```
#include<stdio.h>
#include<pthread.h>
#include<stdlib.h>
#include<semaphore.h>
#include<unistd.h>
#define buffer_size 10
sem_t full,empty;
int buffer[buffer_size];
pthread_mutex_t mutex;
void *producer(void *p);
void *consumer(void *p);
void insert_item(int);
int remove_item();
int counter;
void initialize()
{
pthread_mutex_init(&mutex,NULL);
sem_init(&full,1,0);
sem_init(&empty,1,buffer_size);
counter=0;
}
int main()
{
int n1,n2,i;
printf("\nEnter no. of producers you want to create:");
scanf("%d",&n1);
printf("\nEnter no. of consumers you want to create:");
scanf("%d",&n2);
initialize();
pthread_t tid[n1],tid1[n2];
for(i=0;i<n1;i++)
pthread_create(&tid[i],NULL,producer,NULL);

for(i=0;i<n2;i++)pthread_create(&tid1[i],NULL,consumer,NULL);

sleep(50);

exit(0);

}

void *producer(void *p)

{

int item,waittime;

waittime=rand()%5;

sleep(waittime);

item =rand()%10;

sem_wait(&empty);

pthread_mutex_lock(&mutex);

printf("\n Producer produced %d item",item);

insert_item(item);

pthread_mutex_unlock(&mutex);

sem_post(&full);

}

void *consumer(void *p)

{

int item,waittime;

waittime=rand()%10;

sleep(waittime);

sem_wait(&full);

pthread_mutex_lock(&mutex);

item=remove_item();

printf("\n Consumer consumed %d item",item);

pthread_mutex_unlock(&mutex);

sem_post(&empty);

}

void insert_item(int item)
```

```
{buffer[counter++]=item;
}
int remove_item()
{
return(buffer[--counter]);
}
```