ABIRAMI K

230701008 CSE-A

EXP NO:11

EXP 11A- FIFO

PROGRAM

```c
#include <stdio.h>

#define MAX 50

int main() {
    int ref_str[MAX], frames[MAX];
    int ref_len, frame_size;
    int i, j, k, page_faults = 0, found, next = 0;

    // Step 1: Get reference string
    printf("Enter the size of reference string: ");
    scanf("%d", &ref_len);

    for (i = 0; i < ref_len; i++) {
        printf("Enter [%d] : ", i + 1);
        scanf("%d", &ref_str[i]);
    }

    // Step 2: Get frame size
    printf("Enter page frame size: ");
    scanf("%d", &frame_size);

    // Initialize all frames as empty (-1)
    for (i = 0; i < frame_size; i++) {
        frames[i] = -1;
    }

    printf("\nPage Replacement Process:\n");

    // Step 3-6: Process each page in the reference string
    for (i = 0; i < ref_len; i++) {
        found = 0;

        // Check if page is already in frame
        for (j = 0; j < frame_size; j++) {
            if (frames[j] == ref_str[i]) {
                found = 1;
                break;
            }
        }
```

```
        if (!found) {
            // Page fault occurs
            frames[next] = ref_str[i];
            next = (next + 1) % frame_size;  // FIFO: replace oldest
            page_faults++;

            // Print current frame content
            printf("%d -> ", ref_str[i]);
            for (k = 0; k < frame_size; k++) {
                if (frames[k] != -1)
                    printf("%d ", frames[k]);
                else
                    printf("- ");
            }
            printf("\n");
        } else {
            // No page fault
            printf("%d -> No Page Fault\n", ref_str[i]);
        }
    }

    // Step 8: Display total page faults
    printf("\nTotal page faults: %d\n", page_faults);

    return 0;
}
```

OUTPUT

```
Enter the size of reference string:
Enter [1] : 1
Enter [2] : 2
Enter [3] : 3
Enter [4] : 1
Enter [5] : 4
Enter page frame size: 3

Page Replacement Process:
1 -> 1 - -
2 -> 1 2 -
3 -> 1 2 3
1 -> No Page Fault
4 -> 4 2 3

Total page faults: 4
```

EXP 11B LRU

PROGRAM

```c
#include <stdio.h>

#define MAX 50

int main() {
    int frames[MAX], pages[MAX], temp[MAX];
    int i, j, k, n, f, page_faults = 0, flag1, flag2, pos, max,

    // Step 2: Get number of frames
    printf("Enter number of frames: ");
    scanf("%d", &f);

    // Step 3: Get number of pages
    printf("Enter number of pages: ");
    scanf("%d", &n);

    // Step 4: Get reference string
    printf("Enter reference string: ");
    for (i = 0; i < n; i++) {
        scanf("%d", &pages[i]);
    }

    // Step 5: Initialize frame values
    for (i = 0; i < f; i++) {
        frames[i] = -1;
    }

    // Step 6-7: LRU Algorithm
    for (i = 0; i < n; i++) {
        flag1 = flag2 = 0;

        // Check if page is already in frame
        for (j = 0; j < f; j++) {
            if (frames[j] == pages[i]) {
                counter++;
                temp[j] = counter;
                flag1 = flag2 = 1;
                break;
            }
        }

        // If page not found in frame
        if (flag1 == 0) {
```

```
        // If page not found in frame
        if (flag1 == 0) {
            for (j = 0; j < f; j++) {
                if (frames[j] == -1) {
                    counter++;
                    page_faults++;
                    frames[j] = pages[i];
                    temp[j] = counter;
                    flag2 = 1;
                    break;
                }
            }
        }

        // Replace least recently used
        if (flag2 == 0) {
            pos = 0;
            for (j = 1; j < f; j++) {
                if (temp[j] < temp[pos])
                    pos = j;
            }
            counter++;
            page_faults++;
            frames[pos] = pages[i];
            temp[pos] = counter;
        }

        // Step 8: Display current frame status
        for (j = 0; j < f; j++) {
            if (frames[j] != -1)
                printf("%d ", frames[j]);
            else
                printf("-1 ");
        }
        printf("\n");
    }

    // Final Output
    printf("Total Page Faults = %d\n", page_faults);

    return 0;
}
```

OUTPUT

```
Enter number of frames: 3
Enter number of pages: 6
Enter reference string: 5 7 5 6 7 3
5 -1 -1
5 7 -1
5 7 -1
5 7 6
5 7 6
3 7 6
Total Page Faults = 4
```

EXP NO 11C OPTIMAL

PROGRAM

```c
#include <stdio.h>

#define MAX 50

int main() {
    int pages[MAX], frames[MAX];
    int i, j, k, n, f, flag1, flag2, pos, max, page_fault

    // Step 2-4: Input
    printf("Enter number of frames: ");
    scanf("%d", &f);

    printf("Enter number of pages: ");
    scanf("%d", &n);

    printf("Enter reference string: ");
    for (i = 0; i < n; i++) {
        scanf("%d", &pages[i]);
    }

    // Initialize frames
    for (i = 0; i < f; i++) {
        frames[i] = -1;
    }

    // Step 6-7: Optimal Page Replacement Logic
    for (i = 0; i < n; i++) {
        flag1 = flag2 = 0;

        // Check if page is already in frame
        for (j = 0; j < f; j++) {
            if (frames[j] == pages[i]) {
                flag1 = flag2 = 1;
                break;
            }
        }

        // Empty frame found
        if (flag1 == 0) {
            for (j = 0; j < f; j++) {
                if (frames[j] == -1) {
                    frames[j] = pages[i];
                    page_faults++;
```

```
        if (flag1 == 0) {
            for (j = 0; j < f; j++) {
                if (frames[j] == -1) {
                    frames[j] = pages[i];
                    page_faults++;
                    flag2 = 1;
                    break;
                }
            }
        }

        // No empty frame; find optimal victim
        if (flag2 == 0) {
            int farthest = -1;
            pos = -1;

            for (j = 0; j < f; j++) {
                int found = 0;
                for (k = i + 1; k < n; k++) {
                    if (frames[j] == pages[k]) {
                        if (k > farthest) {
                            farthest = k;
                            pos = j;
                        }
                        found = 1;
                        break;
                    }
                }
                if (!found) {
                    pos = j;
                    break;
                }
            }

            frames[pos] = pages[i];
            page_faults++;
        }

        // Display frame status
        for (j = 0; j < f; j++) {
            if (frames[j] != -1)
                printf("%d ", frames[j]);
            else
```

OUTPUT

```
Enter number of frames: 3
Enter number of pages: 10
Enter reference string: 7 0 1 2 0 3 0 4 2 3
7 -1 -1
7 0 -1
7 0 1
2 0 1
2 0 1
2 0 3
2 0 3
2 4 3
2 4 3
2 4 3
Total Page Faults = 6
```