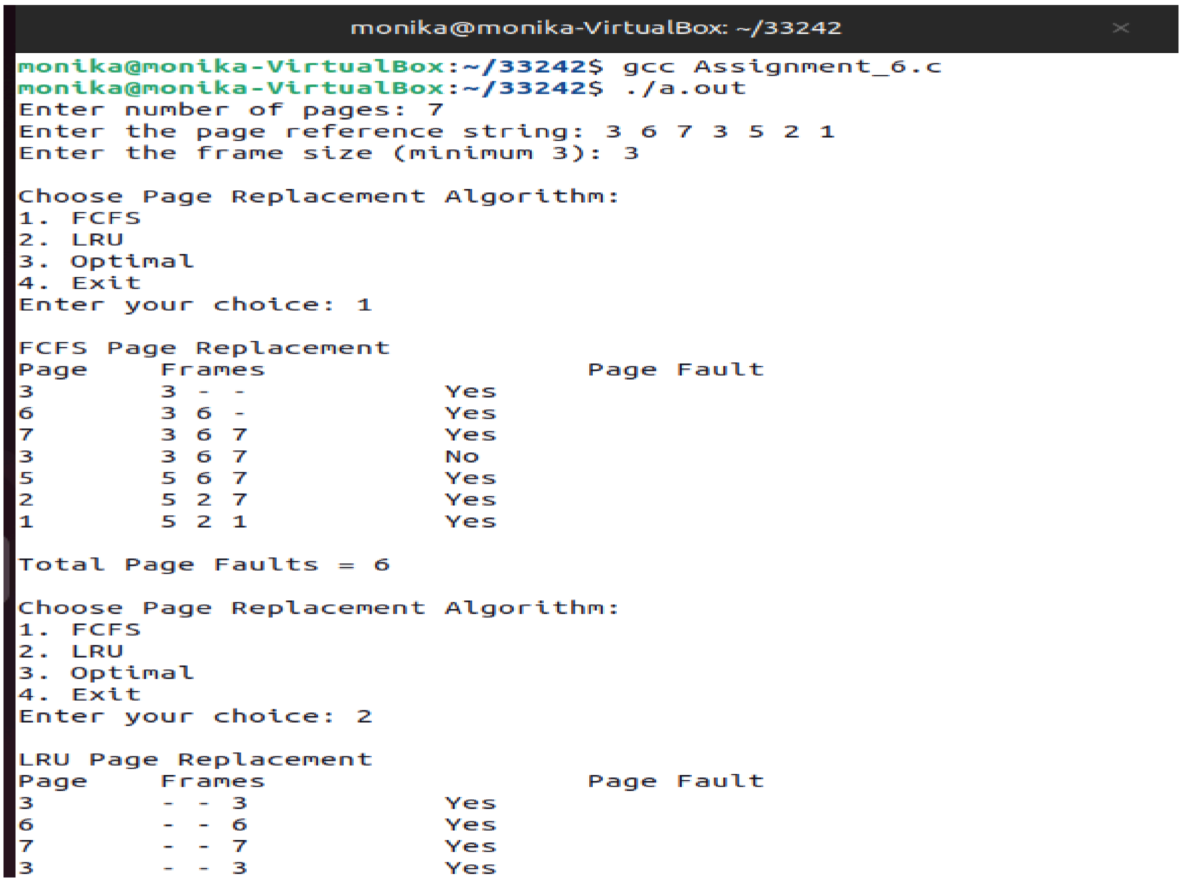
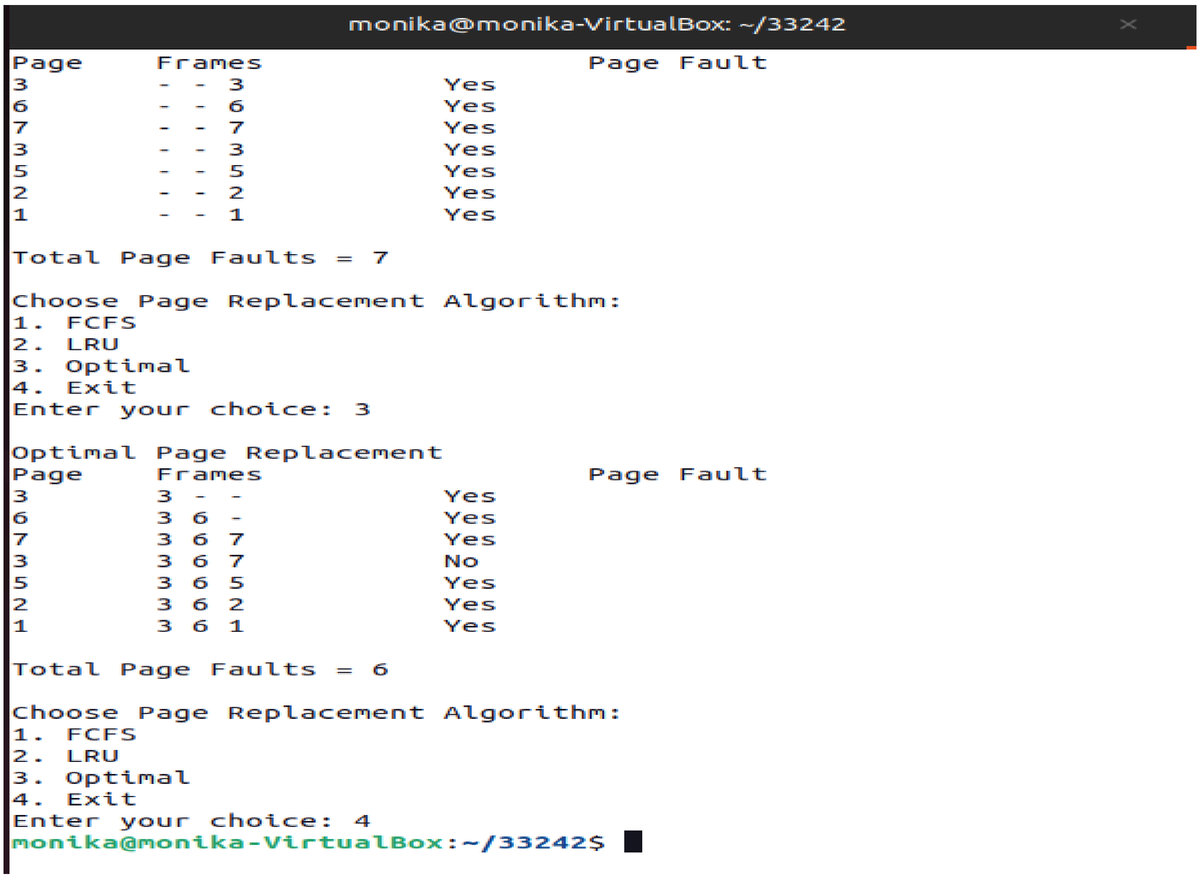
OS Assignment 6

Code:
```
#include <stdio.h>

#define MAX 20

// Function to display the frame content
void display(int frames[], int n) {
    for (int i = 0; i < n; i++) {
        if (frames[i] == -1)
            printf("- ");
        else
            printf("%d ", frames[i]);
    }
}

// FCFS Page Replacement Algorithm
void fcfs(int pages[], int n, int frames[], int frameSize) {
    int index = 0, pageFaults = 0;

    printf("\nFCFS Page Replacement\n");
    printf("Page\tFrames\t\t\tPage Fault\n");

    for (int i = 0; i < frameSize; i++) {
        frames[i] = -1;
    }

    for (int i = 0; i < n; i++) {
        int page = pages[i];
        int found = 0;

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

        // If not found, replace the oldest page (FCFS)
        if (!found) {
            frames[index] = page;
            index = (index + 1) % frameSize;
            pageFaults++;
            printf("%d\t", page);
            display(frames, frameSize);
            printf("\t\tYes\n");
        } else {
            printf("%d\t", page);
            display(frames, frameSize);
            printf("\t\tNo\n");
        }
    }

    printf("\nTotal Page Faults = %d\n", pageFaults);
}

// LRU Page Replacement Algorithm
void lru(int pages[], int n, int frames[], int frameSize) {
    int pageFaults = 0;
    int used[MAX];

    printf("\nLRU Page Replacement\n");
    printf("Page\tFrames\t\t\tPage Fault\n");

    for (int i = 0; i < frameSize; i++) {
        frames[i] = -1;
    }

    for (int i = 0; i < n; i++) {
        int page = pages[i];
        int found = 0;

        // Check if page is already in frame
        for (int j = 0; j < frameSize; j++) {
            if (frames[j] == page) {
                found = 1;
                used[j] = i;  // Update last used time
                break;
            }
        }

        // If not found, replace the least recently used page
        if (!found) {
            int lruIndex = 0;
            for (int j = 1; j < frameSize; j++) {
                if (frames[j] == -1 || used[j] < used[lruIndex]) {
                    lruIndex = j;
                }
            }
            frames[lruIndex] = page;
            used[lruIndex] = i;
            pageFaults++;
            printf("%d\t", page);
            display(frames, frameSize);
            printf("\t\tYes\n");
        } else {
            printf("%d\t", page);
            display(frames, frameSize);
            printf("\t\tNo\n");
        }
    }

    printf("\nTotal Page Faults = %d\n", pageFaults);
}

// Optimal Page Replacement Algorithm
void optimal(int pages[], int n, int frames[], int frameSize) {
    int pageFaults = 0;

    printf("\nOptimal Page Replacement\n");
    printf("Page\tFrames\t\t\tPage Fault\n");

    for (int i = 0; i < frameSize; i++) {
        frames[i] = -1;
    }

    for (int i = 0; i < n; i++) {
        int page = pages[i];
        int found = 0;

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

        // If not found, replace the page that won't be used for the
longest time
        if (!found) {
            int farthest = i + 1, replaceIndex = -1;
            for (int j = 0; j < frameSize; j++) {
                if (frames[j] == -1) {
                    replaceIndex = j;
                    break;
                }
                int nextUse = -1;
                for (int k = i + 1; k < n; k++) {
                    if (pages[k] == frames[j]) {
                        nextUse = k;
                        break;
                    }
                }
                if (nextUse == -1 || nextUse > farthest) {
                    farthest = nextUse;
                    replaceIndex = j;
                }
            }
            frames[replaceIndex] = page;
            pageFaults++;
            printf("%d\t", page);
            display(frames, frameSize);
            printf("\t\tYes\n");
        } else {
            printf("%d\t", page);
            display(frames, frameSize);
            printf("\t\tNo\n");
        }
    }

    printf("\nTotal Page Faults = %d\n", pageFaults);
}

int main() {
    int pages[MAX], frames[MAX], n, frameSize, choice;

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

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

    printf("Enter the frame size (minimum 3): ");
    scanf("%d", &frameSize);

    if (frameSize < 3) {
        printf("Frame size should be at least 3.\n");
        return 0;
    }

    while (1) {
        printf("\nChoose Page Replacement Algorithm:\n");
        printf("1. FCFS\n2. LRU\n3. Optimal\n4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                fcfs(pages, n, frames, frameSize);
                break;
            case 2:
                lru(pages, n, frames, frameSize);
                break;
            case 3:
                optimal(pages, n, frames, frameSize);
                break;
            case 4:
                return 0;
            default:
                printf("Invalid choice!\n");
        }
    }

    return 0;
}
```

Output



```
monika@monika-VirtualBox: ~/33242
monika@monika-VirtualBox:~/33242$ gcc Assignment_6.c
monika@monika-VirtualBox:~/33242$ ./a.out
Enter number of pages: 7
Enter the page reference string: 3 6 7 3 5 2 1
Enter the frame size (minimum 3): 3

Choose Page Replacement Algorithm:
1. FCFS
2. LRU
3. Optimal
4. Exit
Enter your choice: 1

FCFS Page Replacement
Page    Frames                  Page Fault
3       3 - -           Yes
6       3 6 -           Yes
7       3 6 7           Yes
3       3 6 7           No
5       5 6 7           Yes
2       5 2 7           Yes
1       5 2 1           Yes

Total Page Faults = 6

Choose Page Replacement Algorithm:
1. FCFS
2. LRU
3. Optimal
4. Exit
Enter your choice: 2

LRU Page Replacement
Page    Frames                  Page Fault
3       - - 3           Yes
6       - - 6           Yes
7       - - 7           Yes
3       - - 3           Yes
```



```
monika@monika-VirtualBox: ~/33242
Page    Frames                  Page Fault
3       - - 3           Yes
6       - - 6           Yes
7       - - 7           Yes
3       - - 3           Yes
5       - - 5           Yes
2       - - 2           Yes
1       - - 1           Yes

Total Page Faults = 7

Choose Page Replacement Algorithm:
1. FCFS
2. LRU
3. Optimal
4. Exit
Enter your choice: 3

Optimal Page Replacement
Page    Frames                  Page Fault
3       3 - -           Yes
6       3 6 -           Yes
7       3 6 7           Yes
3       3 6 7           No
5       3 6 5           Yes
2       3 6 2           Yes
1       3 6 1           Yes

Total Page Faults = 6

Choose Page Replacement Algorithm:
1. FCFS
2. LRU
3. Optimal
4. Exit
Enter your choice: 4
monika@monika-VirtualBox:~/33242$
```