```
/*
  Mihir Kulkarni
  33132 L9
  Aim : Demonstrate producer consumer problem with counting semaphores and mutex.
*/
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>

#define TRUE 1
#define BUFFER_SIZE 5

int buffer[BUFFER_SIZE];
pthread_mutex_t mutex;        //binary semaphore
sem_t empty;                  //counting semaphore
sem_t full;                           //counting semaphore

int nextIn = 0, nextOut = 0;   //count


void *producer()
{
        int item;

        while(TRUE) {
                sleep(3);
                //if >zero decrement empty, if zero call blocks
                sem_wait(&empty);
                //lock that we set before using a shared resource and release after using it
                pthread_mutex_lock(&mutex);

                int item=rand()%10;
                //Insert item into next position and update next position
                buffer[nextIn] = item;
                nextIn = (nextIn + 1) % BUFFER_SIZE;

                printf("\nProducer id: %u produced %d \n", (unsigned int)pthread_self(), item);
                for(int i=0;i<BUFFER_SIZE;i++)
                {
                        printf("%d ",buffer[i]);
                }
                //unlocks and releases for new thread to lock
                pthread_mutex_unlock(&mutex);
                sem_post(&full);        //increments
        }
}

void *consumer()
{
        int item;
```

```
        while(TRUE){
                sleep(5);
                //if >zero decrement full, if zero call blocks
                sem_wait(&full);
                //lock mutex to this thread
                pthread_mutex_lock(&mutex);
                //Remove item from position and update next position
                int item = buffer[nextOut];
                buffer[nextOut]=0;
                nextOut = (nextOut + 1) % BUFFER_SIZE;

                printf("\nConsumer id: %u consumed %d \n", (unsigned int)pthread_self(), item);

                for(int i=0;i<BUFFER_SIZE;i++)
                {
                        printf("%d ",buffer[i]);
                }
                //unlocks and releases for new thread to lock
                pthread_mutex_unlock(&mutex);
                sem_post(&empty);    //increments
        }
}

int main()
{
        //Declaration of inputs
        int producerThreads, consumerThreads;
        int i, j;
        printf("\n ---PRODUCER CONSUMER PROBLEM---\n");
        //Input
        printf("\nEnter no. of producers :");
        scanf("%d",&producerThreads);
        printf("\nEnter no. of consumers :");
        scanf("%d",&consumerThreads);
        //Initialization
        pthread_mutex_init(&mutex, NULL);
        sem_init(&empty, 0, BUFFER_SIZE);
        sem_init(&full, 0, 0);

        pthread_t *pid,*cid;
        //Dynamic creation of threads
   pid = (pthread_t*)malloc(producerThreads*sizeof(pthread_t));
   cid = (pthread_t*)malloc(consumerThreads*sizeof(pthread_t));

        //pthread_t pid[producerThreads], cid[consumerThreads];

        //creating producer and consumer threads
        for(i = 0; i < producerThreads; i++){
                pthread_create(&pid[i],NULL,producer,NULL);
        }

        for(j = 0; j < consumerThreads; j++){
```

```
                pthread_create(&cid[j],NULL,consumer,NULL);
        }
        //joining producer and consumer threads
        for(int i = 0; i < producerThreads; i++) {
      pthread_join(pid[i], NULL);
    }
    for(int i = 0; i < consumerThreads; i++) {
      pthread_join(cid[i], NULL);
    }

    //exit
        pthread_mutex_destroy(&mutex);
        sem_destroy(&empty);
        sem_destroy(&full);

        return 0;
}
```

```
mihir@pop-os:~/TE/OS-lab/a4$ gcc -D_REENTRANT a4_test.c -o a4_test -lpthread
a4_test.c: In function ‘producer’:
a4_test.c:27:3: warning: implicit declaration of function ‘sleep’ [-Wimplicit-function-declaration]
   27 |   sleep(3);
      |   ^~~~~
mihir@pop-os:~/TE/OS-lab/a4$ ./a4_test

 ---PRODUCER CONSUMER PROBLEM---

Fixed buffer Size : 5

Enter no. of producers :4

Enter no. of consumers :2

Producer id: 2646394624 produced 3
3 0 0 0 0
Producer id: 2621216512 produced 6
3 6 0 0 0
Producer id: 2629609216 produced 7
3 6 7 0 0
Producer id: 2638001920 produced 5
3 6 7 5 0
Consumer id: 2604431104 consumed 3
0 6 7 5 0
Consumer id: 2612823808 consumed 6
0 0 7 5 0
Producer id: 2646394624 produced 3
0 0 7 5 3
Producer id: 2621216512 produced 5
5 0 7 5 3
Producer id: 2629609216 produced 6
5 6 7 5 3
Consumer id: 2604431104 consumed 7
5 6 0 5 3
Consumer id: 2612823808 consumed 5
5 6 0 0 3
Producer id: 2638001920 produced 2
5 6 2 0 3
Producer id: 2646394624 produced 9
5 6 2 9 3
Consumer id: 2604431104 consumed 3
5 6 2 9 0
Consumer id: 2612823808 consumed 5
0 6 2 9 0
Producer id: 2621216512 produced 1
0 6 2 9 1
Producer id: 2629609216 produced 2
^C
mihir@pop-os:~/TE/OS-lab/a4$
```