LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

Operating System CSE316

Page-Fault Rate

Student Name: Ansh Bargoti
Student ID: 11810525
Section: K18FG
Email ID: anshbargoti@gmail.com
GitHub Link: https://github.com/anshbargoti/Demand-Paging

QUESTION NO 21:

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 of problem:

The problem is a scenario of demand paged memory, where we are dealing with page-fault rate. 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). Memory Access Time is how long it takes for a character in RAM to be transferred to or from the CPU and Effective Access Time is essentially the (weighted) average time it takes to get a value from memory. A page fault(sometimes called hard fault) is a type of exception raised by computer hardware when a running program accesses a memory page that is not current mapped by the memory management unit into the virtual address space of a process.

In the problem, Constraint values are given for generating a solution of maximum acceptable page-fault rate.

ALGORITHM:

The algorithm for the program is as follows:

1. Get the user input with choice (i.e., a choice between finding maximum acceptable page-fault rate or to exit the program)

2. If user choose to find the maximum acceptable page-fault rate, Enter the time to service a page-fault if an empty page is available or the replaced page is not modified.

3. Get the time to service a page-fault if the replaced page is modified.

4. Take the input for memory access time.

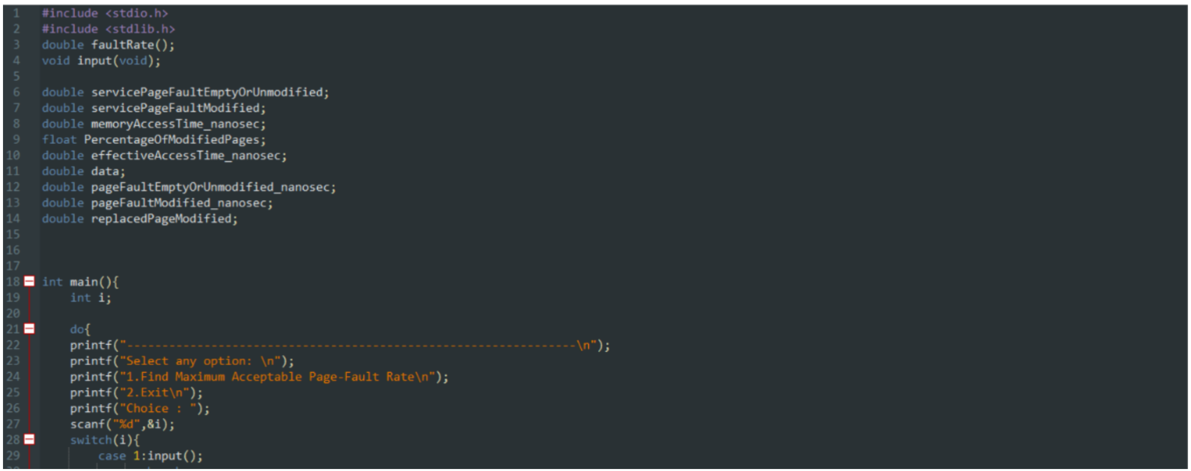
5. Get the percentage of time for which the page to be modified is replaced

6. Take the input for effective access time.

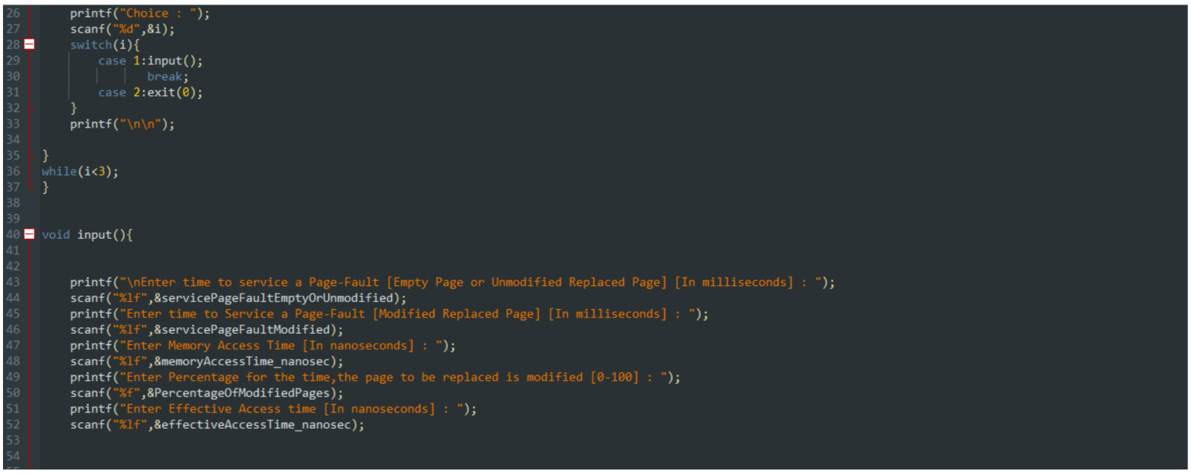
7. Print the maximum acceptable page-fault rate.

8. Exit the program.

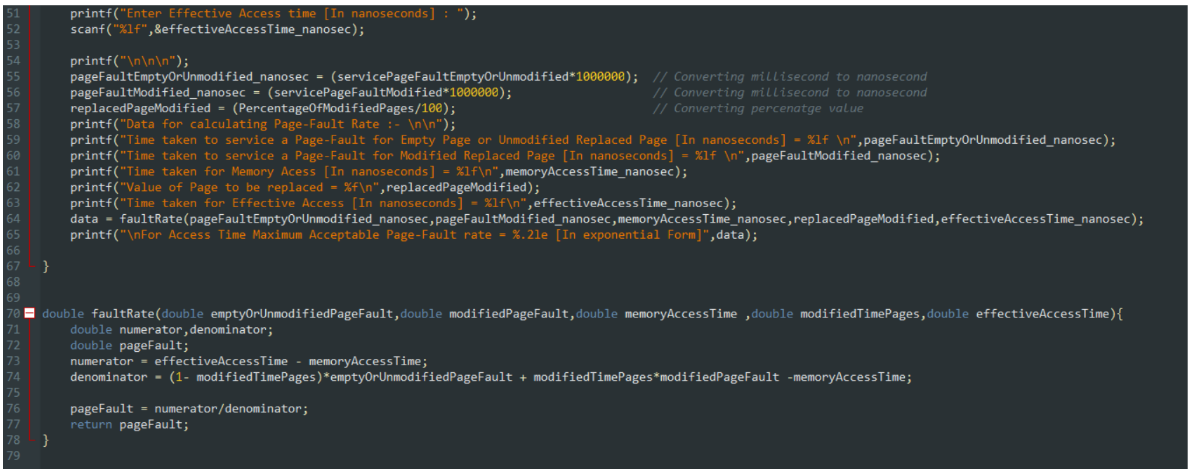
Program Snippet:



```c
#include <stdio.h>
#include <stdlib.h>
double faultRate();
void input(void);

double servicePageFaultEmptyOrUnmodified;
double servicePageFaultModified;
double memoryAccessTime_nanosec;
float PercentageOfModifiedPages;
double effectiveAccessTime_nanosec;
double data;
double pageFaultEmptyOrUnmodified_nanosec;
double pageFaultModified_nanosec;
double replacedPageModified;



int main(){
    int i;

    do{
    printf("----------------------------------------------------------------\n");
    printf("Select any option: \n");
    printf("1.Find Maximum Acceptable Page-Fault Rate\n");
    printf("2.Exit\n");
    printf("Choice : ");
    scanf("%d",&i);
    switch(i){
        case 1:input();
```

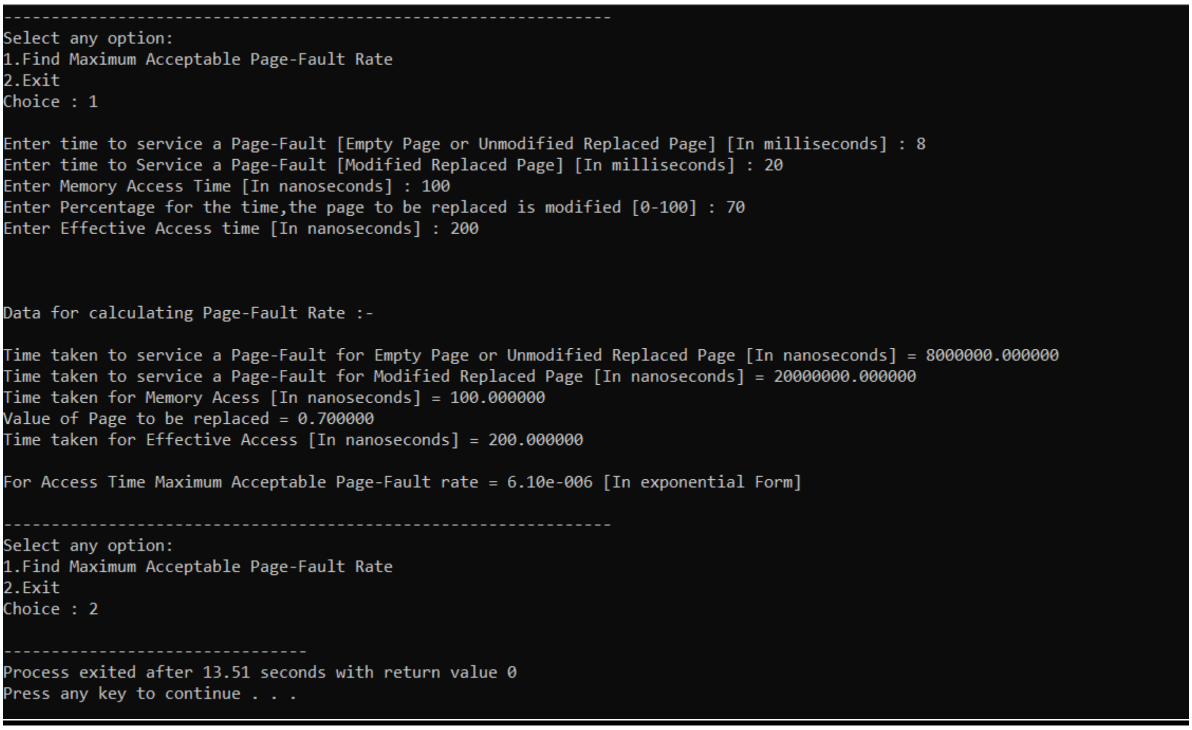


```c
    printf("Choice : ");
    scanf("%d",&i);
    switch(i){
        case 1:input();
                break;
        case 2:exit(0);
    }
    printf("\n\n");

}
while(i<3);
}


void input(){


    printf("\nEnter time to service a Page-Fault [Empty Page or Unmodified Replaced Page] [In milliseconds] : ");
    scanf("%lf",&servicePageFaultEmptyOrUnmodified);
    printf("Enter time to Service a Page-Fault [Modified Replaced Page] [In milliseconds] : ");
    scanf("%lf",&servicePageFaultModified);
    printf("Enter Memory Access Time [In nanoseconds] : ");
    scanf("%lf",&memoryAccessTime_nanosec);
    printf("Enter Percentage for the time,the page to be replaced is modified [0-100] : ");
    scanf("%f",&PercentageOfModifiedPages);
    printf("Enter Effective Access time [In nanoseconds] : ");
    scanf("%lf",&effectiveAccessTime_nanosec);


```



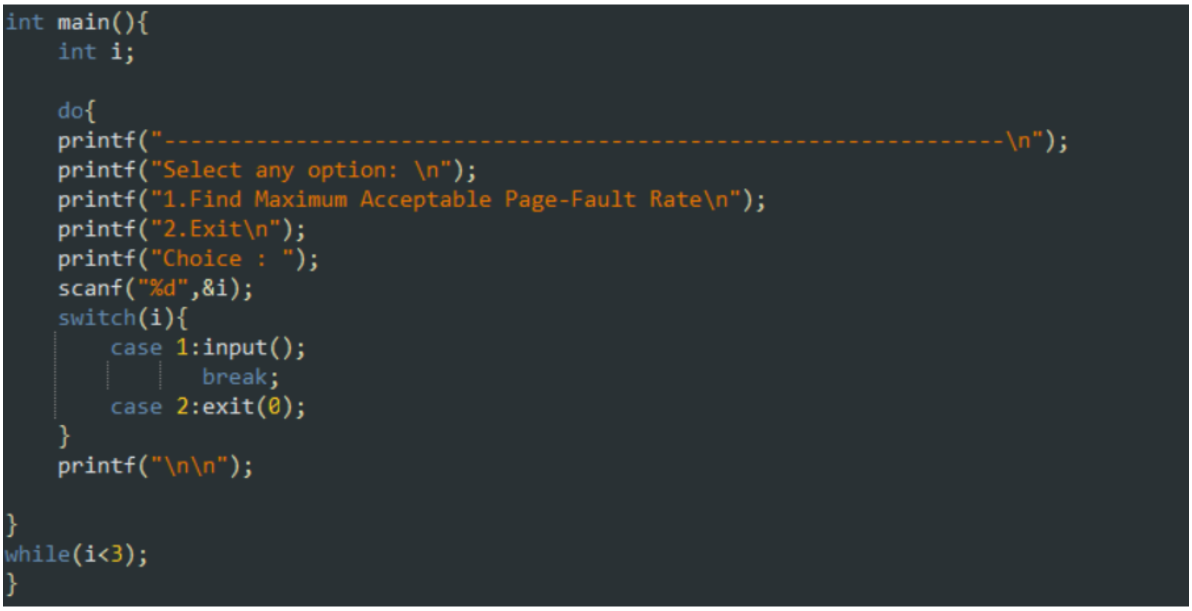
```c
    printf("Enter Effective Access time [In nanoseconds] : ");
    scanf("%lf",&effectiveAccessTime_nanosec);

    printf("\n\n\n");
    pageFaultEmptyOrUnmodified_nanosec = (servicePageFaultEmptyOrUnmodified*1000000);  // Converting millisecond to nanosecond
    pageFaultModified_nanosec = (servicePageFaultModified*1000000);                    // Converting millisecond to nanosecond
    replacedPageModified = (PercentageOfModifiedPages/100);                            // Converting percenatge value
    printf("Data for calculating Page-Fault Rate :- \n\n");
    printf("Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page [In nanoseconds] = %lf \n",pageFaultEmptyOrUnmodified_nanosec);
    printf("Time taken to service a Page-Fault for Modified Replaced Page [In nanoseconds] = %lf \n",pageFaultModified_nanosec);
    printf("Time taken for Memory Acess [In nanoseconds] = %lf\n",memoryAccessTime_nanosec);
    printf("Value of Page to be replaced = %f\n",replacedPageModified);
    printf("Time taken for Effective Access [In nanoseconds] = %lf\n",effectiveAccessTime_nanosec);
    data = faultRate(pageFaultEmptyOrUnmodified_nanosec,pageFaultModified_nanosec,memoryAccessTime_nanosec,replacedPageModified,effectiveAccessTime_nanosec);
    printf("\nFor Access Time Maximum Acceptable Page-Fault rate = %.2le [In exponential Form]",data);

}


double faultRate(double emptyOrUnmodifiedPageFault,double modifiedPageFault,double memoryAccessTime ,double modifiedTimePages,double effectiveAccessTime){
    double numerator,denominator;
    double pageFault;
    numerator = effectiveAccessTime - memoryAccessTime;
    denominator = (1- modifiedTimePages)*emptyOrUnmodifiedPageFault + modifiedTimePages*modifiedPageFault -memoryAccessTime;

    pageFault = numerator/denominator;
    return pageFault;
}

```

Output:



```
----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 1

Enter time to service a Page-Fault [Empty Page or Unmodified Replaced Page] [In milliseconds] : 8
Enter time to Service a Page-Fault [Modified Replaced Page] [In milliseconds] : 20
Enter Memory Access Time [In nanoseconds] : 100
Enter Percentage for the time,the page to be replaced is modified [0-100] : 70
Enter Effective Access time [In nanoseconds] : 200



Data for calculating Page-Fault Rate :-

Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page [In nanoseconds] = 8000000.000000
Time taken to service a Page-Fault for Modified Replaced Page [In nanoseconds] = 20000000.000000
Time taken for Memory Acess [In nanoseconds] = 100.000000
Value of Page to be replaced = 0.700000
Time taken for Effective Access [In nanoseconds] = 200.000000

For Access Time Maximum Acceptable Page-Fault rate = 6.10e-006 [In exponential Form]

----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 2

--------------------------------
Process exited after 13.51 seconds with return value 0
Press any key to continue . . .
```

Complexity:



```c
int main(){
    int i;

    do{
    printf("----------------------------------------------------------------\n");
    printf("Select any option: \n");
    printf("1.Find Maximum Acceptable Page-Fault Rate\n");
    printf("2.Exit\n");
    printf("Choice : ");
    scanf("%d",&i);
    switch(i){
        case 1:input();
              break;
        case 2:exit(0);
    }
    printf("\n\n");

}
while(i<3);
}
```

The overall complexity of the program is **Constant**

Constraints:

Constraints are restrictions put on the program by the coder keeping the output in mind. For the given program certain constraints have been put to make the program efficient. The constraints are as follows:

1. Choice entered by user should be 0 or 1 initially, for making the program further executive(by choosing 0) or exiting(by choosing 1).
2. Every variable value lies in double data type range except percentage(In float data type).
3. Memory Access Time and Effective Access Time must be in nanoseconds, and remaining variables should be in milliseconds.

Boundary Conditions:

The boundary condition in this program is to enter the value of every variable in between the range of their data type. If the value of any variable is out of range, then the program will not display the required output.

Solution:

We have to find page-fault rate('p') for given effective access time('EAT') so by using the following equation, solve for 'p':

(Note: 1 millisecond = 1,000,000 nanoseconds = 1e6 nanoseconds)

EAT= (1-p)*(memory access time) + p*(1-percentage value)*(page fault service with no moddcation) + p*(percentage value)*(page fault service with moddcation)

EAT = (1-p)*(100) + (p)*((1-0.7)*(8msec) + (0.7)*(20msec))

= 100 - 100p + (2.4e6)*p + (14e6)*p

= 100 + (16.4e6)*p - 100p

200 = 100 + (16.4e6)*p - 100p

p =6.0975981560e-6 ~ 6.1e-6

p-->page Fault Rate

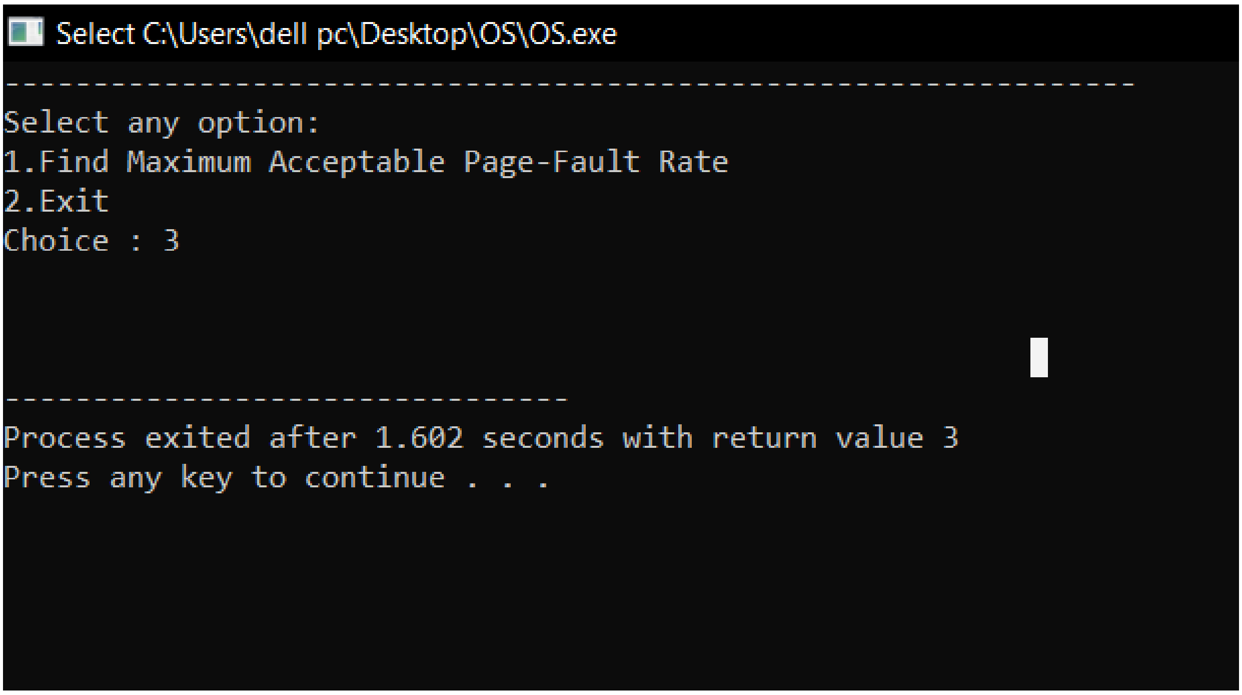
Test Cases:

1.

Description: Value of choice greater than 2

Expected Output: Program should terminate

Actual Output:



```
----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 3




--------------------------------
Process exited after 1.602 seconds with return value 3
Press any key to continue . . .
```

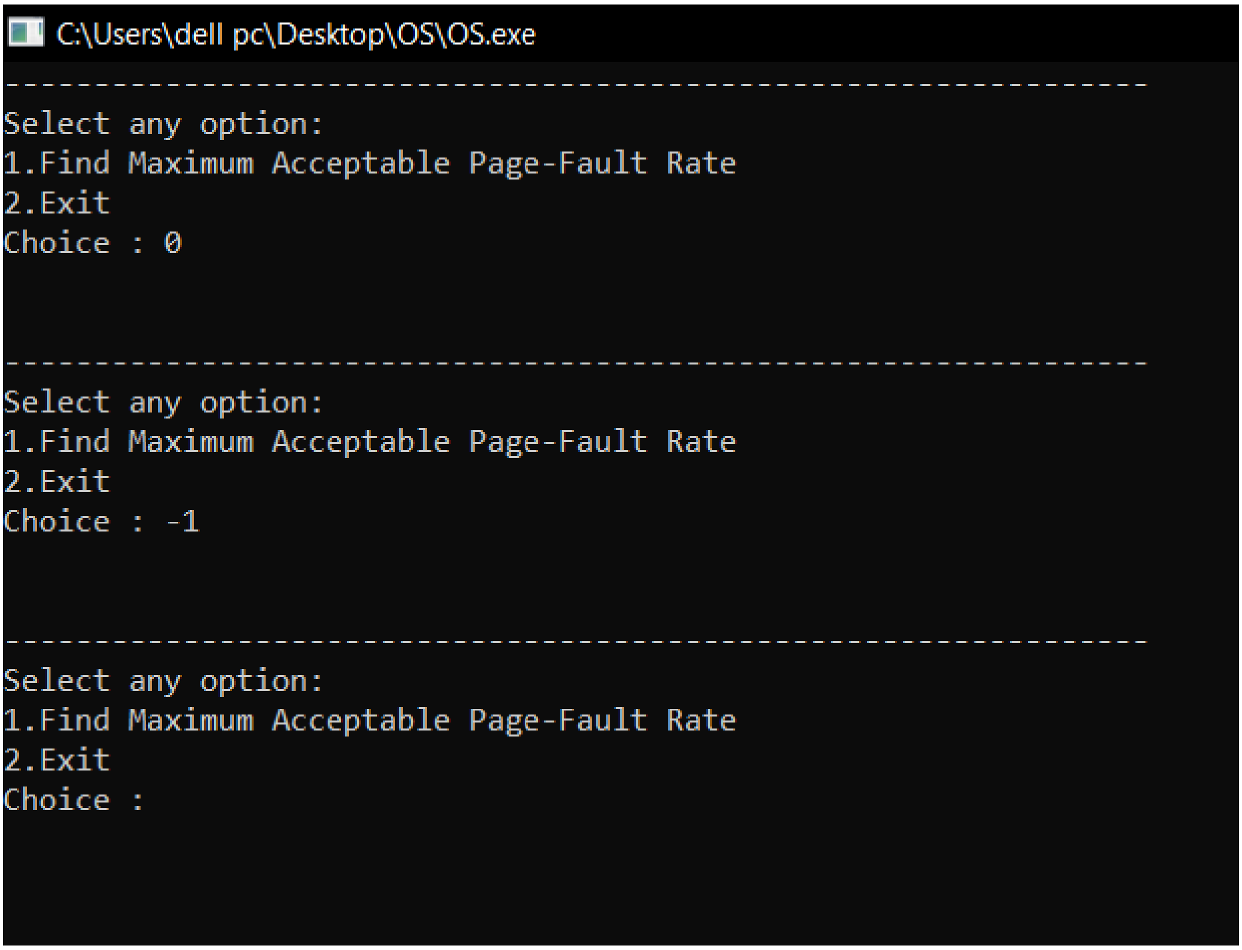
Conclusion: PASS

2.

Description: Value of choice less than 1

Expected Output: Program should iterate

Actual Output:



```
C:\Users\dell pc\Desktop\OS\OS.exe
----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 0


----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : -1


----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice :
```

Conclusion: PASS

3.

Description: Value of choice is 1 for program execution,

Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page = 10ms.

Time taken to service a Page-Fault for Modified Replaced Page = 25ms

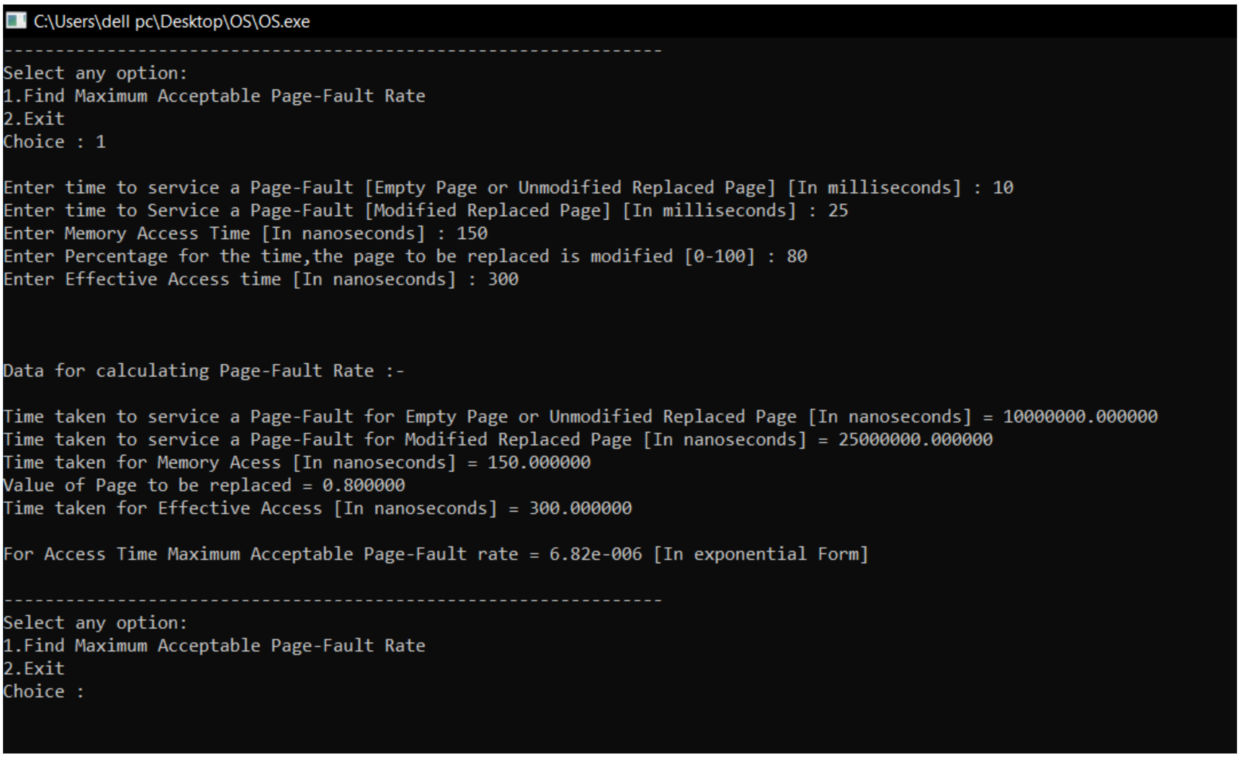
Memory access time = 150ns

Percentage for the time, the page to be replaced is modified = 80%

Effective Access time = 300ns

Expected Output: Program should execute successfully and print output for maximum acceptable page-fault rate.

Actual Output:



```
----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 1

Enter time to service a Page-Fault [Empty Page or Unmodified Replaced Page] [In milliseconds] : 10
Enter time to Service a Page-Fault [Modified Replaced Page] [In milliseconds] : 25
Enter Memory Access Time [In nanoseconds] : 150
Enter Percentage for the time,the page to be replaced is modified [0-100] : 80
Enter Effective Access time [In nanoseconds] : 300



Data for calculating Page-Fault Rate :-

Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page [In nanoseconds] = 10000000.000000
Time taken to service a Page-Fault for Modified Replaced Page [In nanoseconds] = 25000000.000000
Time taken for Memory Acess [In nanoseconds] = 150.000000
Value of Page to be replaced = 0.800000
Time taken for Effective Access [In nanoseconds] = 300.000000

For Access Time Maximum Acceptable Page-Fault rate = 6.82e-006 [In exponential Form]

----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice :
```

Conclusion: PASS

4.

Description: Value of choice is 1 for program execution,

Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page = 30ms.

Time taken to service a Page-Fault for Modified Replaced Page = 35ms

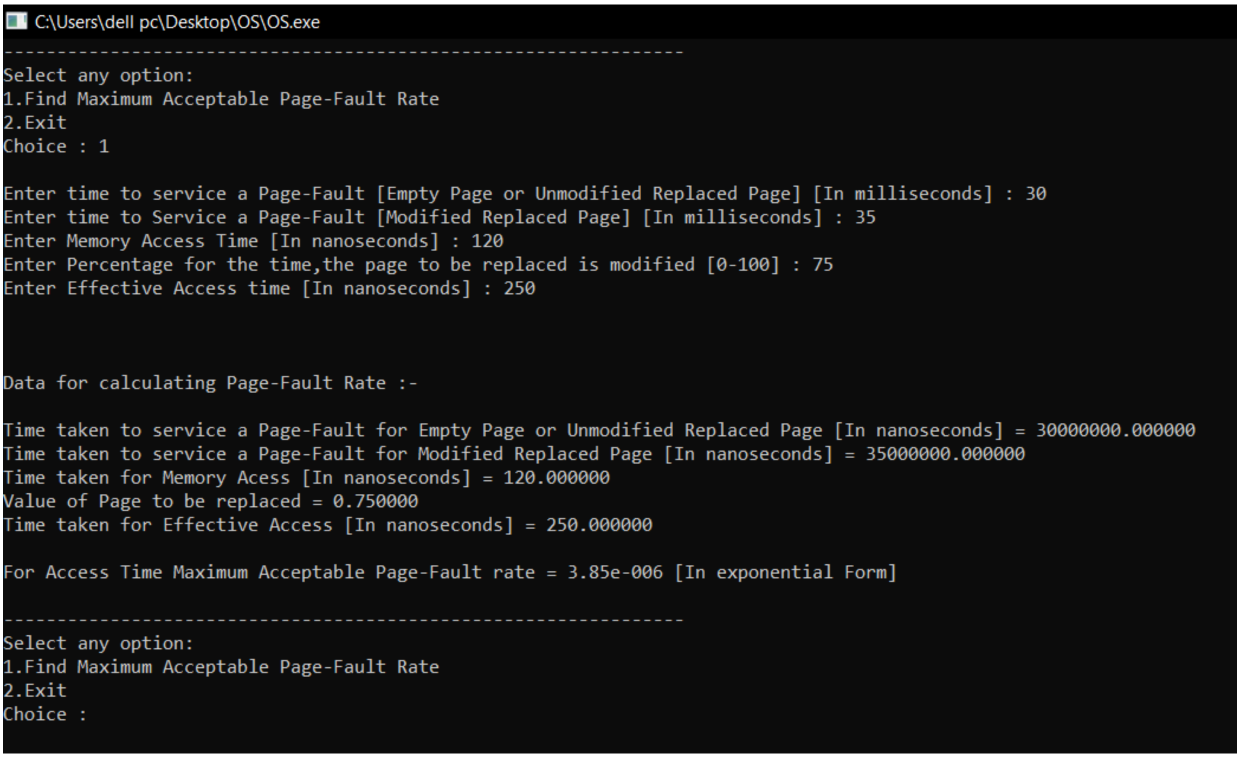
Memory access time = 120ns

Percentage for the time, the page to be replaced is modified = 75%

Effective Access time = 250ns

Expected Output: Program should execute successfully and print output for maximum acceptable page-fault rate.

Actual Output:



```
----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 1

Enter time to service a Page-Fault [Empty Page or Unmodified Replaced Page] [In milliseconds] : 30
Enter time to Service a Page-Fault [Modified Replaced Page] [In milliseconds] : 35
Enter Memory Access Time [In nanoseconds] : 120
Enter Percentage for the time,the page to be replaced is modified [0-100] : 75
Enter Effective Access time [In nanoseconds] : 250



Data for calculating Page-Fault Rate :-

Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page [In nanoseconds] = 30000000.000000
Time taken to service a Page-Fault for Modified Replaced Page [In nanoseconds] = 35000000.000000
Time taken for Memory Acess [In nanoseconds] = 120.000000
Value of Page to be replaced = 0.750000
Time taken for Effective Access [In nanoseconds] = 250.000000

For Access Time Maximum Acceptable Page-Fault rate = 3.85e-006 [In exponential Form]

----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice :
```

Conclusion: PASS

5.

Description: Value of choice is 1 for program execution,

Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page = 8ms.

Time taken to service a Page-Fault for Modified Replaced Page = 20ms

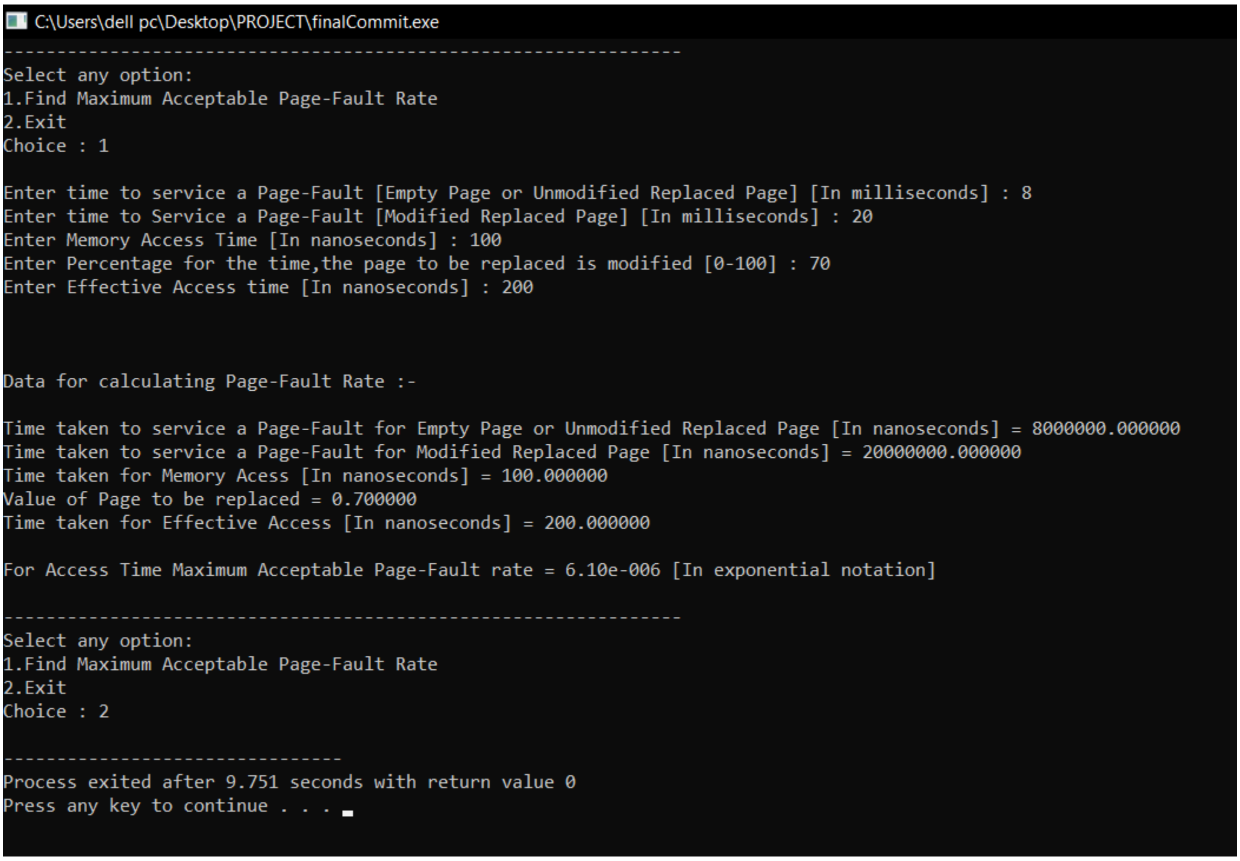
Memory access time = 100ns

Percentage for the time, the page to be replaced is modified = 70%

Effective Access time = 200ns

Expected Output: Program should execute successfully and print output for maximum acceptable page-fault rate.

Actual Output:



```
----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 1

Enter time to service a Page-Fault [Empty Page or Unmodified Replaced Page] [In milliseconds] : 8
Enter time to Service a Page-Fault [Modified Replaced Page] [In milliseconds] : 20
Enter Memory Access Time [In nanoseconds] : 100
Enter Percentage for the time,the page to be replaced is modified [0-100] : 70
Enter Effective Access time [In nanoseconds] : 200



Data for calculating Page-Fault Rate :-

Time taken to service a Page-Fault for Empty Page or Unmodified Replaced Page [In nanoseconds] = 8000000.000000
Time taken to service a Page-Fault for Modified Replaced Page [In nanoseconds] = 20000000.000000
Time taken for Memory Acess [In nanoseconds] = 100.000000
Value of Page to be replaced = 0.700000
Time taken for Effective Access [In nanoseconds] = 200.000000

For Access Time Maximum Acceptable Page-Fault rate = 6.10e-006 [In exponential notation]

----------------------------------------------------------------
Select any option:
1.Find Maximum Acceptable Page-Fault Rate
2.Exit
Choice : 2

--------------------------------
Process exited after 9.751 seconds with return value 0
Press any key to continue . . .
```

Conclusion: PASS