Hoy ocurre un **eclipse solar anular**, también conocido como *"anillo de fuego"*. Durante este fenómeno, la Luna pasa frente al Sol, formando un brillante aro alrededor de su silueta. La Luna **no cubre totalmente el Sol**, dejando un borde luminoso visible desde determinadas zonas de la Tierra.

El propósito de este trabajo es **practicar conceptos de Programación Orientada a Objetos**, incluyendo:

- Herencia y polimorfismo
- Interfaces
- Enumeraciones (enum)
- Fechas y horas (LocalDate y LocalTime)
- Colecciones (ArrayList)
- Exportación a formato JSON

Se propone trabajar con la siguiente jerarquía de clases e interfaces (detalladas más adelante en el enunciado):

1. **Clase abstracta EventoAstronomico** – representa cualquier fenómeno astronómico observable.
2. **Clase Eclipse** – hereda de EventoAstronomico y agrega atributos específicos de los eclipses.
3. **Clase LluviaMeteoros** – hereda de EventoAstronomico y modela lluvias de meteoros.
4. **Enum TipoEclipse** – define los distintos tipos de eclipse: solar total, solar parcial, solar anular, lunar total, lunar parcial y lunar penumbral.
5. **Interfaz JsonExportable** – permite exportar un evento a formato JSON.
6. **Interfaz EventoRepository** – define las operaciones de consulta y gestión de eventos.
7. **Clase Observatorio** – implementará EventoRepository y funcionará como punto central para acceder a los datos de los eventos.
8. **DAO (EventoAstronomicoDAO)** – proporciona un conjunto inicial de datos de eclipses y lluvias de meteoros, tanto pasados como futuros.

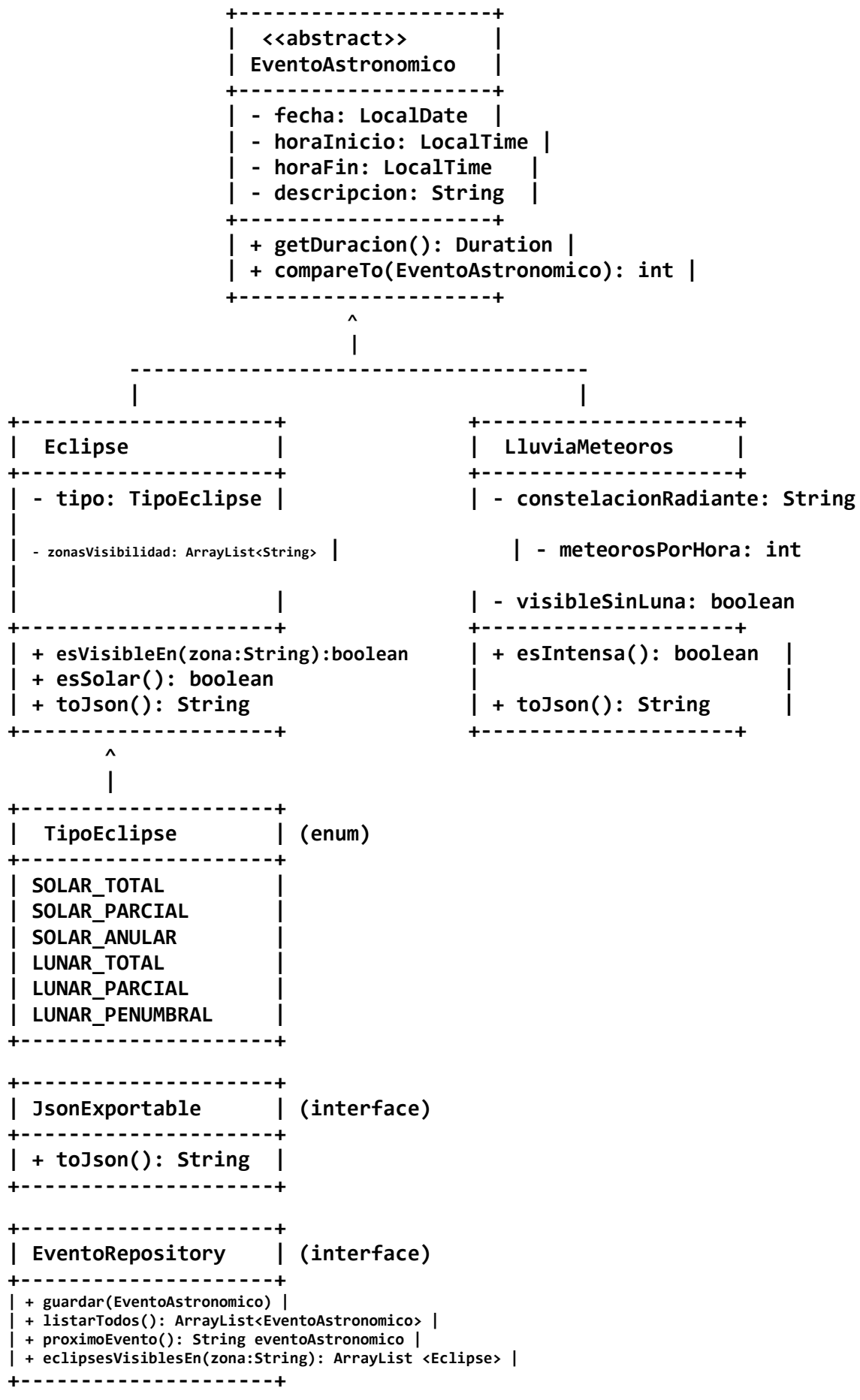
Desarrollo de la clase Principal

- Deberá permitir invocar los métodos del Observatorio y comprobar funcionalidades como:
    - Listar todos los eventos
    - Filtrar eclipses por tipo o año
    - Consultar eventos visibles en una zona concreta
    - Mostrar información de eventos en formato JSON

La implementación de la clase Principal es **libre**, pero debe permitir probar adecuadamente toda la jerarquía de clases y la integración con el DAO.

Reglas y pautas

- Este trabajo se realizará **en grupos durante la clase**.
- Se **permite el uso de herramientas de inteligencia artificial** para asistencia en el desarrollo, pero **debe documentarse claramente** qué se ha consultado y cómo se ha utilizado.
- El proyecto debe ser **claro, coherente y correctamente comentado**, reflejando el diseño orientado a objetos propuesto.


<<abstract>>
EventoAstronomico
- fecha: LocalDate
- horaInicio: LocalTime
- horaFin: LocalTime
- descripcion: String
+ getDuracion(): Duration
+ compareTo(EventoAstronomico): int
Eclipse
- tipo: TipoEclipse
- zonasVisibilidad: ArrayList<String>
+ esVisibleEn(zona:String):boolean
+ esSolar(): boolean
+ toJson(): String
LluviaMeteoros
- constelacionRadiante: String
- meteorosPorHora: int
- visibleSinLuna: boolean
+ esIntensa(): boolean
+ toJson(): String
TipoEclipse
(enum)
SOLAR_TOTAL
SOLAR_PARCIAL
SOLAR_ANULAR
LUNAR_TOTAL
LUNAR_PARCIAL
LUNAR_PENUMBRAL
JsonExportable
(interface)
+ toJson(): String
EventoRepository
(interface)
+ guardar(EventoAstronomico)
+ listarTodos(): ArrayList<EventoAstronomico>
+ proximoEvento(): String eventoAstronomico
+ eclipsesVisiblesEn(zona:String): ArrayList <Eclipse>

1. **Clase abstracta EventoAstronomico**

Atributos

```
private LocalDate fecha;
private LocalTime horaInicio;
private LocalTime horaFin;
private String descripcion;
```

Métodos

```
public Duration getDuracion();
```

Implementar `Comparable<EventoAstronomico>`

Orden natural:

- Por fecha
- Luego por horaInicio

2. **Interfaz `JsonExportable`**

```
public interface JsonExportable {

    String toJson();
}
```

`Nota:` No se permite usar bibliotecas externas (Gson, Jackson, etc.). Debe construirse el JSON manualmente.

3. Enum `TipoEclipse`

```
public enum TipoEclipse {
    SOLAR_TOTAL,
    SOLAR_PARCIAL,
    SOLAR_ANULAR,
    LUNAR_TOTAL,
    LUNAR_PARCIAL,
    LUNAR_PENUMBRAL
}
```

4. **Clase Eclipse**

Hereda de EventoAstronomico

Implementa JsonExportable

Atributos específicos

```
private TipoEclipse tipo;
private ArrayList<String> zonasVisibilidad;
```

Métodos

```
public boolean esVisibleEn(String zona);
```

Salida esperada en formato json:

```
{
  "fecha": "2026-02-17",
  "horaInicio": "10:00",
  "horaFin": "12:30",
  "descripcion": "Eclipse solar anular",
  "tipo": "SOLAR_ANULAR",
  "zonasVisibilidad": ["Chile", "Argentina"]
}
```

Clase `LluviaMeteoros`

Hereda de `EventoAstronomico`

Implementa `JsonExportable`

Atributos específicos

```
private String constelacionRadiante;
private int meteorosPorHora;
private boolean visibleSinLuna;
private int intensidadLimite;
```

Métodos

```
public boolean esIntensa();
// Devuelve true si meteorosPorHora > intensidadLimite.
```

Salida esperada en formato json:

```
{
  "tipoEvento": "LLUVIA_METEOROS",
  "fecha": "2026-08-12",
  "horaInicio": "22:00",
  "horaFin": "04:00",
  "descripcion": "Perseidas",
  "constelacionRadiante": "Perseo",
  "meteorosPorHora": 80,
  "visibleSinLuna": true
}
```

Clase `Observatorio`

Usando el DAO como colección de datos, que además deberás completar para poder guardar elementos.

Implementa EventoRepository

| + guardar(EventoAstronomico) |
| + listarTodos(): ArrayList<EventoAstronomico> |
| + proximoEvento(): String eventoAstronomico |
| + eclipsesVisiblesEn(zona:String): ArrayList <Eclipse> |

`proximoEvento devuelve el String del eventoAstronomico en formato json.`

```
///DAO – colección de elementos a usar en el programa

import java.time.LocalDate;
import java.time.LocalTime;
import java.util.ArrayList;
import java.util.List;

public class EventoAstronomicoDAO {

    private ArrayList<EventoAstronomico> eventos;

    public EventoAstronomicoDAO() {
        eventos = new ArrayList<>();
        cargarDatos();
    }

    private void cargarDatos() {
/*
      private LocalDate fecha;
      private LocalTime horaInicio;
      private LocalTime horaFin;
      private String descripcion;

*/

        // ==============================
        // ECLIPSES
         //private TipoEclipse tipo;
        //private ArrayList <String> zonasVisibilidad;

        // ==============================

        eventos.add(new Eclipse(
                LocalDate.of(2024, 4, 8),
                LocalTime.of(17, 38),
                LocalTime.of(20, 55),
                "Eclipse solar total visible en Norteamérica",
                TipoEclipse.SOLAR_TOTAL,
                List.of("México", "Estados Unidos", "Canadá")
        ));

        eventos.add(new Eclipse(
                LocalDate.of(2026, 2, 17),
                LocalTime.of(9, 56),
                LocalTime.of(14, 27),
                "Eclipse solar anular (anillo de fuego)",
                TipoEclipse.SOLAR_ANULAR,
                List.of("Chile", "Argentina", "Antártida")
        ));

        eventos.add(new Eclipse(
                LocalDate.of(2026, 8, 12),
                LocalTime.of(17, 00),
```

```
        LocalTime.of(19, 30),
        "Eclipse solar total visible en España",
        TipoEclipse.SOLAR_TOTAL,
        List.of("España", "Groenlandia", "Islandia")
  ));

  eventos.add(new Eclipse(
        LocalDate.of(2025, 3, 14),
        LocalTime.of(3, 00),
        LocalTime.of(6, 00),
        "Eclipse lunar total",
        TipoEclipse.LUNAR_TOTAL,
        List.of("Europa", "África", "América")
  ));

  eventos.add(new Eclipse(
        LocalDate.of(2023, 10, 28),
        LocalTime.of(19, 35),
        LocalTime.of(23, 52),
        "Eclipse lunar parcial",
        TipoEclipse.LUNAR_PARCIAL,
        List.of("Europa", "Asia", "África")
  ));

  // ==============================
  // LLUVIAS DE METEOROS
// private String constelacionRadiante;
// private int meteorosPorHora;
// private boolean visibleSinLuna;
// private int intensidadLimite;
  // ==============================


  eventos.add(new LluviaMeteoros(
        LocalDate.of(2025, 8, 12),
        LocalTime.of(22, 0),
        LocalTime.of(4, 0),
        "Perseidas (máximo de actividad)",
        "Perseo",
        80,
        True,50

  ));

  eventos.add(new LluviaMeteoros(
        LocalDate.of(2025, 12, 14),
        LocalTime.of(21, 0),
        LocalTime.of(3, 0),
        "Gemínidas (máximo de actividad)",
        "Géminis",
        120,
        True,50
  ));
```

```
        eventos.add(new LluviaMeteoros(
                LocalDate.of(2025, 10, 21),
                LocalTime.of(23, 0),
                LocalTime.of(5, 0),
                "Oriónidas",
                "Orión",
                25,
                False,50
        ));
    }

    public List<EventoAstronomico> obtenerTodos() {
        return new ArrayList<>(eventos); // devuelve copia para proteger
la lista interna
    }
}
```