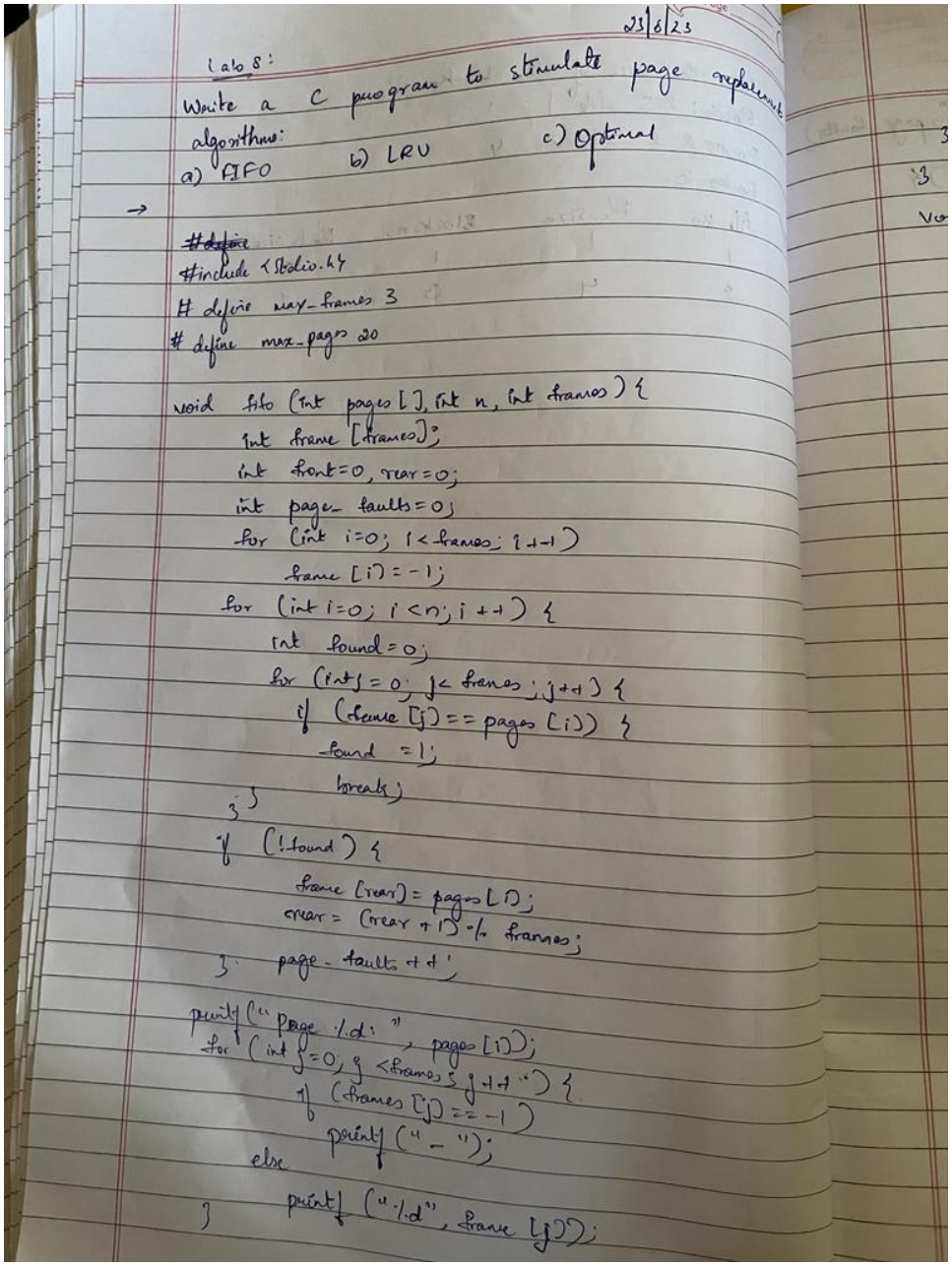
LAB 9

Write a C program to simulate page replacement algorithms

a) FIFO

b) LRU

c) Optimal



23/6/23

Lab 8:

Write a C program to stimulate page replacement algorithms:

a) FIFO b) LRU c) Optimal

```
#include <stdio.h>
# define max_frames 3
# define max_pages 20

void fifo (int pages[], int n, int frames) {
    int frame[frames];
    int front=0, rear=0;
    int page_faults=0;
    for (int i=0; i<frames; i++)
        frame[i] = -1;
    for (int i=0; i<n; i++) {
        int found=0;
        for (int j=0; j<frames; j++) {
            if (frame[j] == pages[i]) {
                found = 1;
                break;
            }
        }
        if (!found) {
            frame[rear] = pages[i];
            rear = (rear+1) % frames;
            page_faults++;
        }
        printf("Page %d: ", pages[i]);
        for (int j=0; j<frames; j++) {
            if (frames[j] == -1)
                printf("_ ");
            else
                printf("%d", frame[j]);
        }
```

```
    printf("\n");
}
printf("Total page faults (FIFO) : %d \n", page_faults);

void lru(int pages[], int n, int frames){
    int frame[frames];
    int page_faults = 0;
    int used[max_pages] = {0};
    for (int i=0; i<frames; i++)
        frame[i] = -1;
    for (int i = 0; i<n; i++) {
        int found = 0;
        for (int j = 0; j< frames; j++) {
            if (frame[j] == pages[i]) {
                found = 1;
                used[j] = i;
                break;
            }
        }
        if (!found) {
            int min = 0;
            for (int j = 1; j < frames; j++) {
                if (used[j] < used[min])
                    min = j;
            }
            frame[min] = pages[i];
            used[min] = i;
            page_faults++;
        }
        printf("Page %d: ", pages[i]);
        for (int j=0; j< frames; j++) {
            if (frame[j] == -1)
                printf("_ ");
            else
                printf("%d ", frame[j]);
        }
        printf("\n");
    }
```

```
printf("Total page faults (LRU) : %d \n", page_faults);
}
void optimal(int pages[], int n, int frames) {
    int frame[frames];
    int page_faults = 0;
    for(int i=0; i<frames; i++)
        frame[i] = -1;
    for(int i=0; i<n; i++) {
        int found = 0;
        for(int j=0; j<frames; j++) {
            if(frame[j] == pages[i]) {
                found = 1;
                break;
            }
        }
        if(!found) {
            if(i < frames)
                frame[i] = pages[i];
            else {
                int max_dist = -1;
                int replace_page = -1;
                for(int j=0; j<frames; j++) {
                    int dist = MAX_PAGES;
                    for(int k=i+1; k<n; k++) {
                        if(pages[k] == frame[j]) {
                            dist = k-i;
                            break;
                        }
                    }
                    if(dist > max_dist) {
                        max_dist = dist;
                        replace_page = j;
                    }
                }
                frame[replace_page] = pages[i];
            }
            page_faults++;
        }
```

```
- faults);
        printf("Page %d", pages[i]);
        for (int j=0; j<frames; j++) {
            if (frame[j] == -1)
                printf("- ");
            else
                printf("%d", frame[j]);
        }
        printf("\n");
    }
    printf("Total page faults (optimal): %d\n", page_faults);
}

int main() {
    int pages[max_pages];
    int n, frames;
    printf("Enter the number of pages: ");
    scanf("%d", &n);
    printf("Enter the reference string: ");
    for (int i=0; i<n; i++)
        scanf("%d", &pages[i]);
    printf("Enter the no. of frames: ");
    scanf("%d", &frames);
    printf("Enter the reference string: ");
    for (int i=0; i<n; i++)
        scanf("%d", &pages[i]);
    printf("Enter the number of frames: ");
    scanf("%d", &frames);
    printf("Enter the number of frames: ");
    scanf("%d", &frames);
    printf("Enter FIFO page Replacement:\n");
    fifo(pages, n, frames);
    printf("LRU Page replacement:\n");
    lru(pages, n, frames);
    printf("Optimal Page Replacement:\n");
```

```
    optimal (pages, n, frames);
    return 0;
}
```

Output:

Enter the number of pages: 14

Enter page no.s: 0 4 3 2 1 4 6 3 0 8 9 3 8 3

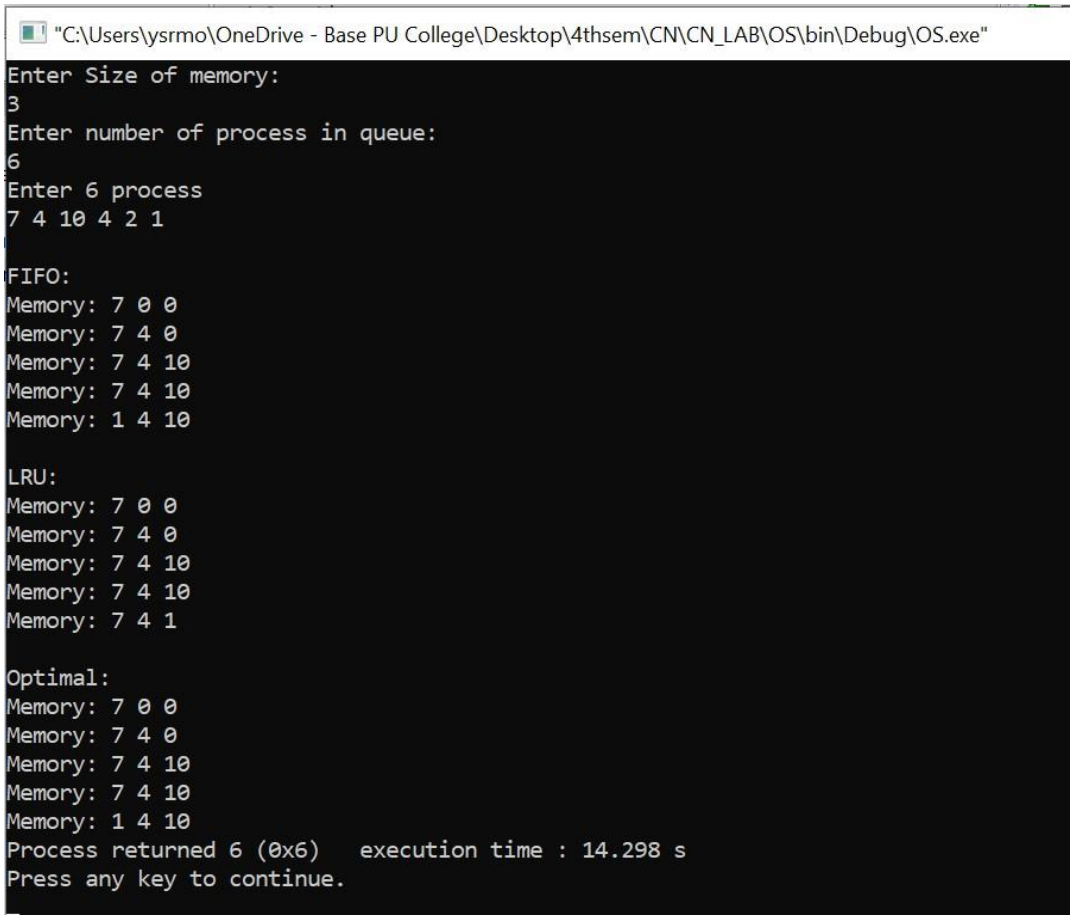
Enter the no. of frames: 3

FIFO - page fault = 13

optimal - page fault = 16

LRU - page fault = 13

OUTPUT:


"C:\Users\ysrmo\OneDrive - Base PU College\Desktop\4thsem\CN\CN_LAB\OS\bin\Debug\OS.exe"
Enter Size of memory:
3
Enter number of process in queue:
6
Enter 6 process
7 4 10 4 2 1
FIFO:
Memory: 7 0 0
Memory: 7 4 0
Memory: 7 4 10
Memory: 7 4 10
Memory: 1 4 10
LRU:
Memory: 7 0 0
Memory: 7 4 0
Memory: 7 4 10
Memory: 7 4 10
Memory: 7 4 1
Optimal:
Memory: 7 0 0
Memory: 7 4 0
Memory: 7 4 10
Memory: 7 4 10
Memory: 1 4 10
Process returned 6 (0x6) execution time : 14.298 s
Press any key to continue.