i) The page replacement Algorithm : FIFO, Optimal & LRU

```
#include <stdio.h>
#include <stdlib.h>
void printFrames (int frames[], int n, const char* msg){
    for(int i=0 ; i<n; i++){
        if(frames[i] == -1){
            printf ("-");
        }
        else {
            printf ("%d", frames[i]);
        }
    }
    printf("%s\n", msg);
}

void fifo (int pages[], int n, int frames[], int frame
        count){
    int front=0 , faults =0;
    printf(" The page Replacement Process for FIFO
        is : \n");
    for(int i = 0 ;i<n; i++){
        int found=0;
        for(int j=0 ;j<frame count ; j++){
            if (frames [j] == pages [i]){
                found=1;
                break;
            }
        }
        if(!found){
            frames[ front] = pages [i];
            front = (front +1)% framecount ;
            faults ++ ;
            char msg[20];
            snprintf(msg, sizeof(msg), "PF No.%d",
                     faults);
            printframes(frames, framecount, msg);
        }
        else{
            printframes(frames, framecount, " ");
        }
    }
    printf("The nor of page faults using FIFO are
         \n", faults);
}
void lru(int pages[], int n, int frames[], int frame
        count){
    int time[framecount], faults=0, counter=0;
    printf("The page replacement process for LRU is:
         \n");
    for(int i=0; i<frame count; i++){
        frames[i] = -1;
        time[i] = -1;
    }
    for(int i=0; i<n; i++){
        int found=0, least = counter;
        for(int j=0; j<framecount; j++){
            if(frames[j]==pages[i]){
                found=1;
                time[j] = counter++;
                break;
            }
            if(time[j] < least){
                least = time[j];
            }
        }
```

```
if (!found ) {
    int replace =0;
    for (int j=0 ; j< framecount ; j++ ){
        if (time [j] == least ) {
            replace = j;
            break;
        }
    }
    frames [replace] = pages [i];
    time [replace] = Counter ++;
    faults ++;
    char msg [20];
    snprintf (msg, sizeof (msg) , "PF No.%d",
            faults );
    printframes (frames, framecount , msg);
}
else {
    printframes (frames, framecount, " ");
}
}
printf (" The nor of page faults using LRU are
        %d \n", faults);
}
void optimal ( int pages [], int n, int frames [], int
        framecount ){
int fault =0;
printf (" The page replacement process for
        optimal is : \n");
for (int i=0 ; i<n ; i++){
    int found = 0;
    for (int j =0 ; j < framecount ; j++ ){
        if (frames [j] == pages [i] ){
            found = 1;
            break;
        }
    }
    if(! found){
        int replace = -1, farthest = i;
        for (int j = 0 ; j < framecount ; j++){
            int nextUse = n;
            for (int k = i+1; k<n ; k++){
                if (frames [j] == pages [k]) {
                    nextUse = k;
                    break ;
                }
            }
            if (next Use > farthest) {
                farthest = nextUse;
                replace = j;
            }
        }
        if (replace ==-1){
            replace = 0;
        }
        frames[replace] = pages [i];
        faults ++;
        char msg [20];
        snprintf(msg, sizeof(msg), "PF no. %d",
                faults );
        printframes(frames, framecount, msg);
    }
    else {
printframes (" The no of page faults using optimal
        are %d \n", faults );
}
```

```
int main(){
    int n, framecount;
    printf ("Enter nor of frames :");
    scanf ("%d", & framecount);
    printf(" Enter nor of pages:");
    scanf ("%d", & n);
    int pages[n], frames[framecount];
    printf (" Enter page reference sequence :");
    for (int i =0 ; i<n ; i++){
        scanf ("%d", &pages[i]);
    }
    printf (" FIFO :\n");
    for (int i=0 ;i< framecount; i++){
        frames[i] = -1;
    }
    fifo(pages, n, frames, framecount);

    printf ("LRU: \n");
    for (int i=0 ; i< frame count; i++) {
        frames [i] = -1;
    }
    lru (pages, n, frames, framecount);

    printf (" Optimal: \n");
    for (int i =0; i< framecount ; i++){
        frames [i]=-1;
    }
    optimal (pages , n , frames , framecount);

    return 0;

}
```

→ Output:

```
Enter nor of frames: 3
Enter nor of pages: 7
Enter page reference sequence: 1 3 0 3 5 6 3

FIFO:
The page replacement Process for FIFO is:
1 _ _      PF NO. 1
1 3 _      PF NO. 2
1 3 0      PF NO. 3
1 3 0
5 3 0      PF NO. 4
5 6 0      PF NO. 5
5 6 3      F No. 6

The nor of Page Faults using FIFO are 6

LRU:
The page replacement Process for LRU is
1 _ _     PF NO. 1
1 3 _     PF NO. 2
1 3 0     PF No. 3
1 3 0
5 3 0     PF No. 4
5 3 6     PF No. 5
5 3 6

Page Faults = 5

Optimal:
1 _ _      PF No. 1
3 _ _      PF NO. 2
3 0 _      PF No. 3
```

30 —

35 — PF No. 4

36 — PF No. 5

36 —

Page fault = 5

✓ 11/7/24