Program:

```
#include <iostream>

#include <queue>

using namespace std;

 int FIFO(int pages[], int n, int frames) {

    queue<int> q; // Queue to store pages in memory

    int pageFaults = 0; // Counter for page faults


    for (int i = 0; i < n; i++) {

        // Check if the page is already in memory

        bool found = false;

        queue<int> temp = q;

        while (!temp.empty()) {

            if (temp.front() == pages[i]) {

                found = true;

                break;

            }

            temp.pop();

        }


        // If page is not in memory, replace the oldest page

        if (!found) {

            if (q.size() == frames) {

                q.pop(); // Remove the oldest page

            }

            q.push(pages[i]); // Add the new page

            pageFaults++; // Increment page fault counter

        }

    }
    return pageFaults;
}

int main() {
    int n, frames;

    cout << "Enter the number of pages: ";
    cin >> n;

    int pages[n];
    cout << "Enter the page reference string: ";
    for (int i = 0; i < n; i++) {
        cin >> pages[i];
    }

    cout << "Enter the number of frames: ";
    cin >> frames;
    int pageFaults = FIFO(pages, n, frames);
    cout << "Total Page Faults: " << pageFaults << endl;

    return 0;
}
```

Output:

```
Enter the number of pages: 20
Enter the page reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
Enter the number of frames: 3
Total Page Faults: 15
```

Conclusion: First In First Out Page Replacement Algorithm program was implemented

successfully.