Crea un programa que simule un **sistema de gestión de inventario de productos** usando programación orientada a objetos. El sistema debe incluir:

1. **Clase principal** `Inventario` con:

•Atributos: nombre de la tienda y diccionario de productos (clave: nombre, valor: cantidad en stock).

Métodos para manejar operaciones (cada uno debe usar `raise` para lanzar excepciones personalizadas cuando corresponda):

•`agregar_producto(nombre, cantidad)`: Añade o actualiza un producto.
•`vender_producto(nombre, cantidad)`: Reduce el stock si hay suficiente.
•`consultar_stock(nombre)`: Devuelve la cantidad disponible.
•`eliminar_producto(nombre)`: Elimina un producto del inventario.
•`mostrar_inventario()`: Lista todos los productos con su stock.

2. **Excepciones personalizadas** (cada una hereda de `Exception`):

•`ProductoNoEncontradoError`: Cuando un producto no existe.
•`StockInsuficienteError`: Cuando no hay suficiente stock para vender.
•`CantidadInvalidaError`: Cuando se ingresa una cantidad negativa o cero.
•`ProductoDuplicadoError`: Cuando se intenta agregar un producto ya existente (en agregar nuevo).

3. **Menú principal** que:

•Crea una instancia de `Inventario`.
•Presenta opciones numéricas para cada operación.
•Usa bloques `try-except` para capturar las excepciones personalizadas y mostrar mensajes específicos.
•También maneja `ValueError` para entradas numéricas incorrectas.

4. **Requisitos adicionales**:

•Validar que cantidades sean números positivos.
•El inventario inicial puede tener algunos productos precargados.
•Cada método de la clase debe documentar qué excepciones lanza con `raise`.

Objetivo: Practicar el uso de `try-except`, `raise`, excepciones personalizadas y programación orientada a objetos en un contexto integrado y realista.