Kamalesh S P
230701138
II-CSE-B

11a) FIFO

Program Code:

```
from collections import deque

# Input the reference string
ref_len = int(input("Enter the size of reference string: "))
reference = []

for i in range(ref_len):
    value = int(input(f"Enter [{i+1}] : "))
    reference.append(value)

# Input frame size
frame_size = int(input("Enter page frame size : "))

# Initialize queue and other variables
frames = deque()
page_faults = 0

print()  # For spacing

for i in reference:
    if i not in frames:
        if len(frames) < frame_size:
            frames.append(i)
        else:
            frames.popleft()
            frames.append(i)
        page_faults += 1
        print(f"{i} ->", end=" ")
        for f in frames:
            print(f, end=" ")
        for _ in range(frame_size - len(frames)):
            print("-", end=" ")
        print()
    else:
        print(f"{i} -> No Page Fault")

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

Output:

Kamalesh S P
230701138
II-CSE-B

CS23431-Operating System

```
PS C:\Users\kamal\OneDrive\Desktop\program\OS program> python fifo.py
Enter the size of reference string: 5
Enter [1] : 7
Enter [2] : 0
Enter [3] : 1
Enter [4] : 2
Enter [5] : 0
Enter page frame size : 2

7 -> 7 -
0 -> 7 0
1 -> 0 1
2 -> 1 2
0 -> 2 0

Total page faults: 5
```

Kamalesh S P
230701138
II-CSE-B

11b) LRU

Program 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[10], pages[30], time[10];
    int totalFrames, totalPages, counter = 0, pageFaults = 0;
    int i, j, flag1, flag2, pos;
    printf("Enter number of frames: ");
    scanf("%d", &totalFrames);
    printf("Enter number of pages: ");
    scanf("%d", &totalPages);
    printf("Enter reference string: ");
    for(i = 0; i < totalPages; ++i) {
        scanf("%d", &pages[i]);
    }
    for(i = 0; i < totalFrames; ++i) {
        frames[i] = -1;
    }
    printf("\n");
    for(i = 0; i < totalPages; ++i) {
        flag1 = flag2 = 0;
        for(j = 0; j < totalFrames; ++j) {
            if(frames[j] == pages[i]) {
                counter++;
                time[j] = counter;
                flag1 = flag2 = 1;
                break;
            }
        }
        if(flag1 == 0) {
            for(j = 0; j < totalFrames; ++j) {
                if(frames[j] == -1) {
                    counter++;
                    pageFaults++;
                    frames[j] = pages[i];
                    time[j] = counter;
                    flag2 = 1;
                    break;
                }
            }
        }
        if(flag2 == 0) {
            pos = findLRU(time, totalFrames);
            counter++;
            pageFaults++;
            frames[pos] = pages[i];
            time[pos] = counter;
        }
        for(j = 0; j < totalFrames; ++j) {
            printf("%d ", frames[j]);
        }
        printf("\n");
    }
    printf("Total Page Faults = %d\n", pageFaults);
    return 0;
}
```

Output:

Kamalesh S P
230701138
II-CSE-B

CS23431-Operating System

```
PS C:\Users\kamal\OneDrive\Desktop\program\OS program> ./lru.exe
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
```

Kamalesh S P
230701138
II-CSE-B

11c) Optimal

Program Code:

```c
#include <stdio.h>
int predict(int pages[], int frames[], int totalPages, int totalFrames, int index) {
    int pos = -1, farthest = index;
    for (int i = 0; i < totalFrames; i++) {
        int j;
        for (j = index; j < totalPages; j++) {
            if (frames[i] == pages[j]) {
                if (j > farthest) {
                    farthest = j;
                    pos = i;
                }
                break;
            }
        }
        if (j == totalPages)
            return i;
    }
    return (pos == -1) ? 0 : pos;
}
int main() {
    int pages[100], frames[10], totalPages, totalFrames;
    int pageFaults = 0, hit;
    printf("Enter number of frames: ");
    scanf("%d", &totalFrames);
    printf("Enter number of pages: ");
    scanf("%d", &totalPages);
    printf("Enter reference string: ");
    for (int i = 0; i < totalPages; i++) {
        scanf("%d", &pages[i]);
    }
    for (int i = 0; i < totalFrames; i++) {
        frames[i] = -1;
    }
    printf("\n");
    for (int i = 0; i < totalPages; i++) {
        hit = 0;
        for (int j = 0; j < totalFrames; j++) {
            if (frames[j] == pages[i]) {
                hit = 1;
                break;
            }
        }
        if (!hit) {
            int replaced = 0;
            for (int j = 0; j < totalFrames; j++) {
                if (frames[j] == -1) {
                    frames[j] = pages[i];
                    replaced = 1;
                    break;
                }
            }
            if (!replaced) {
                int pos = predict(pages, frames, totalPages, totalFrames, i + 1);
                frames[pos] = pages[i];
            }
            pageFaults++;
        }
        for (int j = 0; j < totalFrames; j++) {
            printf("%d ", frames[j]);
        }
        printf("\n");
    }
    printf("Total Page Faults = %d\n", pageFaults);
    return 0;
}
```

Output:

Kamalesh S P
230701138
II-CSE-B

CS23431-Operating System

```
PS C:\Users\kamal\OneDrive\Desktop\program\OS program> ./optimal.exe
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
```