11a.

CODE-

```python
def fifo_page_replacement(ref_string, frame_size):
    memory = []
    page_faults = 0

    print("\nPage replacement process:\n")

    for i, page in enumerate(ref_string):
        print(f"{page} -> ", end="")
        if page not in memory:
            if len(memory) < frame_size:
                memory.append(page)
            else:
                memory.pop(0)
                memory.append(page)
            page_faults += 1
            print(" ".join(map(str, memory)) + (" -" * (frame_size - len
                (memory))))
        else:
            print("No Page Fault")

    print(f"\nTotal page faults: {page_faults}.")

# Main driver
if __name__ == "__main__":
    n = int(input("Enter the size of reference string: "))
    ref_string = []
    for i in range(n):
        ref = int(input(f"Enter [{i + 1}] : "))
        ref_string.append(ref)

    frame_size = int(input("Enter page frame size : "))
    fifo_page_replacement(ref_string, frame_size)
```

OUTPUT-

```
Enter the size of reference string: 20
Enter [1] : 7
Enter [2] : 0
Enter [3] : 1
Enter [4] : 2
Enter [5] : 0
Enter [6] : 3
Enter [7] : 0
Enter [8] : 4
Enter [9] : 2
Enter [10] : 3
Enter [11] : 0
Enter [12] : 3
Enter [13] : 2
Enter [14] : 1
Enter [15] : 2
Enter [16] : 0
Enter [17] : 1
Enter [18] : 7
Enter [19] : 0
Enter [20] : 1
Enter page frame size : 3

Page replacement process:

7 -> 7 - -
0 -> 7 0 -
1 -> 7 0 1
2 -> 0 1 2
0 -> No Page Fault
3 -> 1 2 3
0 -> 2 3 0
4 -> 3 0 4
2 -> 0 4 2
3 -> 4 2 3
0 -> 2 3 0
3 -> No Page Fault
2 -> No Page Fault
1 -> 3 0 1
2 -> 0 1 2
0 -> No Page Fault
1 -> No Page Fault
7 -> 1 2 7
0 -> 2 7 0
1 -> 7 0 1

Total page faults: 15.
```

11.b

CODE-

```c
#include <stdio.h>
int findLRU(int time[], int n) {
    int i, minimum = time[0], pos = 0;
    for(i = 1; i < n; ++i) {
        if(time[i] < minimum) {
            minimum = time[i];
            pos = i;
        }
    }
    return pos;
}
int main() {
    int frames, pages, count = 0, time[10], faults = 0;
    int frame[10], ref[30];
    int i, j, k, pos, flag1, flag2;
    printf("Enter number of frames: ");
    scanf("%d", &frames);
    printf("Enter number of pages: ");
    scanf("%d", &pages);
    printf("Enter reference string: ");
    for(i = 0; i < pages; ++i) {
        scanf("%d", &ref[i]);
    }
    for(i = 0; i < frames; ++i) {
        frame[i] = -1;
    }
    for(i = 0; i < pages; ++i) {
        flag1 = flag2 = 0;
        for(j = 0; j < frames; ++j) {
            if(frame[j] == ref[i]) {
                count++;
                time[j] = count;
                flag1 = flag2 = 1;
                break;
            }
        }
        if(flag1 == 0) {
            for(j = 0; j < frames; ++j) {
                if(frame[j] == -1) {
                    count++;
                    faults++;
                    frame[j] = ref[i];
                    time[j] = count;
                    flag2 = 1;
                    break;
                }
            }
        }
```

```c
        if(flag2 == 0) {
            pos = findLRU(time, frames);
            count++;
            faults++;
            frame[pos] = ref[i];
            time[pos] = count;
        }
        for(j = 0; j < frames; ++j) {
            if(frame[j] != -1)
                printf("%d ", frame[j]);
            else
                printf("-1 ");
        }
        printf("\n");
    }
    printf("Total Page Faults = %d\n", 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
```

11.c

CODE-

```c
#include <stdio.h>

int predict(int pages[], int frames[], int n, int index, int frameSize) {
    int res = -1, farthest = index;

    for (int i = 0; i < frameSize; i++) {
        int j;
        for (j = index; j < n; j++) {
            if (frames[i] == pages[j]) {
                if (j > farthest) {
                    farthest = j;
                    res = i;
                }
                break;
            }
        }

        // If page not used in future
        if (j == n)
            return i;
    }

    return (res == -1) ? 0 : res;
}

int main() {
    int frames[10], pages[30], frameSize, n, pageFaults = 0;
    int i, j, k, found;

    printf("Enter number of frames: ");
    scanf("%d", &frameSize);

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

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

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

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

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

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

        // If not found (Page Fault)
        if (!found) {
            pageFaults++;

            int pos = -1;
            for (j = 0; j < frameSize; j++) {
                if (frames[j] == -1) {
                    pos = j;
                    break;
                }
            }

            if (pos == -1) {
                pos = predict(pages, frames, n, i + 1, frameSize);
            }

            frames[pos] = pages[i];
        }

        // Print current frame status
        for (k = 0; k < frameSize; k++) {
            if (frames[k] != -1)
                printf("%d ", frames[k]);
            else
                printf("-1 ");
        }
        printf("\n");
    }

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

    return 0;
}
```

OUTPUT-

```
Enter number of frames: 3
Enter number of pages: 6
Enter reference string: 7 0 1 2 0 3
7 -1 -1
7 0 -1
7 0 1
2 0 1
2 0 1
3 0 1
Total Page Faults = 5
```