Contexto:
Se pide desarrollar una API RESTful en ASP .NET 8 para la gestión de productos de una librería. Por el momento, la aplicación solo debe manejar objetos simples del tipo “Producto”, **sin base de datos**, utilizando datos en memoria.

Ejercicio 1 – Definición de Clase y Controlador

1.1. Crea una clase llamada Producto con los siguientes atributos:

- int Id
- string Nombre
- decimal Precio

1.2. Crea un controlador de tipo API llamado *ProductosController* (en la carpeta *Controllers*).

Ejercicio 2 – Listar productos (GET)

2.1. Implementa un endpoint GET */api/productos* que devuelva una lista fija de al menos **dos productos** de ejemplo en formato JSON.

- **Aclaración:**
  Los productos pueden estar definidos dentro del método (una List<Producto>).

Ejercicio 3 – Consultar producto por ID (GET)

3.1. Implementa un endpoint GET */api/productos/{id}* que reciba un ID como parámetro y devuelva el producto correspondiente.

3.2. Si no existe un producto con ese ID, el endpoint debe devolver un código de estado HTTP **404 (Not Found)**.

Ejercicio 4 – Crear producto (POST)

4.1. Implementa un endpoint POST */api/productos* que reciba los datos de un producto en el cuerpo del request (formato JSON) y lo devuelva con un **ID generado aleatoriamente** (no importa que no persista).

4.2. Si falta el nombre o el precio es menor o igual a cero, el endpoint debe devolver **400 (Bad Request)** y un mensaje de error descriptivo.

4.3. Si la creación es exitosa, debe devolver **201 (Created)** y el producto creado.

Ejercicio 5 – Actualizar producto (PUT)

5.1. Implementa un endpoint PUT */api/productos/{id}* que permita actualizar los datos de un producto existente.

5.2. Si el producto no existe, debe devolver **404 (Not Found)**.

5.3. Si los datos enviados son inválidos (por ejemplo, nombre vacío o precio <= 0), debe devolver **400 (Bad Request)**.

5.4. Si la actualización es exitosa, debe devolver **204 (No Content)**.

Ejercicio 6 – Eliminar producto (DELETE)

6.1. Implementa un endpoint DELETE */api/productos/{id}* que elimine un producto por su ID.

6.2. Si el producto no existe, debe devolver **404 (Not Found)**.

6.3. Si la eliminación es exitosa, debe devolver **204 (No Content)**.

Aclaraciones Generales

- Todos los endpoints deben respetar el formato y los códigos de estado HTTP indicados.
- **No es necesario implementar persistencia en base de datos**; puedes usar listas en memoria o datos hardcodeados.
- Se valora la correcta estructuración del código y el uso de buenas prácticas de nomenclatura.