Ex. No.: 11a)
Date:

FIFO PAGE REPLACEMENT

Aim:

To find out the number of page faults that occur using First-in First-out (FIFO) page replacement technique.

Algorithm:

1. Declare the size with respect to page length
2. Check the need of replacement from the page to memory
3. Check the need of replacement from old page to new page in memory 4.
Form a queue to hold all pages
5. Insert the page require memory into the queue
6. Check for bad replacement and page fault
7. Get the number of processes to be inserted
8. Display the values

Program Code:

```
def fifo():
    global a, n, m
    f = -1
    page_faults = 0
    page = []

    for i in range(m):
        page.append(-1)

    for i in range(n):
        flag = 0
        for j in range(m):
            if page[j] == a[i]:
                flag = 1
                break

        if flag == 0:
            f = (f + 1) % m
            page[f] = a[i]
            page_faults += 1
            print("\n%d ->" % (a[i]), end=" ")
            for j in range(m):
                if page[j] != -1:
                    print(page[j], end=" ")
                else:
                    print("-", end=" ")
        else:
            print("\n%d -> No Page Fault" % (a[i]))

    print("\nTotal page faults : %d" % (page_faults))

a = []
n = int(input("\nEnter the size of reference string: "))
```

```
for i in range(n):
    a.append(int(input("Enter [%2d] : " % (i+1))))

m = int(input("\nEnter page frame size: "))
fifo()
```

Sample Output:

```
[root@localhost student]# python fifo.py

Enter the size of reference string: 20
Enter [ 1] : 7
Enter [ 2] : 0
Enter [ 3] : 1
Enter [ 4] : 2
Enter [ 5] : 0
Enter [ 6] : 3
Enter [ 7] : 0
Enter [ 8] : 4
Enter [ 9] : 2
Enter [10] : 3
Enter [11] : 0
Enter [12] : 3
Enter [13] : 2
Enter [14] : 1
Enter [15] : 2
Enter [16] : 0
Enter [17] : 1
Enter [18] : 7
Enter [19] : 0
      Enter [20] : 1

      Enter page frame size : 3

     7 -> 7 - -
     0 -> 7 0 -
      1 -> 7 0 1
2 -> 2 0 1
0 -> No Page Fault


3 -> 2 3 1
      0 -> 2 3 0
4 -> 4 3 0
2 -> 4 2 0
3 -> 4 2 3
0 -> 0 2 3
3 -> No Page Fault 2 ->
No Page Fault
      1 -> 0 1 3
2 -> 0 1 2
0 -> No Page Fault
      1 -> No Page Fault
      7 -> 7 1 2
      0 -> 7 0 2
```

1 -> 7 0 1
Total page faults: 15.
[root@localhost student]#

Output:

```
def fifo():
    global a, n, m
    f = -1
    page_faults = 0
    page = []

    for i in range(m):
        page.append(-1)

    for i in range(n):
        flag = 0
        for j in range(m):
            if page[j] == a[i]:
                flag = 1
                break

        if flag == 0:
            f = (f + 1) % m
            page[f] = a[i]
            page_faults += 1
            print("\n%d ->" % (a[i]), end=" ")
            for j in range(m):
                if page[j] != -1:
                    print(page[j], end=" ")
                else:
```

input

```
 -> 7 1 2
 -> 7 0 2
 -> 7 0 1
Total page faults : 15


..Program finished with exit code 0
```

Result:
Program is successfully executed and output is verified.