Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

Operating System Report File

Submitted By-

SHEKH ALIUL (Roll-35)

Submitted to-

Ms. AMANDEEP KAUR

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

Code :

```
//program for Implementing page fault counts using this basic idea


#include<stdio.h>
int main()
{
   int m,n;
      printf("Enter total number of elements\n");
       scanf("%d",&m);//taking input from user for total number of elements in array
       printf("Enter total number of frames\n");
       //taking total number of frames
       scanf("%d",&n);
   int arr[m];//for storing the elementswe are using array structure inplace of queue
   int frame[n];//using the FIFO approch in case of same number of counts to resolve the tie
   int count[n],z;//use of count array for take care of increment and decrement
   int counter=0;//intial elements are fixed for page fault
   printf("Enter elements of array \n");

      for(int i=0;i<m;i++){
   scanf("%d",&arr[i]);
   }



    int k=0,flag=0,index=0,min=0,d=0;

      for(int i=0;i<n;i++){

      frame[i]=arr[i];
      count[i]=count[i]+1;
```

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

```
    }


   counter=4;//inserting four elements
          for(int i=n;i<m;i++){
      for(int j=0;j<n;j++){

    if(frame[j]==arr[i]){
    //if we found out that the element is present will be use in future we change the flag to 1
      flag=1;
      index=j;
      break;
         }
    }

    //checking the condition if the given elment will be use or not using flag
    if(flag==0){
      frame[k%(n)]=arr[i];
      counter++;
      //increment the total number of page fault
       k++;
    }
      flag=0;
   }



    for(int i=0;i<m;i++)
  {

    if(frame[z%n]!=arr[i])
    {
```

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

```
        for(int j=i+1;j<m;j++)



    {
            if(frame[z%n]==arr[n])
               {
                  int x=0;
                    min--;//if we find out the there is a smallest element we put the element in that
frame and decrease it by 1
                     }
            }
          }
        }
      if( z==0)
     {
     count[z]=count[z%n]+1;//otherwise increment it by 1;
       }


  printf("Total number of page fault is %d",counter);


  }
```

//Code Ends here

1.Description:

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

Page replacement algorithms are the techniques using which an OS decides which memory pages to swap out, write in memory when a page of memory needs to be allocated. Paging happens whenever a page fault occurs and a free page cannot be used for allocation purpose accounting to reason that pages are not available or the number of free pages is lower than required pages.

A page replacement algorithm looks at the limited information about accessing the pages provided by hardware, and tries to select which pages should be replaced to minimize the total number of page misses, while balancing it with the costs of primary storage and processor time of the algorithm itself. There are many different page replacement algorithms. We evaluate an algorithm by running it on a particular string of memory reference and computing the number of page faults.

A page replacement algorithm looks at the limited information about accessing the pages provided by hardware, and tries to select which pages should be replaced to minimize the total number of page misses, while balancing it with the costs of primary storage and processor time of the algorithm itself. There are many different page replacement algorithms. We evaluate an algorithm by running it on a particular string of memory reference and computing the number of page faults,

2. Algorithm & 3. Complexity:

This is the Algo I come up with complexities mentioned in each line.
The algorithm minimizes number of faults by distributing heavily used pages evenly over all of memory, rather than having them compete for a small number of page frames.

Step:1--> **Set Counter=0,arr[]=0,count[]=0,flag=0,i=0,j=0,frame[]=0 and m(number of elements),n (number of frames);**
Complexity- O (1)

Note: Counter Determines the number of page fault.

Step:2-->**Push First 4 elements in frame array and Increase the counter by 4**
(Initial Condition)**Complexity- O (1)**

Step:3-->**Loop through all elements from n to until end is reached --> Complexity- O (n)**

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

Step:3.1--> **Use a nested for loop from j=0 to n and check if this element is already in frame array. ---> Complexity- O (n*n)**

Step3.2-->**If element is not in frame array && Count array have same numbers:**

Step3.2.1-->**If element not used in future:**

we will increase the counterarray[j] by 1 and pushed it to the frame array using FIFO Complexity- O (1)

else:

we will decrease the counterarray[j] by 1 and pushed it to the frame array using FIFO Complexity- O (1)

Else:

Step3.2.1-->**If element not used in future:**

we will increase the counterarray[j] by 1 and pushed it to the frame array index(which have smallest number of counter)Complexity- O (1)

Else:

we will decrease the counterarray[j] by 1 and pushed it to the frame array index(which have smallest number of counter)

Complexity- O (1)

Step4--->**If element is in frame array already then:**

continue:

Complexity- O (1)

Step:5--> Exit

Overall Complexity of algorithm = O(n)+O(n*)+O(n*n) == **O(n*n)**

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

CONSTRAINTS

- **Associate with each page frame a counter of the number of pages that are associated with that frame**

```
int arr[m];//for storing the elementswe are using array structure inplace of queue
int frame[n];//using the FIFO approch in case of same number of counts to resolve the tie

int count[n],z;//use of count array for take care of increment and decrement in page frame array

int counter=0;//intial elements are fixed for page fault
```

To apply this Constrain I used an array of same size of nmber of frame to keep a check on frame counter.

If I want to increase counter of any frame I am going to just increased the index value of that page frame in this way I maintain the counter

- **Replace a page, we search for the page frame with the smallest counter**

```
}

//checking the condition if the given elment will be use or not using flag
if(flag==0){
    frame[k%(n)]=arr[i];//Susbtitute the value of arr[i] to frame array with smallest counter using flag
    counter++;
    //increment the total number of page fault
     k++;
}
    flag=0;
}
```

First we calculate the smallest element in counter array using flag value by iterating all over the counter after findind the index just replaced the value of same index in frame array with the main array.

5.

No Additional algo are used. This algo is based on pure logicor we can say brute force.

6. BOUNDARY CONDITONS

- **WHEN LOOP WILL TERMINATE?**

  Loop will terminate when iterator reached the end
- **INCREMENT OF COUNTER ?**

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

When there is need of page in future and also it is not in frame array

- **DECREMENT OF COUNTER?**

  When There is no need of that page in future we will just decrement the counter

- **HOW TO BREAK THE TIES OF COUNTER?**

  We will use FIFO conecpt of os to break the ties

7.

TESTCASE:

b. How many page faults occur for your algorithm for the following reference string, for four page frames(take page frame size from user, Mention details might be use for testing) ? 1, 2, 3,4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, 4, 5, 4, 2.

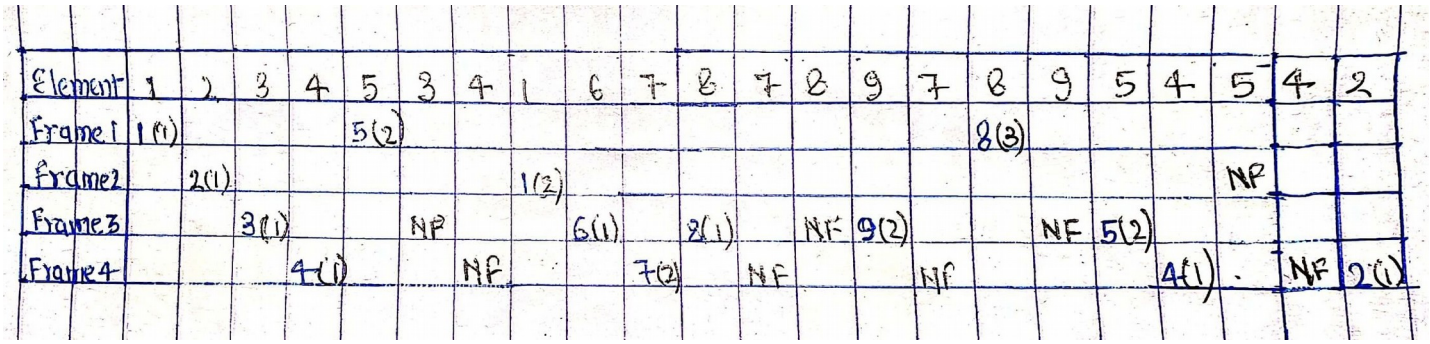
Input: 22 (main array size)

Input: 4 (Frame array size)

Input: 1 2 3 4 5 3 4 1 6 7 8 9 7 8 9 5 4 5 4 2 (elements of main array)

Output: Total Number od page fault is : 14

Explanation:



Total page Faults=14.

Take figure shown above that given in table

1. Page 1,2,3 and 4 are filled to all 4 frames counter are 1.
2. Page 5 moves to Frame-1, based on FIFO algorithm its counter is 2 now.
3. Page 3 and 4 are available, so no fault occurs and counter remain same

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

4. Page 1 is moved to Frame-2 based on FIFO algorithm and its counter is changed to 2
5. Page 6 is moved to Frame-3 based o FiFO algorithm and its counter is changed to 2.But sice 3 is not required later on,its counter is again decreased by 1. So its counter is now 1 again.
6. Page 7 is moved to Frame 4 based on FIFO algorithm and its counter is changed to 2
7. Page 8 is moved to Frame-3 based on new algortihm (its counter is lowest0 and it socunter is changed to 2. But since 6 is not required later on its counter is again decreased by 1. So its counter is now 1 again.
8. Page 7,8 are available in frames 3 and 4 so no fault occurs and count remain same.
9. Page 9 is moved to Frame-2 based on new algorithm (its counter is lowest) and its counter is changed to 2.
10. All the counters are 2 now, so Page 8 is moved to Frame-1 with FIFO algo and its counter is changed to 2
11. Page 9 is available in frames 3 so no fault occurs and count does not incresed.
12. Page 5 is moved to frame-2 based on FIFO algorithm and counter is changed to 2. However 1 will not be required again so it is changed to 1 back.
13. Page 4 is moved to frame4 based on FIFO algorithm and counter is changed to 2. However, 7 will not be required again.so it is changed to 1 back.
14. Page 5 and 4 are available in frames 2 and 3 so no fault occurs and counter remain same.
15. Page 2 is moved to frame-4 Based on new algorithm.
16. The total number of faults are 14.

(a). Define a page-replacement algorithm using this basic idea. Answer the following issues:

1) What is the initial value of the counter;

Ans: The initial value of the counter is zero

2) When is the counter increased and when is the counter is decreased

Ans: The counters are increased whenever a new page is associated with that frame, and a counter is decreased whenever one of the pages associated with that frame is no longer needed

3) How is the page to be replaced selected?

Student Name: Shekh Aliul
Student Id:11802246
Email Address: shekhaliul44@gmail.com
Github Link:https://github.com/alii13/os-assignment-15

Ans: A page to be replaced is selected as the frame with the smallest counter, with a FIFO queue to break ties

(c). What is the minimum number of page faults for an optimal page-replacement strategy for the same reference string using four page-frames(result should be dependent upon frame entered by user as described in part b)**?**

ELEMENT	1	2	3	4	5	3	4	1	6	7	8	7	8	9	7	8	9	5	4	5	4	2
FRAME-1	1							NF	6		8		NF			NF						
FRAME-2		2			5													NF		NF		2
FRAME-3			3			NF				7		NF			NF				4		NF	
FRAME-4				4			NF							9			NF					
FAULTS=	22-11	11																				

Within same page frames 4 **Total number of page fault is =11**

8.have you made minimum 5 commits of solution in github?

Ans : Yes

Github Link:https://github.com/alii13/os-assignment-15