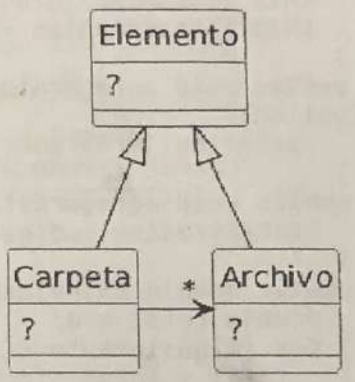
Examen parcial - Orientación a Objetos I - 08/11/2025

Comentarios generales:

- En cada hoja, escriba apellido, nombre, legajo o dni, y número de hoja / el total de hojas entregadas.
- Tenga en cuenta que los 4 ejercicios son **requeridos para aprobar**. No deje ninguno en blanco.
- Debe entregar al menos una hoja con sus datos, aunque no haya completado todo lo pedido.

Ejercicio 1

Se desea construir una plataforma de almacenamiento de archivos en la nube similar a Google Drive. La plataforma brinda espacios personales a cada usuario para que almacene sus carpetas y dentro de ellas, sus archivos. De cada archivo se conoce su nombre y tamaño. Tenga en cuenta que dentro de una carpeta pueden existir archivos, pero no puede haber otras carpetas. Además, a las carpetas se les puede asignar un color para distinguirlas. Los espacios personales de cada usuario tendrán una cantidad límite de carpetas que pueden ser creadas, el cual no podrá ser excedido.



El sistema debe permitir las siguientes funcionalidades:

Agregar un archivo: El usuario debe indicar en qué carpeta de alguno de sus espacios personales se almacenará el archivo. Todos los archivos deben ser almacenados siempre en una carpeta.

Crear una carpeta: Se indica el nombre y color de carpeta a crear en alguno de los espacios personales del usuario. La carpeta se agrega siempre y cuando no supere el límite de carpetas del espacio personal.

Calcular la cantidad de carpetas restantes: se calcula restando al total de carpetas límite con la cantidad de carpetas que tiene actualmente.

Listar contenido de carpeta: se deben retornar todos los archivos en la carpeta seleccionada. El listado deberá estar ordenado por fecha de creación.

Compartir archivo con otro usuario: los archivos pueden compartirse con otro usuario. Para ello, se crea una invitación que contiene un mensaje y que permite al usuario destinatario acceder al archivo.

Listar archivos en los que colabora un usuario: Se debe retornar los archivos que un usuario fue invitado a colaborar, ordenados alfabéticamente por el correo electrónico del usuario dueño.

Tarea:

En el diagrama anterior se muestra un fragmento incompleto del modelo de dominio que fue parcialmente desarrollado. Utilícelo como base para completar el resto del diseño del sistema, completando lo que considere necesario.

Realice un diagrama de clases UML donde identifique las clases del dominio, atributos, y las relaciones con sus roles y cardinalidades según corresponda. No debe incluir comportamiento.

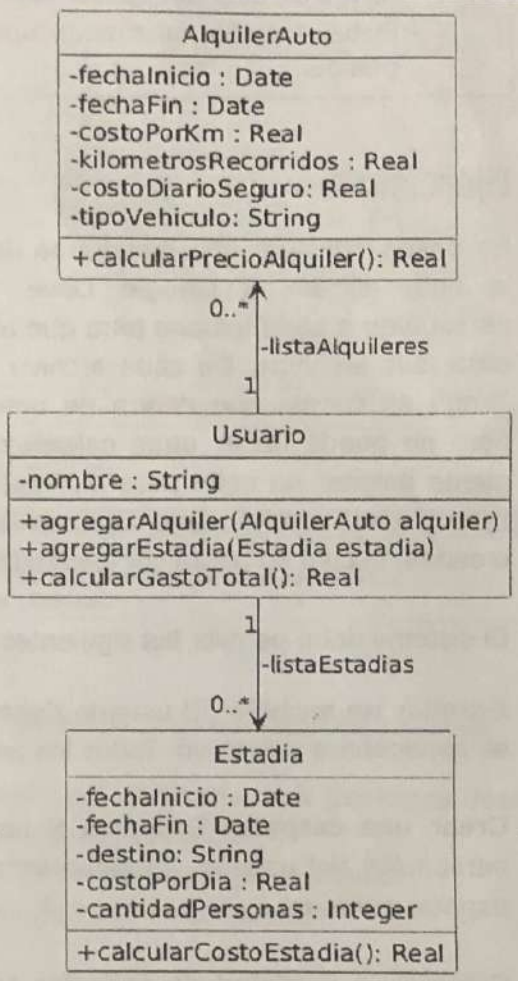
Ejercicio 2

Orientación a Objetos I - 08/11/2025

Se tiene un sistema para registrar los servicios turísticos contratados por un usuario, que incluyen estadías en alojamientos y alquileres de vehículos. Actualmente, la clase Usuario mantiene dos colecciones separadas para cada tipo de servicio, como se muestra en el siguiente diagrama.

```
public class Usuario {
  private String nombre;
  private List<AlquilerAuto> listaAlquileres;
  private List<Estadia> listaEstadias;

  public Usuario(String nombre) {
    this.nombre = nombre;
    this.listaAlquileres = new ArrayList<>();
    this.listaEstadias = new ArrayList<>();
  }
  public void agregarAlquiler(AlquilerAuto
alquiler){
    listaAlquileres.add(alquiler);
  }
  public void agregarEstadia(Estadia estadia) {
    listaEstadias.add(estadia);
  }
  public double calcularGastoTotal() {
    double total = 0;
    for (AlquilerAuto a : listaAlquileres) {
      total = total + a.calcularPrecioAlquiler();
    }
    for (Estadia e : listaEstadias) {
      total = total + e.calcularCostoEstadia();
    }
    return total;
  } }
```




```
public class Estadia {
  private String destino;
  private double costoPorDia;
  private LocalDate fechaInicio, fechaFin;
  private int cantidadPersonas;

  public Estadia(String destino, double costoPorDia, LocalDate fechaInicio, LocalDate
fechaFin, int cantidadPersonas){// asigna cada parámetro a la variable de instancia }

  public double calcularCostoEstadia() {
    long dias = ChronoUnit.DAYS.between(fechaInicio, fechaFin);
    return dias * costoPorDia * cantidadPersonas;
} }
```

```
public class AlquilerAuto {
  private double costoPorKm, kilometrosRecorridos, costoDiarioSeguro;
  private LocalDate fechaInicio, fechaFin;
  private String tipoVehiculo;

  public AlquilerAuto(double costoPorKm, double kilometrosRecorridos, LocalDate
fechaInicio, LocalDate fechaFin, String tipoVehiculo, double costoDiarioSeguro) {
   // asigna cada parámetro a la variable de instancia correspondiente   }

  public double calcularPrecioAlquiler() {
    long dias = ChronoUnit.DAYS.between(fechaInicio, fechaFin);
    return (costoPorKm * kilometrosRecorridos) + (costoDiarioSeguro * dias);
} }
```

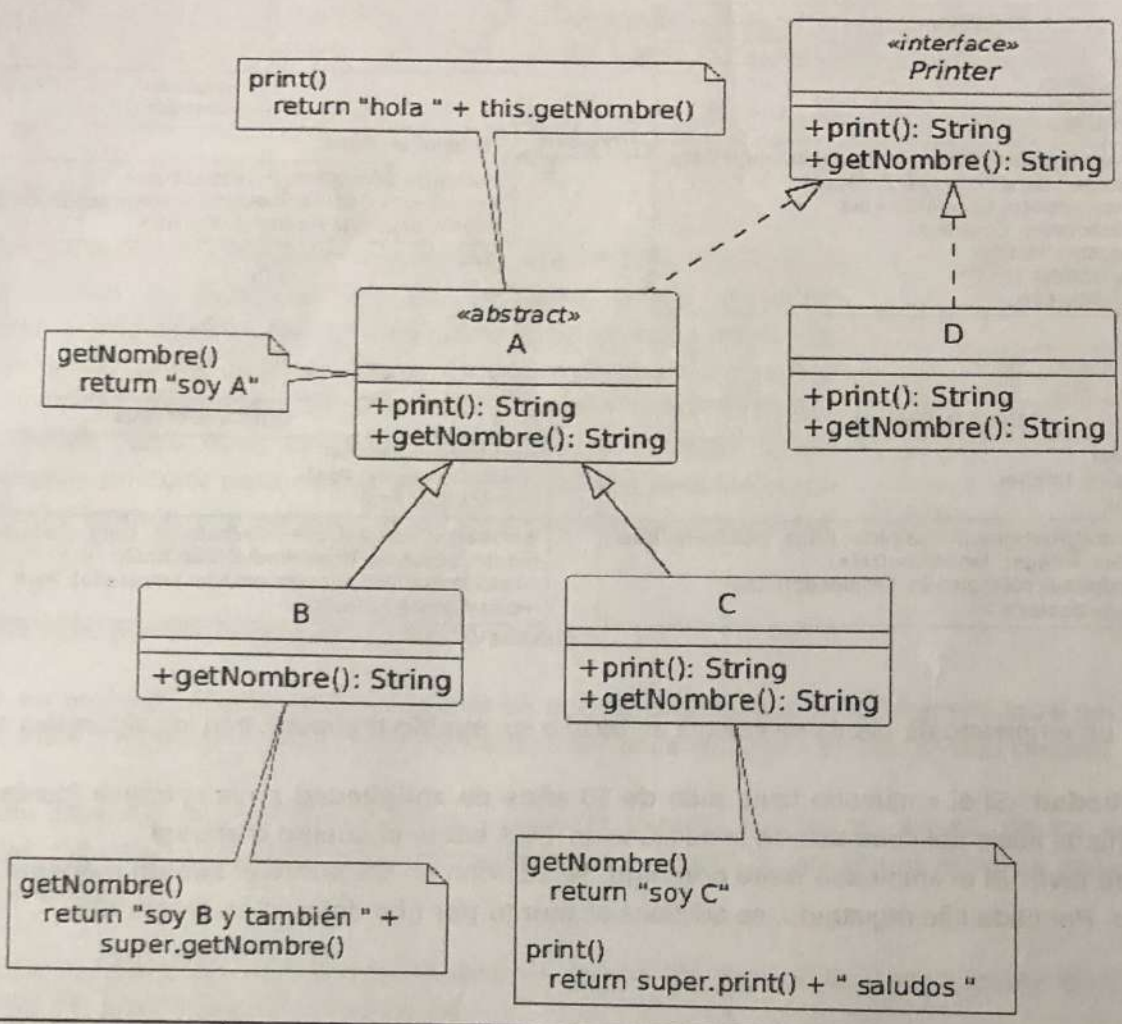
Tareas:

1- Rediseñe lo necesario para permitir que la clase Usuario mantenga una única colección de servicios contratados, evitando duplicación de código y calculando los costos polimórficamente. Aplique los conceptos vistos en la materia. Realice un diagrama de clases UML completo de su diseño (incluyendo los métodos).

2- Reimplemente en Java respetando el nuevo diseño.

Ejercicio 3

Dado el siguiente modelo, y los fragmentos de códigos mostrados, marque la respuesta para cada una de las preguntas enunciadas. Sólo una opción es la correcta.



Observando el diagrama y dado el siguiente fragmento de código, responda (A) y (B):

```
? o1 = new ?()
o1.print();
o1.getNombre();
```

A- Indique la opción que incluye **todos** los tipos válidos con los que puede declararse la variable o1 para que el fragmento anterior sea válido en tiempo de compilación.

1. A, B, C, D
2. Printer, A, C, D
3. Printer, A, B, C, D
4. Printer, A, B, C

B- Suponiendo que o1 fue declarada de tipo Printer, indique la opción que incluye **todos** los tipos que pueden instanciarse y asignarse a la variable o1 en el fragmento anterior.

1. Printer, A, B, C, D
2. C, D
3. Printer, A, B, C
4. B, C, D

C- Observando el diagrama, indique qué texto retorna el siguiente fragmento de código:

```
Printer p = new B();
p.print();
```

1. hola soy A y también soy B
2. hola soy A
3. hola soy B y también soy A
4. hola soy B

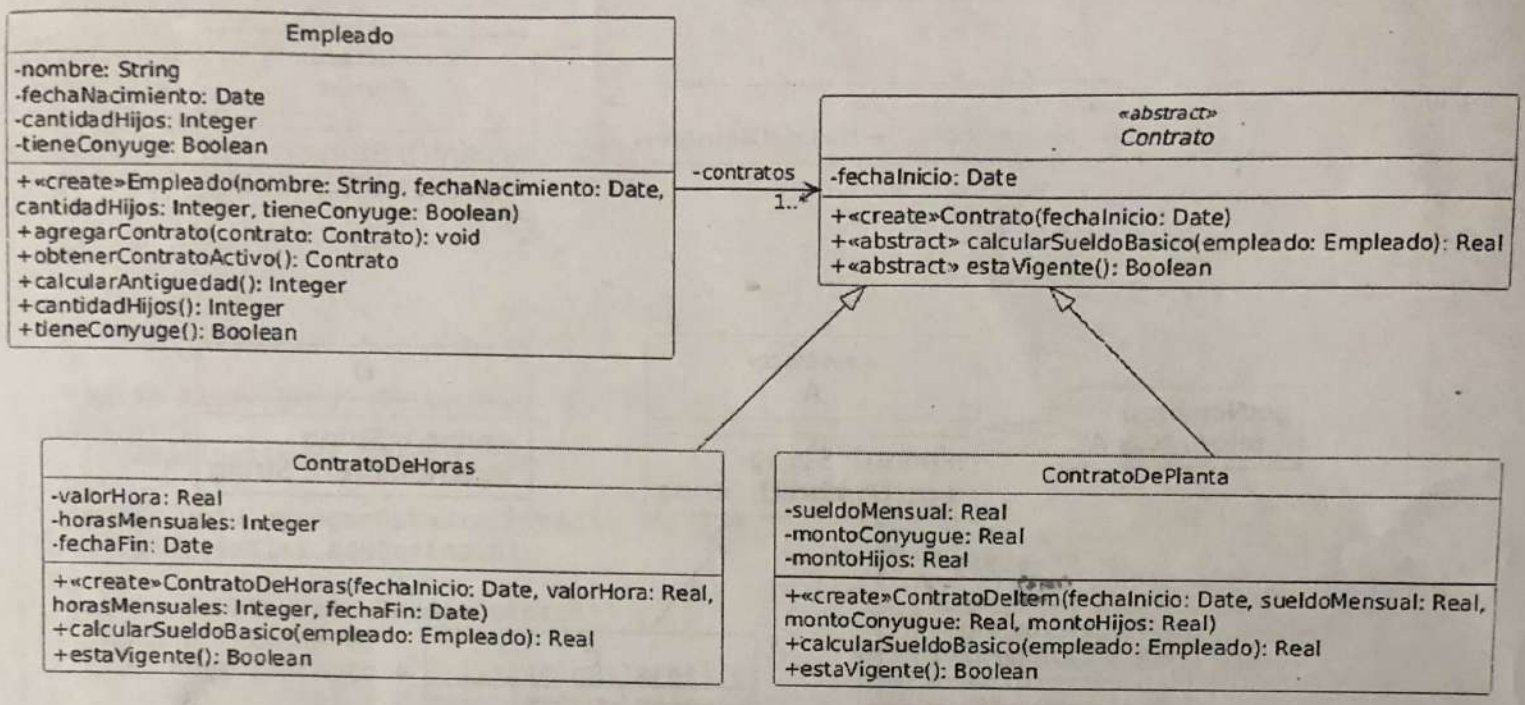
D- Observando el diagrama, indique qué texto retorna el siguiente fragmento de código:

```
A a = new C();
a.print();
```

1. hola soy A
2. hola soy C saludos
3. hola soy C
4. hola soy A saludos

Ejercicio 4 — Orientación a Objetos I - 08/11/2025

Una empresa tiene un sistema que procesa los sueldos de su nómina de empleados, cuyo diseño se muestra en el siguiente diagrama de clases UML:



El sueldo de un empleado de planta se calcula en base a su **sueldo mensual**, con los siguientes ajustes:

- **Antigüedad:** Si el empleado tiene **más de 10 años de antigüedad** en la empresa (contados desde la fecha de inicio del contrato), se le adiciona un **25% sobre el sueldo mensual**.
- **Estado civil:** Si el empleado **tiene cónyuge**, se adiciona un **5% sobre el sueldo mensual**.
- **Hijos:** Por cada hijo registrado, se adiciona el **monto por hijo** definido en el contrato.

Tareas:

1. Defina el constructor de Empleado. Observe que Contrato ya tiene sus constructores definidos.
2. Instancie los siguientes empleados, utilizando la siguiente información:
   - Juan Pérez - Casado con 2 hijos (Nacido: 10/05/1985)
     - 01/06/2020-31/01/2023: Contrato por horas ($900/hora, 160h/mes)
     - Actualmente: De planta desde 01/02/2023 ($120,000 + $6,000 cónyuge + $10,000 monto por cada hijo)
   - Ana García - Soltera con 1 hijo (Nacida: 15/08/1990)
     - 01/07/2022-30/09/2024: De planta ($80,000 + $8,000 hijos)
     - Actualmente: Por horas desde 01/10/2024 ($1,100/hora, 150h/mes, vence 15/12/2025)
3. Diseñe los casos de prueba para el método **calcularSueldoBasico** de **ContratoDePlanta** teniendo en cuenta los conceptos de valores de borde y particiones equivalentes vistos en la materia

Ayuda: Para instanciar un objeto LocalDate, puede usar `LocalDate.of(year, month, day)`