11. Write a C program to simulate page replacement algorithms
a) FIFO
b) LRU
c) Optimal

Code:

```c
#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)
```

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

printf("\n\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;
```

```c
            }
            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)
            {
```

```c
            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++;
    }
}
```

Output:

```
PS D:\VS Code\OS> cd "d:\VS Code\OS\" ; if ($?) { gcc PR.c -o PR } ; if ($?) { .\PR }
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
PS D:\VS Code\OS> []
```