Name-Nissi Rathod

Class-TE-COM .B

Roll NO.-64

ASSIGNMENT NO.5

CODE :

```
#include <bits/stdc++.h>
using namespace std;

// ---------- FIFO ----------
void fifo(vector<int> pages, int f) {
   queue<int> q;
   unordered_set<int> s;
   int faults = 0;
   cout << "\n--- FIFO Page Replacement ---\n";
   for (int p : pages) {
      if (s.size() < f) {
         if (s.find(p) == s.end()) {
            s.insert(p);
            q.push(p);
            faults++;
         }
      } else {
         if (s.find(p) == s.end()) {
            int old = q.front(); q.pop();
            s.erase(old);
```

```
                s.insert(p);
                q.push(p);
                faults++;
            }
        }
        for (int x : s) cout << x << " ";
        cout << endl;
    }
    cout << "Total Page Faults = " << faults << "\n";
}

// ---------- LRU ----------
void lru(vector<int> pages, int f) {
    list<int> dq;
    unordered_map<int, list<int>::iterator> pos;
    int faults = 0;
    cout << "\n--- LRU Page Replacement ---\n";
    for (int p : pages) {
        if (pos.find(p) == pos.end()) {
            faults++;
            if (dq.size() == f) {
                int last = dq.back(); dq.pop_back();
                pos.erase(last);
            }
        } else dq.erase(pos[p]);
        dq.push_front(p);
        pos[p] = dq.begin();
        for (int x : dq) cout << x << " ";
        cout << endl;
    }
    cout << "Total Page Faults = " << faults << "\n";
}

// ---------- OPTIMAL ----------
void optimal(vector<int> pages, int f) {
    vector<int> frames;
    int faults = 0;
    cout << "\n--- Optimal Page Replacement ---\n";
    for (int i = 0; i < pages.size(); i++) {
        int p = pages[i];
        if (find(frames.begin(), frames.end(), p) == frames.end()) {
            faults++;
            if (frames.size() < f) frames.push_back(p);
            else {
                int farthest = i + 1, idx = -1;
                for (int j = 0; j < frames.size(); j++) {
                    int next = INT_MAX;
                    for (int k = i + 1; k < pages.size(); k++)
                        if (frames[j] == pages[k]) { next = k; break; }
                    if (next > farthest) { farthest = next; idx = j; }
                }
                if (idx == -1) idx = 0;
                frames[idx] = p;
            }
        }
        for (int x : frames) cout << x << " ";
        cout << endl;
    }
    cout << "Total Page Faults = " << faults << "\n";
}

// ---------- MAIN ----------
int main() {
    int n, f;
    cout << "Enter number of pages: ";
    cin >> n;
    vector<int> pages(n);
    cout << "Enter page reference string:\n";
    for (int i = 0; i < n; i++) cin >> pages[i];
    cout << "Enter number of frames: ";
    cin >> f;

    fifo(pages, f);
    lru(pages, f);
    optimal(pages, f);
}
```

OUTPUT:

```
--- FIFO Page Replacement ---
```

```
1
3 1
0 3 1
5 0 3
6 5 0
3 6 5
1 3 6
3 1 3
6 3 1
3 6 3
2 3 6
Total Page Faults = 9

--- LRU Page Replacement ---
1
3 1
0 3 1
5 0 3
6 5 0
3 6 5
1 3 6
3 1 3
6 3 1
```

```
3 6 3
2 3 6
Total Page Faults = 9

--- Optimal Page Replacement ---
1
3 1
0 3 1
5 0 3
6 5 0
3 6 5
1 3 6
3 1 3
6 3 1
3 6 3
2 3 6
Total Page Faults = 7
```