**Name : Omkar Arun Shinde**

**PRN No.: 122B1B258**

**Assignment No. 08**

**Problem Statement :** Write a to implement paging replacement algorithms :

a) FCFS

b) Least Recently Used (LRU)

c) Optimal algorithm

**Code :**

#include <stdio.h>

#include <limits.h>

#define MAX\_FRAMES 10

#define MAX\_PAGES 50

int n, Size;

int isHit(int Frame[], int page) {

    for (int i = 0; i < Size; i++) {

        if (Frame[i] == page)

            return 1;

    }

    return 0;

}

void FCFS(int PageSeq[]) {

    printf("\n--- FCFS Page Replacement ---\n");

    int Frame[MAX\_FRAMES];

    int front = 0, faults = 0;

    for (int i = 0; i < Size; i++) {

        Frame[i] = -1;

    }

    for (int i = 0; i < n; i++) {

        if (!isHit(Frame, PageSeq[i])) {

            faults++;

            Frame[front] = PageSeq[i];

            front = (front + 1) % Size;

        }

        printf("Page %d: ", PageSeq[i]);

        for (int j = 0; j < Size; j++) {

            Frame[j] == -1 ? printf("- ") : printf("%d ", Frame[j]);

        }

        printf("\n");

    }

    printf("Total Page Faults (FCFS): %d\n", faults);

    printf("Total Page Hits (FCFS): %d\n", n - faults);

    printf("Hit Ratio: %.2f%%\n", ((float)(n - faults) / n) \* 100);

}

void LRU(int PageSeq[]) {

    printf("\n--- LRU Page Replacement ---\n");

    int Frame[MAX\_FRAMES];

    int count[MAX\_FRAMES] = {0};

    int Time = 0, faults = 0;

    for (int i = 0; i < Size; i++) {

        Frame[i] = -1;

    }

    for (int i = 0; i < n; i++) {

        Time++;

        int hit = 0;

        for (int j = 0; j < Size; j++) {

            if (Frame[j] == PageSeq[i]) {

                hit = 1;

                count[j] = Time;

                break;

            }

        }

        if (!hit) {

            faults++;

            int min = INT\_MAX, replace\_index = -1;

            for (int j = 0; j < Size; j++) {

                if (Frame[j] == -1) {

                    replace\_index = j;

                    break;

                } else if (count[j] < min) {

                    min = count[j];

                    replace\_index = j;

                }

            }

            Frame[replace\_index] = PageSeq[i];

            count[replace\_index] = Time;

        }

        printf("Page %d: ", PageSeq[i]);

        for (int j = 0; j < Size; j++) {

            Frame[j] == -1 ? printf("- ") : printf("%d ", Frame[j]);

        }

        printf("\n");

    }

    printf("Total Page Faults (LRU): %d\n", faults);

    printf("Total Page Hits (LRU): %d\n", n - faults);

    printf("Hit Ratio: %.2f%%\n", ((float)(n - faults) / n) \* 100);

}

int predict(int PageSeq[], int Frame[], int index) {

    int Far = -1, Found = -1;

    for (int i = 0; i < Size; i++) {

        int j;

        for (j = index; j < n; j++) {

            if (Frame[i] == PageSeq[j]) {

                if (j > Far) {

                    Far = j;

                    Found = i;

                }

                break;

            }

        }

        if (j == n)

            return i;

    }

    return (Found == -1) ? 0 : Found;

}

void Optimal(int PageSeq[]) {

    printf("\n--- Optimal Page Replacement ---\n");

    int Frame[MAX\_FRAMES];

    int faults = 0;

    for (int i = 0; i < Size; i++) {

        Frame[i] = -1;

    }

    for (int i = 0; i < n; i++) {

        if (!isHit(Frame, PageSeq[i])) {

            faults++;

            int j;

            for (j = 0; j < Size; j++) {

                if (Frame[j] == -1) {

                    Frame[j] = PageSeq[i];

                    break;

                }

            }

            if (j == Size) {

                int idx = predict(PageSeq, Frame, i + 1);

                Frame[idx] = PageSeq[i];

            }

        }

        printf("Page %d: ", PageSeq[i]);

        for (int j = 0; j < Size; j++) {

            Frame[j] == -1 ? printf("- ") : printf("%d ", Frame[j]);

        }

        printf("\n");

    }

    printf("Total Page Faults (Optimal): %d\n", faults);

    printf("Total Page Hits (Optimal): %d\n", n - faults);

    printf("Hit Ratio: %.2f%%\n", ((float)(n - faults) / n) \* 100);

}

int main() {

    int PageSeq[MAX\_PAGES], choice;

    printf("Enter Number of Pages: ");

    scanf("%d", &n);

    printf("Enter The Page Reference String:\n");

    for (int i = 0; i < n; i++)

        scanf("%d", &PageSeq[i]);

    printf("Enter The Number of Frames: ");

    scanf("%d", &Size);

    do {

        printf("\nChoose Paging Algorithm:\n");

        printf("1. FCFS\n");

        printf("2. LRU\n");

        printf("3. Optimal\n");

        printf("4. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                FCFS(PageSeq);

                break;

            case 2:

                LRU(PageSeq);

                break;

            case 3:

                Optimal(PageSeq);

                break;

            case 4:

                printf("Exiting program.\n");

                break;

            default:

                printf("Invalid choice! Try again.\n");

        }

    } while (choice != 4);

    return 0;

}

**Output :**