WEEK 10

Write a C program to simulate page replacement algorithms

a) FIFO

b) LRU

c) Optimal

CODE :

```
#include<stdio.h>
void main()
{
   int mem[20],process[20],n,m,i,j,k,c,z,a,distance=0,b;
   printf("Enter Size of memory:\n");
   scanf("%d",&n);
   for(i=0;i<n;i++)
        mem[i]=0;
   printf("Enter number of process in queue:\n");
   scanf("%d",&m);
   printf("Enter %d process \n",m);
   for(i=0;i<m;i++)
        scanf("%d",&process[i]);
   j=0;
   i=0;
   printf("\nFIFO:");
   while(j!=m)
   {
        k=0;
```

```
        c=0;
        while(k!=n)
        {
        c++;
        if(mem[k]==process[j])
        {
                j++;
                break;
        }
        k++;
        }
        if(c==n)
        {
        mem[i]=process[j];
        i=(i+1)%n;
        }
        printf("\nMemory: ");
        for(z=0;z<n;z++)
        printf("%d ",mem[z]);
        j++;
}
printf("\nLRU:");
for(i=0;i<n;i++)
        mem[i]=0;
i=0;
j=0;
while(j!=m)
```

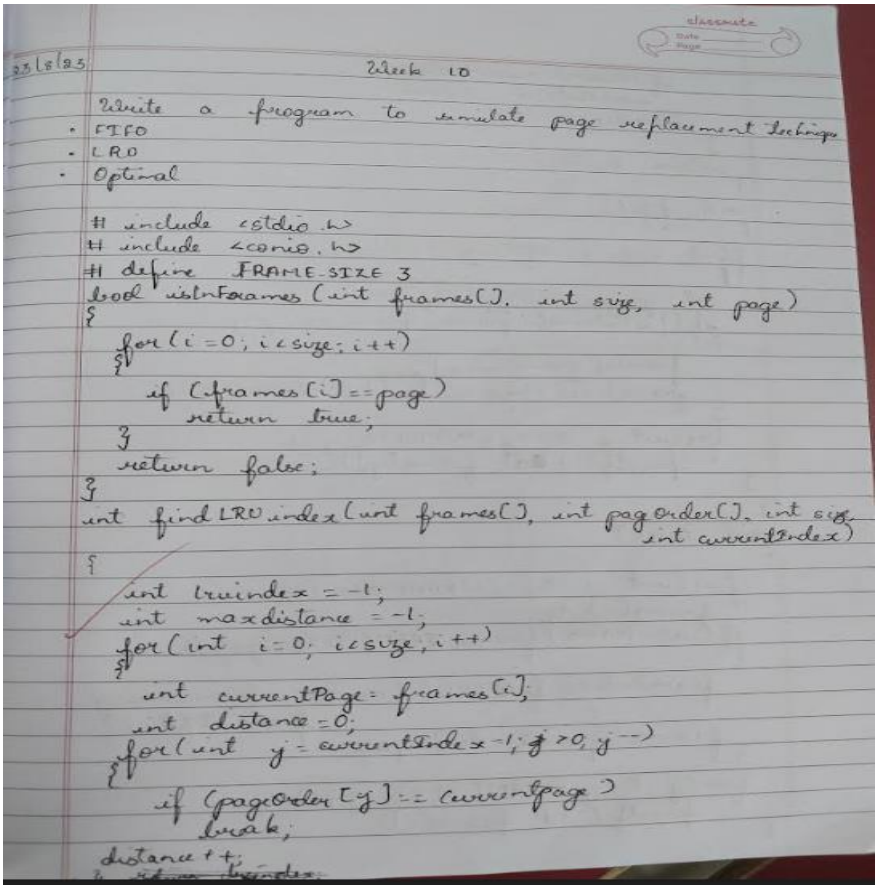
```
{
        k=0;
        c=0;
        while(k!=n)
        {
        c++;
        if(mem[k]==process[j])
        {
                j++;
                break;
        }
        k++;
        }
        if(c==n)
        {
        distance=0;
        for(a=0;a<n;a++)
        {
                b=99;
                z=j;
                while(z>=0)
                {
                if((j-z)>distance)
                if(mem[a]==process[z])
                {
                distance=(z-j);
                b=z;
```

```
                }
                z--;
                }
        }
        if(b==99)
        b=i;
        mem[b]=process[j];
        i=(i+1)%n;
        }
        printf("\nMemory: ");
        for(z=0;z<n;z++)
        printf("%d ",mem[z]);
        j++;
}
printf("\n\nOptimal:");
for(i=0;i<n;i++)
        mem[i]=0;
i=0;
j=0;
while(j!=m)
{
        k=0;
        c=0;
        while(k!=n)
        {
        c++;
        if(mem[k]==process[j])
```

```
{
        j++;
        break;
}
k++;
}
if(c==n)
{
distance=0;
for(a=0;a<n;a++)
{
        b=99;
        z=j;
        while(z!=m)
        {
        if((z-j)>distance)
        if(mem[a]==process[z])
        {
        distance=(z-j);
        b=z;
        }
        z++;
        }
}
if(b==99)
b=i;
mem[b]=process[j];
```

```
        i=(i+1)%n;
        }
        printf("\nMemory: ");
        for(z=0;z<n;z++)
        printf("%d ",mem[z]);
        j++;
    }
}
```

OBSERVATION :

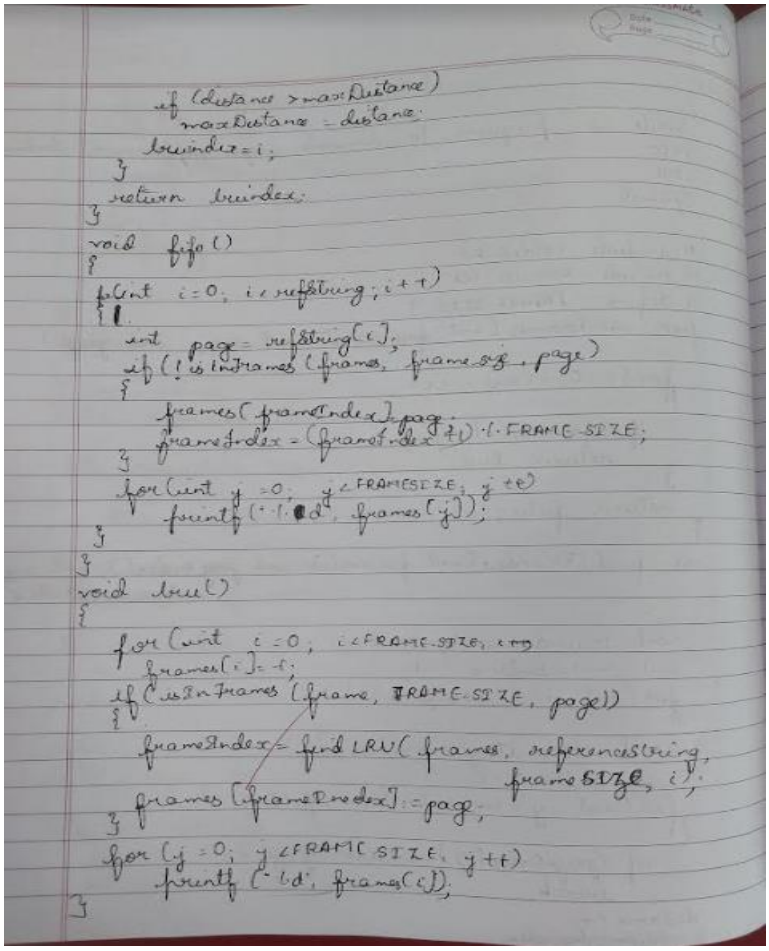


23/8/25

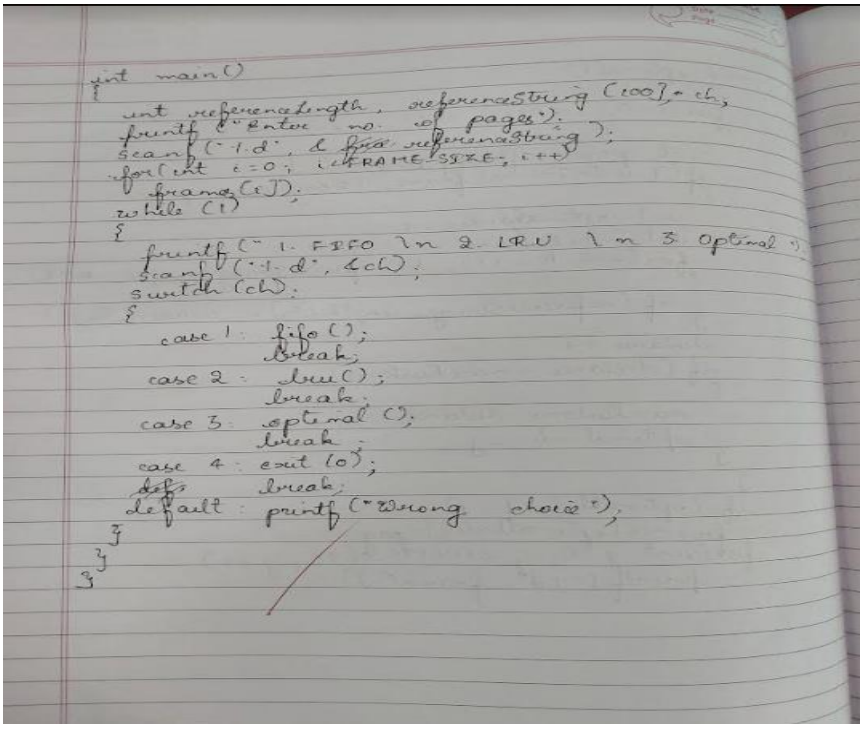
Week 10

Write a program to simulate page replacement techniques
- FIFO
- LRU
- Optimal

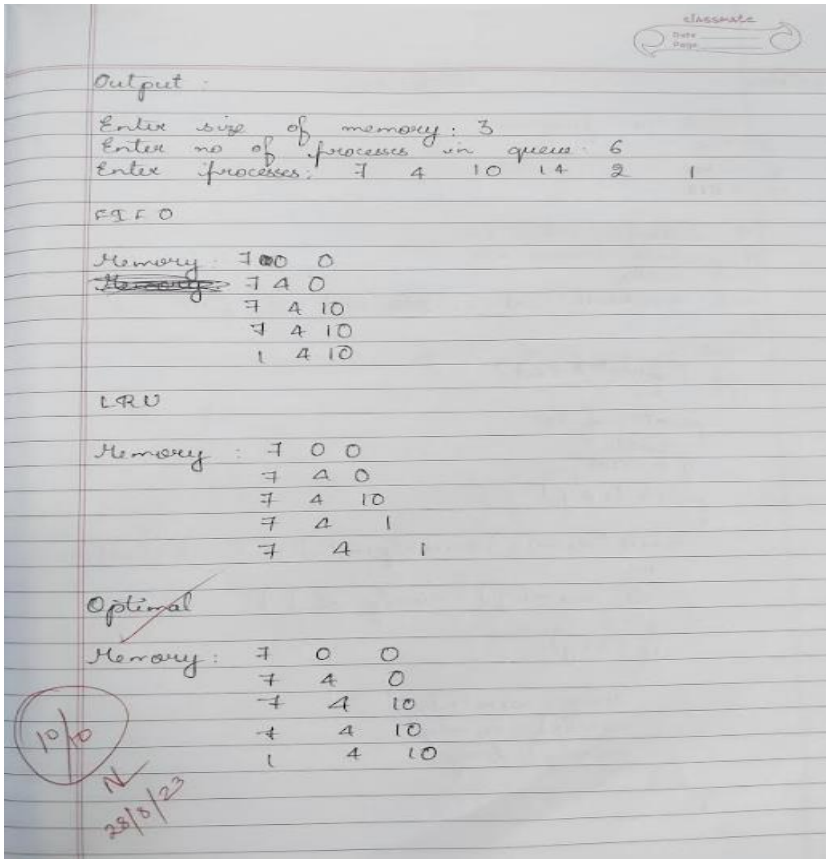
```
#include <stdio.h>
#include <conio.h>
#define FRAME_SIZE 3
bool isInFrames(int frames[], int size, int page)
{
    for(i=0; i<size; i++)
    {
        if (frames[i]==page)
            return true;
    }
    return false;
}
int findLRUindex(int frames[], int pageOrder[], int size,
                 int currentIndex)
{
    int lruIndex = -1;
    int maxDistance = -1;
    for(int i=0; i<size; i++)
    {
        int currentPage = frames[i];
        int distance = 0;
        for(int j = currentIndex-1; j>=0; j--)
        {
            if (pageOrder[j] == currentPage)
                break;
            distance++;
```



```
        if (distance > maxDistance)
            maxDistance = distance;
        lruIndex = i;
    }
    return lruIndex;
}
void fifo()
{
    for(int i=0; i<refStringLength; i++)
    {
        int page = refString[i];
        if(!isInFrames(frames, frameSize, page))
        {
            frames[frameIndex] = page;
            frameIndex = (frameIndex+1) % FRAME_SIZE;
        }
        for(int j=0; j<FRAMESIZE; j++)
            printf("%d", frames[j]);
    }
}
void lru()
{
    for(int i=0; i<FRAME_SIZE; i++)
        frames[i] = -1;
    if(isInFrames(frame, FRAME_SIZE, page))
    {
        frameIndex = findLRU(frames, referenceString,
                             frameSize, i);
    }
    frames[frameIndex] = page;
    for(j=0; j<FRAME_SIZE; j++)
        printf("%d", frames[j]);
}
```



```
int main()
{
    int referenceLength, referenceString[100], ch;
    printf("Enter no. of pages");
    scanf("%d", &referenceString);
    for(int i=0; i<FRAME_SIZE; i++)
        frames[i];
    while(1)
    {
        printf("1. FIFO \n 2. LRU \n 3. Optimal");
        scanf("%d", &ch);
        switch(ch);
        {
            case 1: fifo();
                    break;
            case 2: lru();
                    break;
            case 3: optimal();
                    break;
            case 4: exit(0);
                    break;
            default: printf("Wrong choice");
        }
    }
}
```



OUTPUT :

```
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
```