

Examen de Arquitectura Concurrente "Pizzería Python"

Escenario

El objetivo es programar un sistema de gestión de pedidos para una pizzería que funcione de forma totalmente concurrente. El sistema debe procesar la entrada de pedidos, llevar la contabilidad, cocinar los productos y realizar el reparto a domicilio de forma simultánea.

Requisitos Técnicos

1. Comunicación y Datos

Deberás crear dos colas (**Queue**) y un canal de comunicación directo (**Pipe**):

- **cola_cocina**: Para enviar pedidos del teléfono a la cocina.
- **cola_mostrador**: Para enviar pizzas de la cocina a los repartidores.
- **pipe_contabilidad**: Para enviar el precio de cada pedido del teléfono a la oficina de cuentas.

Cada pedido será un diccionario con esta estructura: `{'id': 1, 'pizza': 'pizza hawaiana', 'precio': 15}`.

2. Definición de Procesos

Debes implementar las siguientes funciones y lanzarlas como procesos independientes:

- **Proceso Teléfono (1)**: Una función `recibir_llamadas` que recorre una lista de pedidos. Por cada uno, pinte en consola el pedido completo, envía el pedido completo a la cocina y, simultáneamente, envía **solo el precio** por el Pipe a contabilidad. Después de enviar el pedido a cocina y a contabilidad, tiene que esperar 3 segundos a "recibir" la siguiente llamada. Al terminar la lista de pedidos debe imprimir en consola el mensaje `"Ya no hay más pedidos, se cierra el teléfono"` (Pista: también debería gestionar el cierre de los procesos que dependen de él).
- **Proceso Contabilidad (1)**: Una función `facturar` que escucha el Pipe de forma ininterrumpida. Debe ir sumando los precios recibidos y, al recibir la señal de parada, imprimir el total de ventas del día: `"Total ventas: XX€"`.
- **Proceso Cocinero (1)**: Una función `cocinar` que extrae pedidos de la `cola_cocina`. Simula el tiempo de cocción con un `time.sleep(XXX)`. El valor de ese XXX debe ser la longitud de la string de `'pizza'`. Una vez terminado, pone el pedido en la `cola_mostrador`. Cuando ya no queden pedidos que cocinar debe imprimir en consola el mensaje `"Ya no hay más pedidos, se cierra la cocina"` (Pista: también debería gestionar el cierre de los procesos que dependen de él).
- **Procesos Repartidores (2)**: Dos procesos que a la vez invocan a la función `repartir` que saca pedidos de la `cola_mostrador`. Cada entrega debe tardar 15 segundos y después imprimir un mensaje indicando qué repartidor ha entregado qué pedido. Cuando ya no quedan pedidos pendientes, cada repartidor debe imprimir en consola el mensaje `"El repartidor XXX ha terminado su jornada"`
- Al final de todo el código debes imprimir por pantalla `Telepizza cerrado`

Puntos clave

1. Todo el código del examen debe estar en un unico archivo `.py`.

2. Todos los procesos deben cerrarse antes de cerrar el programa principal
3. Si ves que te bloqueas al tener dos repartidores a la vez, crea solo uno, mejor eso que nada
4. Asegúrate de que los mensajes por pantalla permitan seguir el flujo (ej: "Cocinando pedido 1...", "Repartiendo pedido 1...").
5. El código debe estar comentado explicando la estrategia de cierre de los procesos. Si comentais las funciones que se usan en cada proceso os va a ser mucho más fácil tener un código que funcione