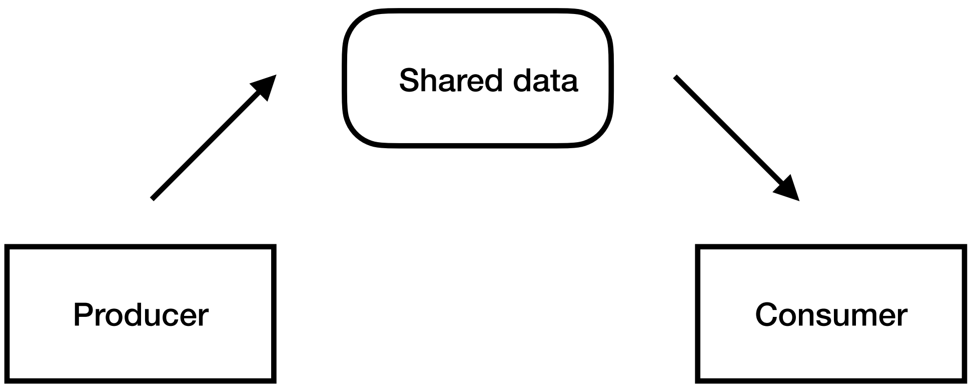
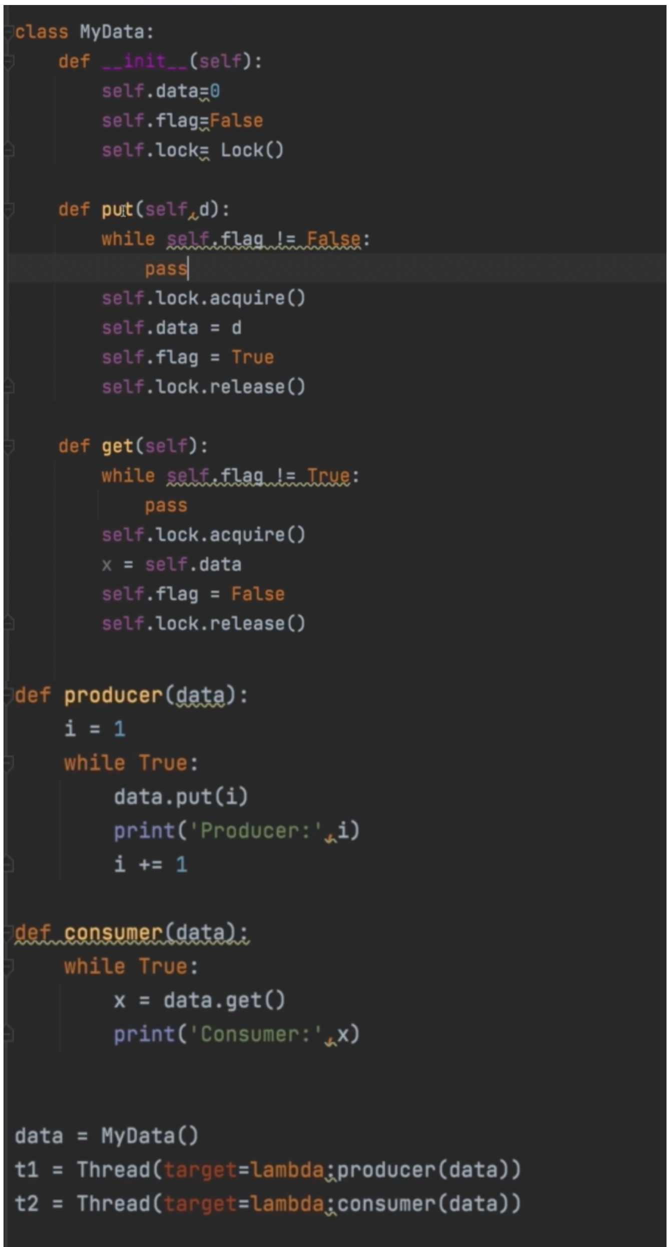
InterProcess Communication

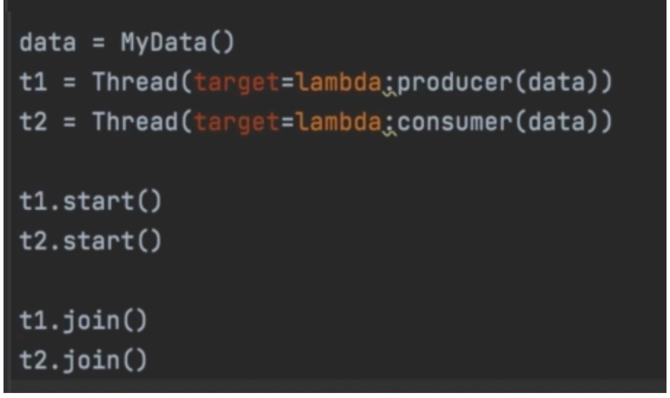
- Two or more processors communicating with each other is said said inter Process Communication but in this we are having threads so this is inter Process thread Communication
- Lets take example



- Producer will produce the data and consumer will consumes the data
- If you want that one by one then we use flag . Suppose if flag = false then it is producers turn after putting the value the flag will be true that means now its consumers turn .
- Consumer will consume that data and make the flag = false
- Shared data have two methods that is put ( ) and get ( )
- put( ) is for producer and get( ) is for consumer
- They will exchange term by flag value
- Whenever producer is putting the value first it should lock it then put the value and same goes for consumer whenever consumer is consuming the value first it should put a lock and then consumes the value



```python
class MyData:
    def __init__(self):
        self.data=0
        self.flag=False
        self.lock= Lock()

    def put(self,d):
        while self.flag != False:
            pass
        self.lock.acquire()
        self.data = d
        self.flag = True
        self.lock.release()

    def get(self):
        while self.flag != True:
            pass
        self.lock.acquire()
        x = self.data
        self.flag = False
        self.lock.release()

def producer(data):
    i = 1
    while True:
        data.put(i)
        print('Producer:',i)
        i += 1

def consumer(data):
    while True:
        x = data.get()
        print('Consumer:',x)


data = MyData()
t1 = Thread(target=lambda:producer(data))
t2 = Thread(target=lambda:consumer(data))
```



```python
data = MyData()
t1 = Thread(target=lambda:producer(data))
t2 = Thread(target=lambda:consumer(data))
```

```python
t1.start()
t2.start()

t1.join()
t2.join()
```