Ex. No.: 11a)
Date: 16/4/25

## 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:

# windlude 2 status-le)

viit main ()

2 virt n;

viit at ?

Scary ["./.d"; &n);

sary ["./.d"; &f);

point temple f);

for last m=0; m2 n; i++)

2 scary ["./.d"; & a [m]);

g

to [m=0; m2 f: m++)

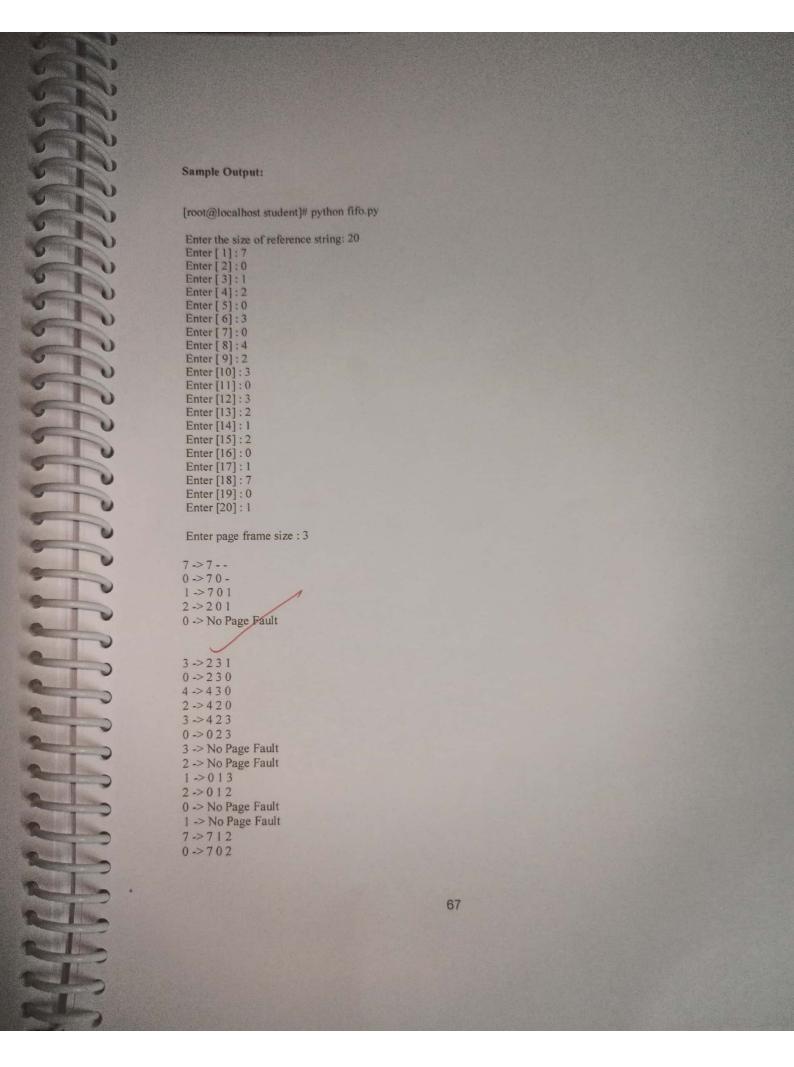
2 temp [m] = -1;

9

65

for (m=0; m = n; m++) \$=0; for (j=0; 32 f; 5+4) if (a cm) == temp(g))

8 fet; y Pf -- ; 段七十; if ((18 2= 1) ss (1==0)) temp [m]= a [m]; elx if ({ ==0) & toup [(Pf-1)/. f]=a(m); prints ("In"); point [". y.d lt [+ [+", a[m]); Ports=0:, j 2 d'ij et) if ( temp[j] ! =-1) parity (" . I. d. lt (t ( templi)); paint { 1 " - 1+ 1+ (+"); painty ("In Pf); 66 actumo;



1) 0 O V 1-701 Total page faults: 15. [root@localhost student]# 5 July by by by by by ded deleter 1245 3 A 4 2 4 2 2 5 5 Result: code for FIFO page replacement is successfully. ea e cuted 68