Report
Assignment 3

Answer to Question 1 –
In this implementation of Producer and Consumer, synchronization is achieved using the pre-implemented semaphores of xinu. We use two semaphores here, namely, *produced* and *consumed*. The consumer waits on the semaphore *produced* whereas the producer waits on the semaphore *consumed*. The semaphores are created dynamically using the ***semcreate*** system call. It takes the desired initial count as an argument. The ***wait*** system call decrements a semaphore and adds the calling process to the set of waiting process if the result is negative. The ***signal*** system call does the opposite function of incrementing and allowing one of the waiting processes to continue if there are any. Thus the producer waits on the semaphore *consumed* and consumer waits on the semaphore *produced*. The critical section is between the wait and signal calls so, synchronization is achieved by allowing either the producer or consumer to do their operations on '*n*'.

Answer to Question 2 –
The above synchronization cannot be achieved using just one semaphore because we need two, one on which the producer waits and one on which the consumer will wait. If we use only one semaphore, then the system calls wait and signal will operate on the same semaphore, thus not blocking a process which should otherwise be blocked.

Functions:

- void producer(int) – Omkar
- void consumer(int) - Ninad