Assignment - 4

Aim : Write a Python program to simulate producer-consumer problem using semaphores.

Theory :

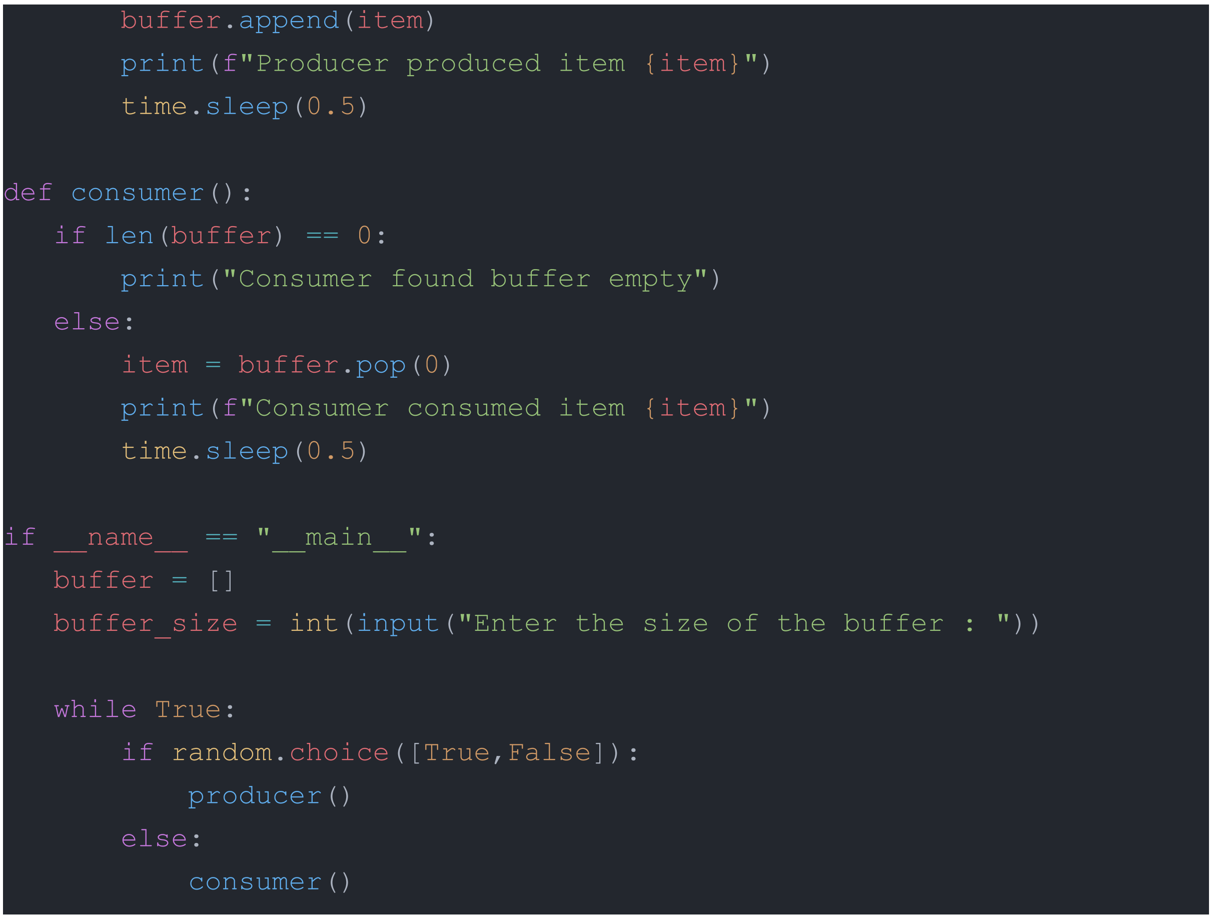
Semaphore:
A semaphore is a special kind of synchronization data that can be used only through specific synchronization primitives.
When a process performs a wait operation on a semaphore, the operation checks whether the value of the semaphore is >0. If so, it decrements the value of the semaphore and lets the process continue its execution; otherwise, it blocks the process on the semaphore.
A signal operation on a semaphore activates a process blocked on the semaphore if any, or increments the value of the semaphore by 1.
Due to these semantics, semaphores are also called counting semaphores. The initial value of a semaphore determines how many processes can get past the wait operation.

Bounded buffer:
Bounded buffer problem, which is also called producer consumer problem, is one of the classic problems of synchronization. A bounded buffer lets multiple producers and multiple consumers share a single buffer. Producers write data to the buffer and consumers read data from the buffer. Producers must block if the buffer is full.

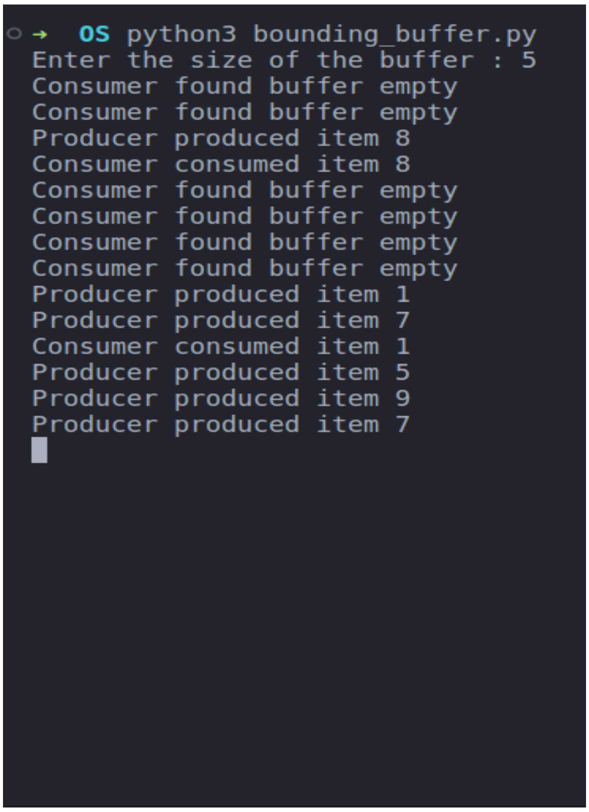
1. There is a buffer of n slots and each slot is capable of storing one unit of data. There are two processes running, namely, producer and consumer, which are operating on the buffer.
2. A producer tries to insert data into an empty slot of the buffer.
3. A consumer tries to remove data from a filled slot in the buffer.

Program & Output :

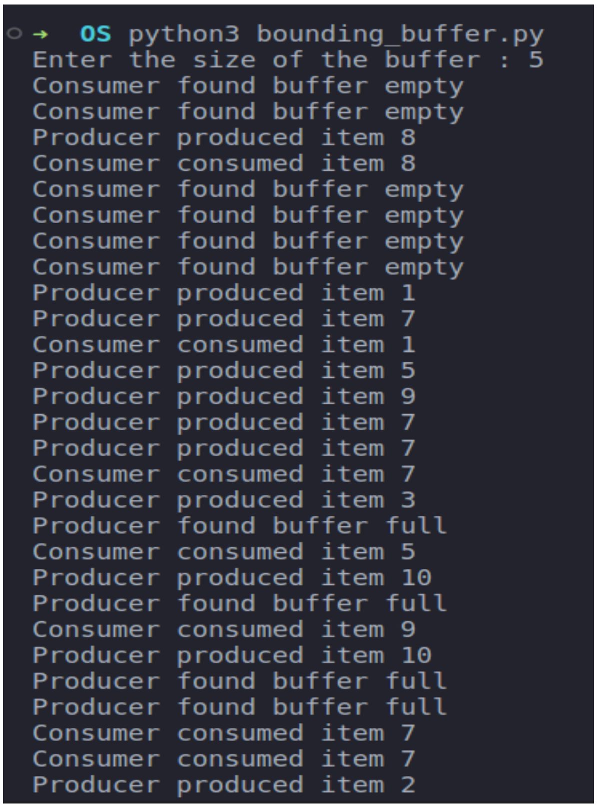
```
import time
import random

def producer():
   if len(buffer) >= buffer_size:
       print("Producer found buffer full")
   else:
       item = random.randint(1, 10)
```



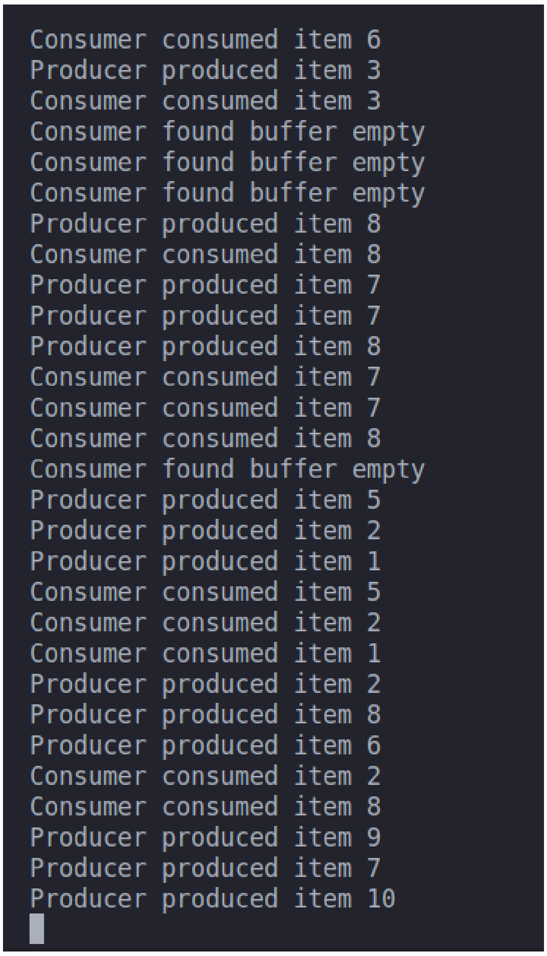
```
        buffer.append(item)
        print(f"Producer produced item {item}")
        time.sleep(0.5)

def consumer():
    if len(buffer) == 0:
        print("Consumer found buffer empty")
    else:
        item = buffer.pop(0)
        print(f"Consumer consumed item {item}")
        time.sleep(0.5)

if __name__ == "__main__":
    buffer = []
    buffer_size = int(input("Enter the size of the buffer : "))

    while True:
        if random.choice([True,False]):
            producer()
        else:
            consumer()
```



```
○ →  OS python3 bounding_buffer.py
  Enter the size of the buffer : 5
  Consumer found buffer empty
  Consumer found buffer empty
  Producer produced item 8
  Consumer consumed item 8
  Consumer found buffer empty
  Consumer found buffer empty
  Consumer found buffer empty
  Consumer found buffer empty
  Producer produced item 1
  Producer produced item 7
  Consumer consumed item 1
  Producer produced item 5
  Producer produced item 9
  Producer produced item 7
```



```
○ →  OS python3 bounding_buffer.py
  Enter the size of the buffer : 5
  Consumer found buffer empty
  Consumer found buffer empty
  Producer produced item 8
  Consumer consumed item 8
  Consumer found buffer empty
  Consumer found buffer empty
  Consumer found buffer empty
  Consumer found buffer empty
  Producer produced item 1
  Producer produced item 7
  Consumer consumed item 1
  Producer produced item 5
  Producer produced item 9
  Producer produced item 7
  Producer produced item 7
  Consumer consumed item 7
  Producer produced item 3
  Producer found buffer full
  Consumer consumed item 5
  Producer produced item 10
  Producer found buffer full
  Consumer consumed item 9
  Producer produced item 10
  Producer found buffer full
  Producer found buffer full
  Consumer consumed item 7
  Consumer consumed item 7
  Producer produced item 2
```



```
Consumer consumed item 6
Producer produced item 3
Consumer consumed item 3
Consumer found buffer empty
Consumer found buffer empty
Consumer found buffer empty
Producer produced item 8
Consumer consumed item 8
Producer produced item 7
Producer produced item 7
Producer produced item 8
Consumer consumed item 7
Consumer consumed item 7
Consumer consumed item 8
Consumer found buffer empty
Producer produced item 5
Producer produced item 2
Producer produced item 1
Consumer consumed item 5
Consumer consumed item 2
Consumer consumed item 1
Producer produced item 2
Producer produced item 8
Producer produced item 6
Consumer consumed item 2
Consumer consumed item 8
Producer produced item 9
Producer produced item 7
Producer produced item 10
```

Conclusion : Here in this assignment, we studied bounded buffer and implemented producer & consumer in python.