the future. Optimal page replacement is perfect, but not possible in practice as the operating system cannot know future requests. The use of Optimal Page replacement is to set up a benchmark so that other replacement algorithms can be analyzed against it.
- **Least Recently Used (LRU):** In Least Recently Used (LRU) algorithm is a Greedy algorithm where the page to be replaced is least recently used. The idea is based on locality of reference, the least recently used page is not likely.

Step 1- Start traversing the pages.

- i. If set holds less pages than capacity.
 - a. Insert page into the set one by one until the size of set reaches capacity or all page requests are processed.
 - b. Simultaneously maintain the recent occurred index of each page in a map called indexes.
 - c. Increment page fault
- ii. Else
 - a. If current page is present in set
 - do nothing.
 - b. Else
 - Find the page in the set that was least recently used. We find it using index array. We basically need to replace the page with minimum index.
 - Replace the found page with current page.
 - Increment page faults.
 - Update index of current page.

Step 2- Return page faults.

Complexity: $O(\log(n))$

Source Code:

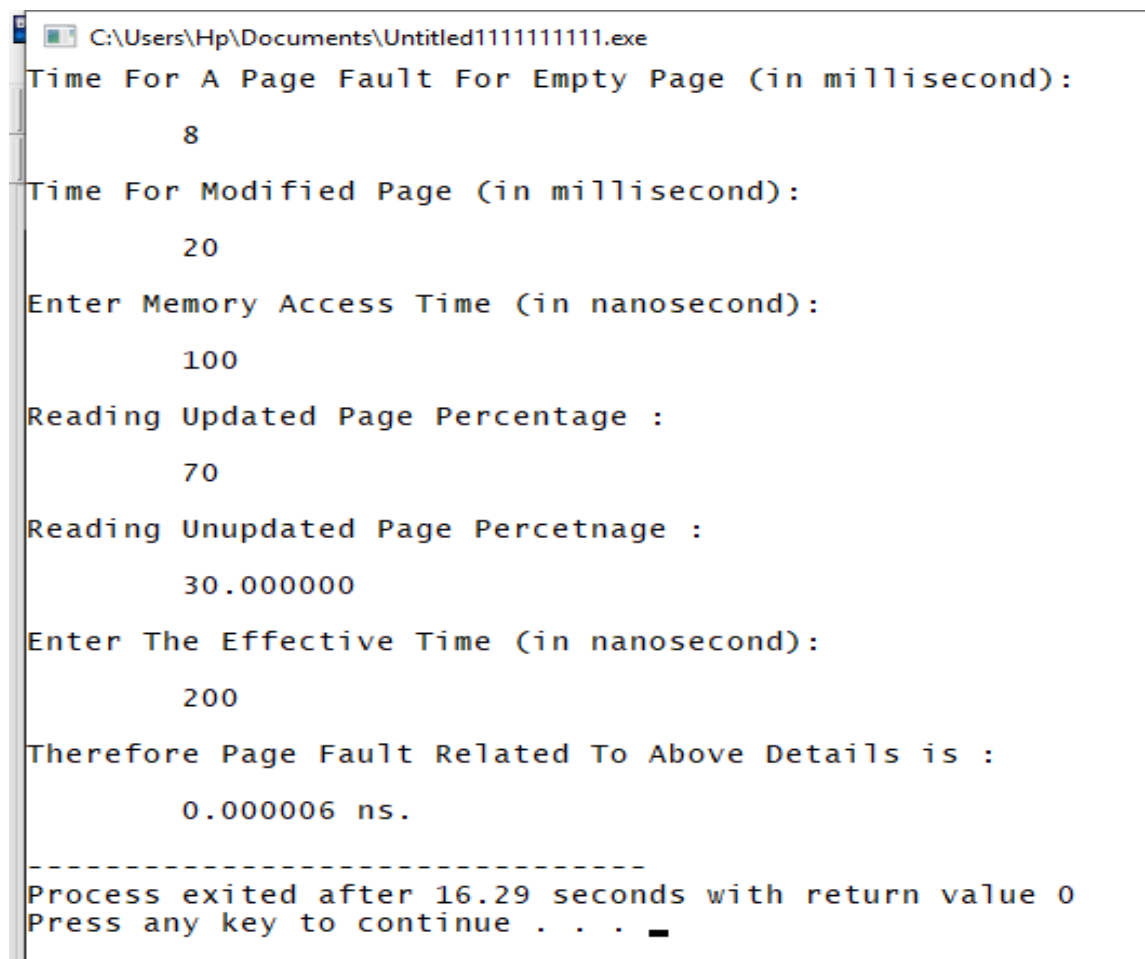
```
#include<stdio.h>
#include<math.h>
int main()
{
    float PF,PFN,PM,PMN,MAT,SUCC,SUCCESS;
    float FAIL,FAILURE,ET,PASR,LOSR,MR,FN,FI,PN;
    int C;
    float N=1000000;
    z:printf("Time For A Page Fault For Empty Page (in millisecond):\n\n\t");
    scanf("%f",&PF);
    PFN=PF*N;
    printf("\nTime For Modified Page (in millisecond):\n\n\t");
    scanf("%f",&PM);
    PMN=PM*N;
    printf("\nEnter Memory Access Time (in nanosecond):\n\n\t");
    scanf("%f",&MAT);
    printf("\nReading Updated Page Percentage :\n\n\t");
    scanf("%f",&SUCC);
    SUCCESS=SUCC/100;
    FAIL=100-SUCC;
    FAILURE=FAIL/100;
    printf("\nReading Unupdated Page Percetnage :\n\n\t%f %\n",FAIL);
    printf("\nEnter The Effective Time (in nanosecond):\n\n\t");
    scanf("%f",&ET);
    LOSR=FAILURE*PFN;
    PASR=SUCCESS*PMN;
    MR=MAT+PASR+LOSR;
    FN=MAT-MR;
    FI=MAT-ET;
    PN=FI/FN;
    if(PN>=0)
    {
        printf("\nTherefore Page Fault Related To Above Details is : ");
        printf("\n\n\t%f ns.\n",PN);
    }
    else
    {
        printf("\nData is invalid\n");
    }
}
```

```

        printf("\nEnter your Choice : \n\n\t1.Enter Data Again\n\t2.Exit\n\n\t");
        scanf("%d",&C);
        if(C=1)
            goto z;
        else
            return 0;
    }
    return 0;
}

```

Output:



```

C:\Users\Hp\Documents\Untitled1111111111.exe
Time For A Page Fault For Empty Page (in millisecond):
8
Time For Modified Page (in millisecond):
20
Enter Memory Access Time (in nanosecond):
100
Reading Updated Page Percentage :
70
Reading Unupdated Page Percetnage :
30.000000
Enter The Effective Time (in nanosecond):
200
Therefore Page Fault Related To Above Details is :
0.000006 ns.
-----
Process exited after 16.29 seconds with return value 0
Press any key to continue . . . _

```

Github Link: https://github.com/AjayKumarNatwariya/OS_Project

Thank you