UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS

DIVISIÓN DE ELECTRÓNICA Y COMPUTACIÓN

DIVTIC

Tema: Producto-Consumidor

PRESENTA:
(Jared Isaías Monje Flores)

Docente:
(Javier Rosales Martinez)

Materia:
(Seminario de solución de sistemas operativos)

Producto-Consumidor

El problema del productor y consumidor es muy popular en el mundo de la programación paralela. El problema considera tres objetos: un productor, un consumidor, y un contenedor. Las reglas del problema son estas:

- El productor tiene la tarea de producir N productos.
- Cada producto nuevo deberá ser colocado en el contenedor.
- Si el contenedor está lleno, el productor deberá esperar a que el consumidor haga espacio para nuevos productos.
- El consumidor tiene la tarea de procesar los N productos.
- Cada vez que se procesa un producto, se quita del contenedor.
- Si el contenedor está vacío, el consumidor deberá esperar al productor para que haya productos para procesar.

Este problema se utiliza mucho para aplicar conceptos de sincronía de procesos o hilos (un proceso que ejecuta dos o más algoritmos).

Los problemas y aplicaciones más comunes asociados con el problema del producto-consumidor incluyen:

- **Buffer limitado:** Un escenario común es cuando los productores generan datos más rápido de lo que los consumidores pueden procesarlos. Esto puede llevar a un desbordamiento del búfer o una cola si no se gestionan adecuadamente los límites del búfer.
- **Sincronización:** Es necesario garantizar que los productores no produzcan datos en un búfer lleno y que los consumidores no consuman de un búfer vacío. La sincronización adecuada es fundamental para evitar problemas como el acceso concurrente a datos compartidos.
- **Consumidores lentos:** Si los consumidores son más lentos que los productores, es posible que se acumulen datos en el búfer, lo que puede llevar a un aumento de la latencia o al agotamiento de los recursos.
- **Productores excesivos:** Si hay demasiados productores generando datos, el búfer puede llenarse rápidamente y agotar los recursos del sistema.
- **Bloqueo y esperas:** En algunos casos, los productores y consumidores pueden bloquearse mutuamente si no se implementan correctamente mecanismos de bloqueo y espera. Esto puede llevar a la ineficiencia en la utilización de recursos.
- **Pérdida de datos:** Si no se maneja adecuadamente el flujo de datos, es posible que se pierdan datos si el búfer se llena y no se puede manejar más entrada.

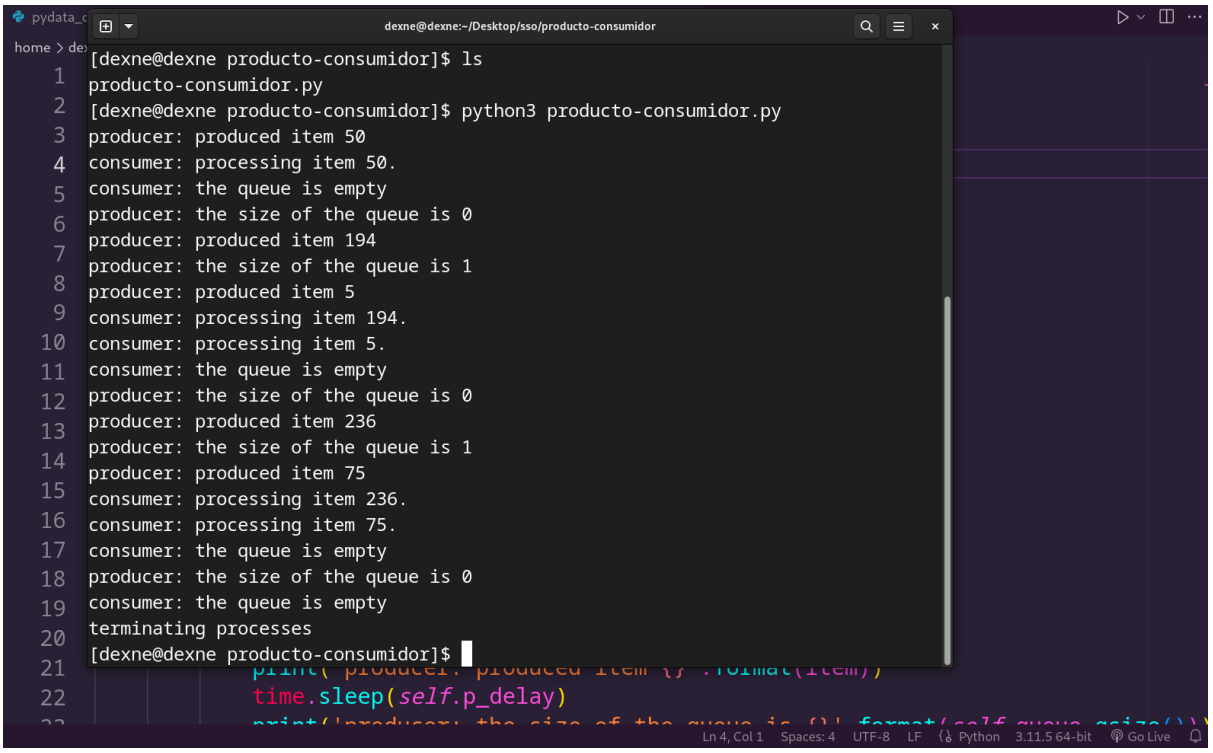
Aplicaciones comunes del problema del productor-consumidor incluyen la impresión en cola, la gestión de recursos compartidos, la comunicación entre procesos, la gestión de hilos de ejecución en programación multihilo y la gestión de tareas en sistemas de procesamiento en lotes.

En esta ocasión utilizo Python para resolver el problema. Python tiene varias utilerías incluidas que ayudan a resolver este problema muy fácil.
Para empezar, hagamos que el productor sea un proceso (p). Este productor se encargará de producir una cantidad específica (p_load) de números aleatorios. Los números producidos se colocarán en un contenedor tipo "cola" (q). La producción de cada número le tomara p_delay segundos. Contará además con una señal especial llamada pf cuyo valor numérico es 0 mientras produce y 1 cuando termine de producir.
El consumidor (c) será también otro proceso. El consumidor tomará los números de la "cola" llamada q cada c_delay segundos. Y cuando no haya productos en el contenedor, el consumidor hará que cf tenga valor de 1. Pero si encuentra números en el contenedor, hará que cf sea 0.
Si el contenedor se llena, el productor revisará cada segundo si el consumidor ya hizo espacio. Si el contenedor se vacía, el consumidor revisará cada segundo si el productor ha puesto nuevos números en dicho contenedor.
El programa completo terminará cuando tanto el consumidor como el productor indiquen que terminaron sus tareas mediante pf y cf.

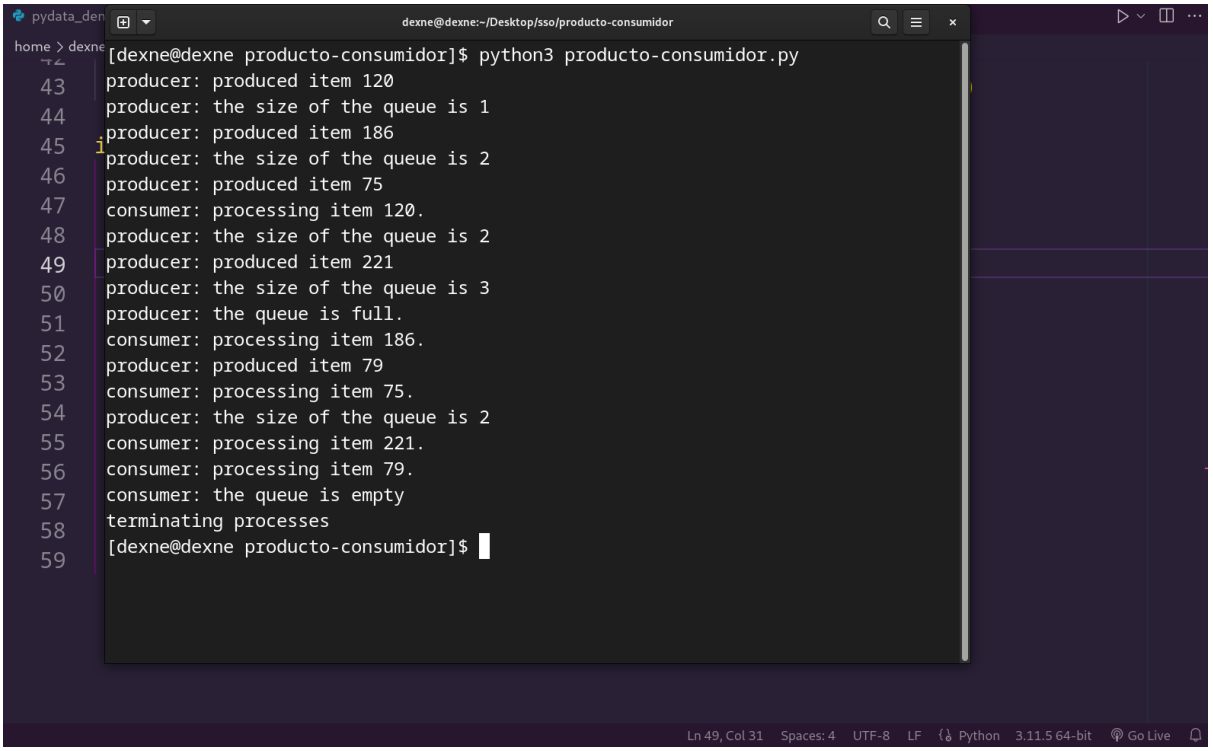
Código fuente:

```python
import multiprocessing
import random
import time

class Producer(multiprocessing.Process):
    def __init__(self, queue, p_load, p_delay, p_done):
        multiprocessing.Process.__init__(self)
        self.queue = queue
        self.p_done = p_done
        self.p_delay = p_delay
        self.p_load = p_load

    def run(self):
        for i in range(self.p_load):
            while self.queue.full():
                print('producer: the queue is full.')
                time.sleep(1)
            item = random.randint(0, 256)
            self.queue.put(item)
            print('producer: produced item {}'.format(item))
            time.sleep(self.p_delay)
            print('producer: the size of the queue is {}'.format(self.queue.qsize()))
        self.p_done.value = 1.0

class Consumer(multiprocessing.Process):
    def __init__(self, queue, c_delay, c_done):
        multiprocessing.Process.__init__(self)
        self.queue = queue
        self.c_done = c_done
        self.c_delay = c_delay

    def run(self):
        while True:
            if self.queue.empty():
                self.c_done.value = 1.0
                print('consumer: the queue is empty')
                time.sleep(1)
            else:
                self.c_done.value = 0.0
                time.sleep(self.c_delay)
                item = self.queue.get()
                print('consumer: processing item {}.'.format(item))

if __name__ == '__main__':
    pf = multiprocessing.Value('d', 0.0)
    cf = multiprocessing.Value('d', 0.0)
    q = multiprocessing.Queue(3)
    p = Producer(q, 5, 1.0, pf)
    c = Consumer(q, 0.5, cf)
    p.start()
    c.start()
    while True:
        if int(pf.value + cf.value) == 2:
            print('terminating processes')
            p.terminate()
            c.terminate()
            break
        time.sleep(1)
```

Ejecución de prueba: Consumidor más rápido que el productor (“p_load” = 5, “p_delay” = 1, “c_delay” = 0.5)



Productor más rápido que el consumidor (“p_load” = 5, “p_delay” = 0.5, “c_delay” = 1)



Referencias:

LinkedIn. (2023). Linkedin.com. https://www.linkedin.com/pulse/problema-del-productor-y-el-consumidor-ricardo-alejos/?originalSubdomain=es

con, P. (2019, August 4). *Problema con el modelo Productor-Consumidor*. Stack Overflow En Español. https://es.stackoverflow.com/questions/283688/problema-con-el-modelo-productor-consumidor

Wikiwand - Problema productor-consumidor. (2022). Wikiwand; Wikiwand. https://www.wikiwand.com/es/Problema_productor-consumidor