1. Realizar un servicio REST llamado PronosticoDemanda en Laravel para prever la demanda de productos en una tienda. (20 puntos)
   Los datos iniciales están guardados en una base de datos bd_demandas en una tabla pronostico con los siguientes registros:

| Fecha | CantidadEstimada |
|---|---|
| 01-06-25 | 150 |
| 02-06-25 | 145 |
| 03-06-25 | 165 |
| 04-06-25 | 170 |
| 05-06-25 | 180 |
| 06-06-25 | 130 |
| 07-06-25 | 160 |
| 08-06-25 | 190 |
| 09-06-25 | 200 |

   Se deben implementar todas las operaciones CRUD REST y además, crear un servicio GraphQL que permita las **mutaciones para crear y actualizar** un dato de pronóstico.
   Finalmente, desarrollar una aplicación de escritorio en C# que consuma el servicio, solicitando al usuario si desea usar REST o GraphQL para obtener y modificar los registros.
   .

2. Diseñar un sistema de monitoreo donde el cliente envía un **ID de sensor** (ej: "sensor01") y una cadena de estado ("bajo", "medio", "alto") al servidor.
   El servidor deberá almacenar los estados de los sensores en variables correspondientes bajo, medio, alto.
   El servidor responderá al cliente con el estado recibido seguido de un **OK**.(20 puntos)
   Posteriormente, el cliente enviará el mensaje "reporte", y el servidor responderá con el **sensor que ha reportado el estado más crítico** (ej: el primer sensor con estado "alto").
   Debe utilizar **multihilos** para permitir múltiples clientes a la vez, y cada cliente debe mantener **su propio estado interno** (no compartido).

3. Crear una aplicación RMI para el manejo remoto de cadenas. La interfaz remota debe contener los métodos(20 puntos)

java
CopiarEditar
boolean guardarFrase(String frase);
String convertirMayusculas();
String duplicarEspacios(int veces);
String concatenar(String extra);
El servidor debe mantener en memoria una cadena general que se modifica con guardarFrase.
El cliente debe presentar un menú para ejecutar cualquiera de los métodos anteriores, los cuales se aplicarán sobre la cadena actual.
Cada operación devuelve el resultado para ser mostrado al cliente.

4. Crear un servicio web SOAP llamado SEDUINFO que tenga los siguientes métodos: (20 puntos)

   Estudiante ObtenerDatosAcademicos(String CI) → devuelve:
   CI, nombres, apellidos, carrera, semestre, promedio
   Estudiante ObtenerDatosTutor(String CI) → devuelve:
   nombres del estudiante, tutorAsignado, correoTutor, teléfonoTutor

Los datos pueden ser estáticos o simulados.
Luego, crear una aplicación de escritorio en C# que permita al usuario ingresar el CI de un estudiante y consultar cualquiera de los dos métodos anteriores, mostrando los resultados en pantalla