Enunciado del Problema:

Implementa en C# dos clases llamadas `Producto` y `Comparativa` según las siguientes especificaciones:

Clase Producto

Debe tener los siguientes atributos:

- `Codigo` (string): Identificador único del producto.
- `Nombre` (string): Nombre descriptivo del producto.

Incluye contructor(es) para inicializar cómodamente nuevos productos.

Además, debe implementar:

1. **Equals**: Compara dos productos por su `Codigo` (dos productos son iguales si tienen el mismo código).
2. **IComparable**: Permite ordenar productos por su `Nombre`.
3. **ToString()**: Devuelve una cadena formateada con los datos del producto (ej: `"Código: P001, Nombre: Laptop"`).

Clase Articulo

Debe tener los siguientes atributos:

- `Vendedor` (string): Nombre del vendedor.
- `Producto` (Producto): Objeto de la clase `Producto`.
- `Precio` (decimal): Precio del producto para ese vendedor (puede diferir del precio base del producto).

Incluye contructor(es) para inicializar cómodamente nuevos artículos.

Clase Comparativa

Debe tener los siguientes atributos:

- `List<Articulo>` (articulos).

Además, debe incluir los siguientes métodos:

1. **Constructor**:
    - Recibe la ruta de un archivo CSV con el formato:

      ```
      Vendedor,CodigoProducto,NombreProducto,Precio
      Ejemplo:
      "Amazon,P001,Laptop,1200.50"
      ```

    - Lee el archivo y carga los datos en una lista de objetos `Comparativa`.(Deberá manejar excepciones al leer/escribir el archivo CSV)

2. **Guardar en CSV**:
    - Permite guardar los datos modificados en un archivo CSV con el mismo formato.
3. **Operaciones**:
    - **AñadirArticulo**: Agrega un nuevo artículo a la comparativa.
    - **ModificarPrecio**: Actualiza el precio de un producto para un vendedor específico.
    - **ToString()**: Devuelve un string con toda la comparativa (listado de productos con sus vendedores y precios).
    - **ListarPreciosDeProducto**: Muestra todos los vendedores y precios asociados a un producto específico (buscado por código).

Requisitos Adicionales Opcionales

- Validar que no se dupliquen productos con el mismo código al añadirlos.
- Manejar excepciones al leer/escribir el archivo CSV.

Ejemplo de uso:

```
Comparativa comp = new Comparativa("datos.csv");
comp.AñadirArticulo("MediaMarkt", "P002", "Smartphone", 599.99);
comp.ModificarPrecio("P001", "Amazon", 1150.00);
Console.WriteLine(comp.ListarPreciosDeProducto("P001"));
comp.GuardarCSV("datos_actualizados.csv");
```

Salida esperada (ejemplo):

```
=== Comparativa de precios ===
Amazon      - Laptop (P001):        1150.00€
MediaMarkt  - Smartphone (P002):    599.99€

=== Precios del producto P001 ===
Amazon:           1150.00€
OtroVendedor:     1200.50€
```