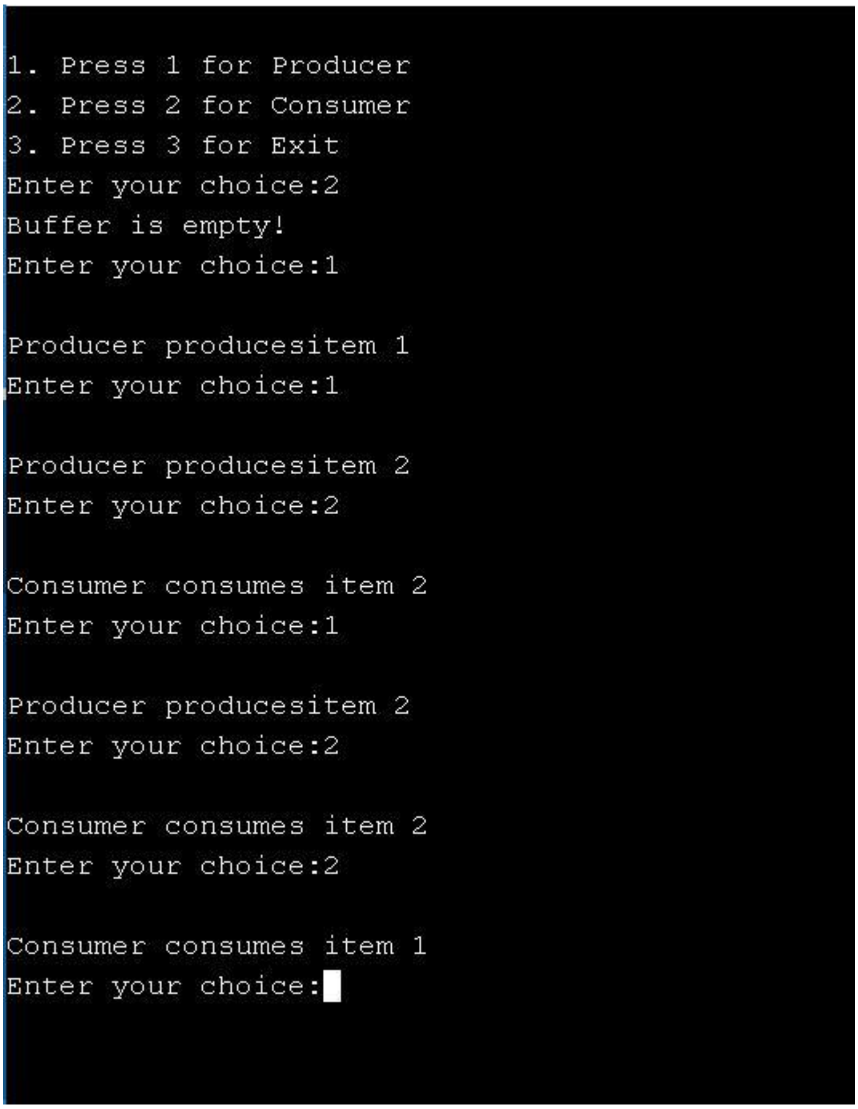
1. Producer Consumer

```
#include <stdio.h>
#include <stdlib.h>

int mutex = 1;

int full = 0;

int empty = 10, x = 0;

void producer()
{

    --mutex;

    ++full;


    --empty;


    x++;
    printf("\nProducer produces"
           "item %d",
           x);


    ++mutex;
}

void consumer()
{

    --mutex;


    --full;

    ++empty;
    printf("\nConsumer consumes "
           "item %d",
           x);
    x--;


    ++mutex;
}

// Driver Code
int main()
{
    int n, i;
    printf("\n1. Press 1 for Producer"
           "\n2. Press 2 for Consumer"
           "\n3. Press 3 for Exit");
#pragma omp critical

    for (i = 1; i > 0; i++) {

        printf("\nEnter your choice:");
        scanf("%d", &n);


        switch (n) {
        case 1:

            if ((mutex == 1)
                && (empty != 0)) {
                producer();
            }

            else {
                printf("Buffer is full!");
            }
            break;

        case 2:

            if ((mutex == 1)
                && (full != 0)) {
                consumer();
            }


            else {
                printf("Buffer is empty!");
            }
            break;

        case 3:
            exit(0);
            break;
        }
    }
}
```

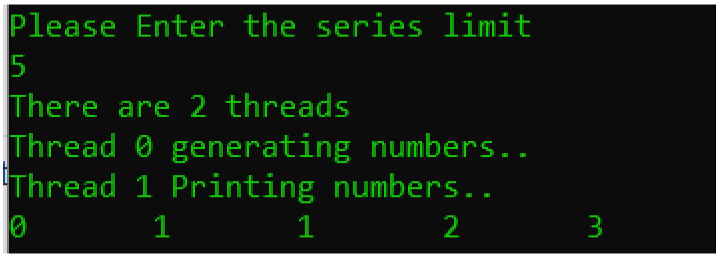


```
1. Press 1 for Producer
2. Press 2 for Consumer
3. Press 3 for Exit
Enter your choice:2
Buffer is empty!
Enter your choice:1

Producer producesitem 1
Enter your choice:1

Producer producesitem 2
Enter your choice:2

Consumer consumes item 2
Enter your choice:1

Producer producesitem 2
Enter your choice:2

Consumer consumes item 2
Enter your choice:2

Consumer consumes item 1
Enter your choice:
```

2. Multi Thread Program

```
#include<stdio.h>
#include<omp.h>

int fib(int n)
{
if(n<2) return n;
else return fib(n-1)+fib(n-2);
}

int main()
{
int fibnumber[100],i,j,n;
printf("Please Enter the series limit\n");
scanf("%d",&n);
#pragma omp parallel num_threads(2)
{
```

```
#pragma omp critical
if(omp_get_thread_num()==0)
{
printf("There are %d threads\n", omp_get_num_threads());
printf("Thread %d generating numbers..\n", omp_get_thread_num());
for(i=0;i<n;i++)
fibnumber[i]=fib(i);
}
else
{
printf("Thread %d Printing numbers..\n", omp_get_thread_num());
for(j=0;j<n;j++)
printf("%d\t", fibnumber[j]);
}

}
 return 0;
}
```



```
Please Enter the series limit
5
There are 2 threads
Thread 0 generating numbers..
Thread 1 Printing numbers..
0       1       1       2       3
```