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:

```
#include <stdio.h>
int main()
{
   int pageFaults = 0;
   int frames = 0;
   int m, n, s, pages;
   printf("Enter the no.of pages in reference string :");
   scanf("%d",&pages);
   int page[pages];
   printf("Enter the page values :");
   for(m=0;m<pages;m++)
      scanf("%d",&page[m]);
   printf("Enter the no.of frames :");
   scanf("%d",&frames);
   printf("Reference String \t Frame 1 \t\t Frame 2 \t\t Frame 3 ");
   int temp[frames];
   for(m = 0; m < frames; m++)
   {
      temp[m] = -1;
   }
   for(m = 0; m < pages; m++)
   {
```

```
        s = 0;
        for(n = 0; n < frames; n++)
        {
            if(page[m] == temp[n])
            {
                s++;
                pageFaults--;
            }
        }
        pageFaults++;
        if((pageFaults <= frames) && (s == 0))
        {
            temp[m] = page[m];
        }
        else if(s == 0)
        {
            temp[(pageFaults - 1) % frames] = page[m];
        }
        printf("\n");
        printf("%d\t\t\t",page[m]);
        for(n = 0; n < frames; n++)
        {
            if(temp[n] != -1)
                printf(" %d\t\t\t", temp[n]);
            else
                printf(" - \t\t\t");
        }
    }
    printf("\n\nTotal Page Faults:%d\n", pageFaults);
    return 0;
}
```