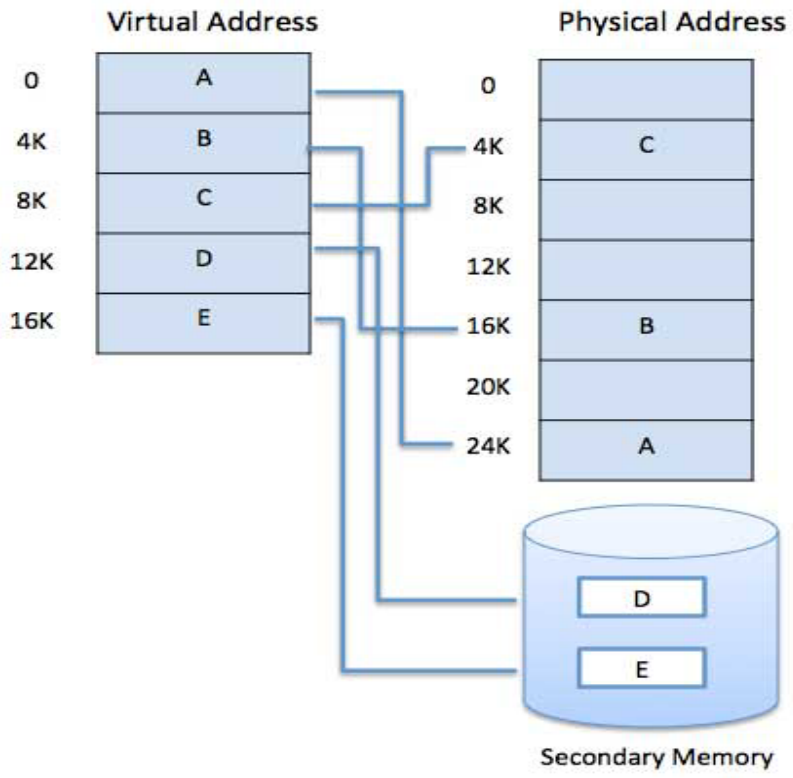
Student Name:
Student ID:11809898
Email Address: saimounikachowdary123@gmail.com

Git hub:-http://https://github.com/saimounika180/hka17_23os-project

Ques 23. 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.

Description:-

In computer operating systems, demand paging (as opposed to anticipatory paging) is a method of virtual memory management. In a system that uses demand paging, the operating system copies a disk page into physical memory only if an attempt is made to access it and that page is not already in memory (i.e., if a page fault occurs). It follows that a process begins execution with none of its pages in physical memory, and many page faults will occur until most of a process's working set of pages are located in physical memory. This is an example of a lazy loading techniuue.



Advantages :-

1. Large virtual memory.
2. More efficient use of memory.
3. There is no limit on degree of multiprogramming.

Disadvantages:-

Number of tables and the amount of processor overhead for handling page interrupts are greater than in the case of the simple paged management techniques

Code:-

```
#include<iostream>
using namespace std;
#include<stdio.h>
#include<conio.h>
#include<iostream>
#include<math.h>
using namespace std;
int main()
{
float pf,pfn,pm,pmn,mat,succ,success,fail,failure,et,pasr,losr,mr,fn,fi,pn;
float n=1000000;
cout<<"Time For A Page Fault For Empty Page :\n ";
cin>>pf;
pfn=pf*n;
cout<<"Time For Modified Page :\n";
cin>>pm;
pmn=pm*n;
cout<<"Enter Memory Access Time :\n";
cin>>mat;
cout<<"Reading Updated Page Percentage :\n";
cin>>succ;
success=succ/100;
fail=100-succ;
failure=fail/100;
cout<<"Reading Unupdated Page Percetnage :"<<fail<<"%";
cout<<"\nEnter The Effective Time : ";
cin>>et;
losr=failure*pfn;
pasr=success*pmn;
mr=mat+pasr+losr;
fn=mat-mr;
fi=mat-et;
pn=fi/fn;
system("cls");
cout<<"Therefore Page Fault Related To Above Details is : ";
cout<<"\n\n is : "<<pn<<"ns.\n";
return 0;
}
```

Output :-

```
Time For A Page Fault For Empty Page :
 8
Time For Modified Page :
20
Enter Memory Access Time :
100
Reading Updated Page Percentage :
70
Reading Unupdated Page Percetnage :30%
Enter The Effective Time : 200
```

```
Time For A Page Fault For Empty Page :
 8
Therefore Page Fault Related To Above Details is :

 is : 6.09756e-06ns.

[Program finished]
```