ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
PROGRAMACIÓN ORIENTADA A OBJETOS
EXAMEN PRIMERA EVALUACIÓN 2026 - 1T

Nombre: _____ **Paralelo: _____**

1. [5 puntos] Responda Verdadero o Falso según corresponda.

Cuando se crea un objeto de una clase hija, siempre se ejecuta primero un constructor de la clase padre antes de completar la construcción del objeto hijo.	
Una constante declarada con la palabra reservada final puede recibir un nuevo valor en cualquier método de la clase siempre que conserve el mismo tipo de dato.	
Los objetos creados mediante la palabra reservada new se almacenan en el Heap de memoria.	
El polimorfismo permite que una referencia de la clase padre almacene objetos de cualquiera de sus clases hijas.	
Un método declarado como static puede acceder directamente a los atributos y métodos no estáticos de la clase usando this	

2. [10 puntos] Análisis de código

El siguiente código no tiene errores de compilación.

```
package com.example;
public class  GrupoAlimentario {
    protected int calorias;
    protected double grasas;

    public GrupoAlimentario() {
        this(100, 2.5);
        System.out.println("Grupo base");
    }

   public GrupoAlimentario(int calorias, double grasas)
   {
        this.calorias = calorias;
        this.grasas = grasas;
        System.out.println(calorias);
    }
}
```

```
package com.example;
public class Lacteo extends GrupoAlimentario {

    public Lacteo() {
        System.out.println("Lácteo");
    }

    public Lacteo(int calorias, double grasas){
        super(calorias, grasas);
        System.out.println("Lácteo personalizado");
    }
}
```

```
package com.example;

public class Principal {
public static void main(String[] args) {

        // Reemplace esta línea por cada opción

        }
}
```

Escriba la salida, al ejecutar cada línea de forma independiente en el método main de la clase Principal.

Línea de código	Salida
new GrupoAlimentario();	
new GrupoAlimentario(250, 8.0);	
Lacteo l= new Lacteo();	
GrupoAlimentario alimento = new Lacteo(180, 5.5);	

3. [10 puntos] Este código contiene 5 errores de compilación. Para cada error indica:

- La línea donde se encuentra el error (aproximado).
- Una breve descripción del error.

```
1   package com.example;
2   public class EquipoMedicion {
3     protected String codigo;
4     protected String marca;
5   public EquipoMedicion(String codigo, String marca) {
6       this.codigo = codigo;
7       this.marca = marca;
8   }
9   public double calcularCostoMantenimiento() {
10        return 50.0;
11    }
12    public String obtenerResumen() {
13        return codigo + " - " + marca;
14    }
15  }
```

```
16  package com.example;

17  public class SensorTemperatura extends EquipoMedicion {
18    private double temperaturaMaxima;

19    public SensorTemperatura(String codigo, String marca, double temperaturaMaxima) {
20        super(codigo, marca);
21        this.temperaturaMaxima = temperaturaMaxima;
22    }

23    @Override
24    public int calcularCostoMantenimiento() {
25        return 80;
26    }

27    public void calibrar(double valorReferencia) {
28        System.out.println("Calibrando con referencia: " + valorReferencia);
29    }

30    @Override
31    public void calibrar(int valorReferencia) {
32        System.out.println("Calibrando con referencia entera: " + valorReferencia);
33    }
34  }
```

```
35  package com.example;
36  public class MedidorPresion extends EquipoMedicion {
37     private double presionMaxima;
38     public MedidorPresion(String codigo, String marca, double presionMaxima) {
39         this.presionMaxima = presionMaxima;
40         super(codigo, marca);
41     }
42    @Override
43    public String obtenerResumen() {
44        return codigo + " - " + marca + " [" + super.presionMaxima + "]";
45    }
46  }
```

```
47  package com.example;
48  import java.util.ArrayList;
49  public class Main {
50     public static void main(String[] args) {
51        ArrayList<EquipoMedicion> equipos = new ArrayList<>();
52        equipos.add(new SensorTemperatura("T01", "Bosch", 120.5));
53        equipos.add(new MedidorPresion("P01", "Siemens", 300.0));
54
55        EquipoMedicion equipo = new SensorTemperatura("T02", "Fluke", 100.0);
56        equipo.calibrar(25.0);
57
58        ((SensorTemperatura) equipos.get(0)).calibrar(25.0);
59
60     }
61  }
```

4. [20 puntos] Dibuje el diagrama de clases del siguiente caso de estudio:

Una galería de arte busca desarrollar un sistema para organizar sus exposiciones y obras. El sistema necesita representar a los artistas, las obras expuestas y los eventos organizados por la galería.

La galería posee diversas obras de arte. Cada obra tiene un título, el año de creación, el artista y un precioBase. Además, cada obra debe ser capaz de calcular y retornar su precio final, a partir del precio base, siguiendo una lógica que varía según el tipo de obra.

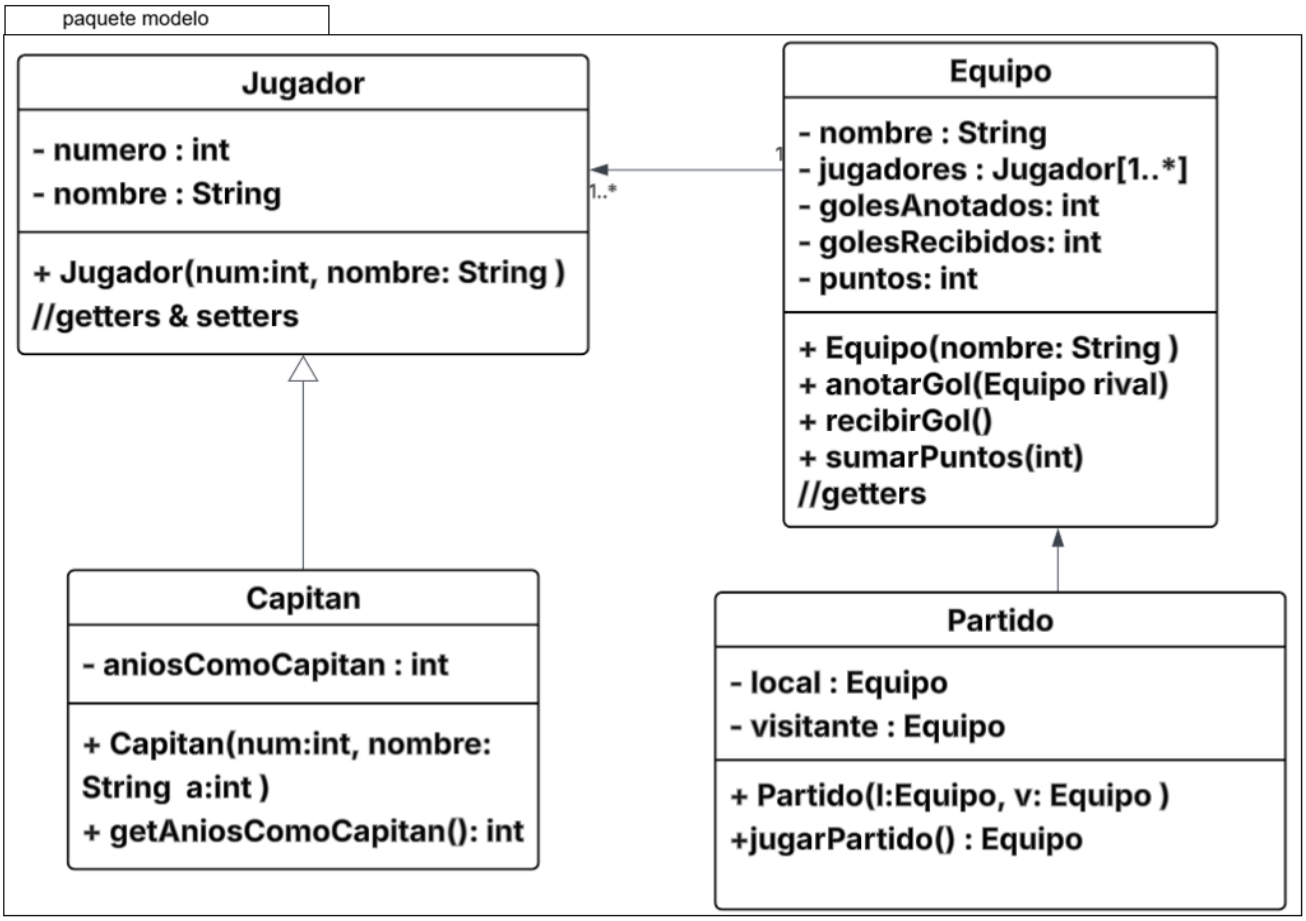
En la galería se exhiben dos tipos específicos de obras: pinturas y esculturas. Las pinturas deben registrar la técnica utilizada (como óleo, acuarela o acrílico.), y su precio final se calcula aumentando un 15% sobre el precio base. Por otro lado, las esculturas deben registrar el material con el que están hechas y su peso en kilogramos. En este caso, el precio final se incrementa en un 25% si la escultura pesa más de 20 kg, o en un 10% si pesa menos o igual a ese valor.

Cada obra está relacionada con un artista, del cual se debe registrar su nombre, nacionalidad y fecha de nacimiento. Un artista puede tener múltiples obras registradas en el sistema.

Además, la galería organiza exposiciones temporales, que agrupan un conjunto de obras en un periodo determinado. Cada exposición tiene un nombre, una fecha de inicio y una fecha de fin. Es necesario poder calcular el valor total de una exposición, que corresponde a la suma de los precios finales de todas las obras que forman parte de ella.

5. [55 puntos] Tema de Desarrollo

Se le ha pedido desarrollar una aplicación para simular la primera fecha de un campeonato de fútbol. Durante la simulación se generarán aleatoriamente los goles de cada equipo y, al finalizar todos los encuentros, se deberá obtener una lista con los equipos que ganaron su primer partido. Se le proporciona el siguiente diagrama de clases.



La clase Jugador está completamente implementada y representa un jugador de fútbol.

1.- Clase Capitan

Esta clase representa el jugador que es el capitán del equipo. Implemente la clase de acuerdo con el diagrama.

A. Definición de paquete.
B. Definición de la clase.
C. Atributo aniosComoCapitan de tipo int.
D. Constructor para inicializar las variables de instancia.
E. get de la variable de instancia.

2.-Clase Equipo

Esta clase representa un equipo participante en el campeonato. En esta clase **solamente** implemente el método **anotarGol(Equipo rival)**. Este método deberá realizar las siguientes acciones:

1. Mostrar el mensaje “Gol de“ + nombre del equipo
2. Incrementar en uno los goles anotados del equipo.
3. Registrar el gol recibido por el equipo **rival** utilizando el método recibirGol().

4. Seleccionar aleatoriamente un jugador de la lista de jugadores del equipo.

   Para generar un número aleatorio puede utilizar el siguiente código:

```
// Genera un número entero aleatorio entre 0 y 9
int numero = (int) (Math.random() * 10);
```

5. Mostrar el nombre del jugador seleccionado indicando que es el anotador del gol.
6. Si el jugador seleccionado es un objeto de la clase Capitan, mostrar en un mensaje los años que lleva como capitán.

Ejemplos

¡Gol de Tigres FC! Anotador: Andrés López Capitán desde hace 4 años.	¡Gol de Halcones FC! Anotador: Diego Herrera

Asuma que los demás métodos ya están implementados (NO LOS ESCRIBA):

- Método recibirGol(), el cual incrementa en uno los goles recibidos.
- Método sumarPuntos(int puntos), el cual incrementa los puntos del equipo.
- Getters para sus atributos

3.- Clase Partido

Representa un encuentro entre dos equipos.

En esta clase **solamente** implemente el método jugarPartido()

Este método deberá:

1. Mostrar los nombres de los equipos que se enfrentan: Local vs Visitante
2. Generar aleatoriamente la cantidad de goles del equipo local (entre **0 y 4**).
3. Generar aleatoriamente la cantidad de goles del equipo visitante (entre **0 y 4**).
4. Registrar los goles del equipo local llamando al método anotarGol(visitante) tantas veces como indique el número aleatorio generado.
5. Registrar los goles del equipo visitante de la misma manera explicada en el literal anterior.
6. Al finalizar el partido, comparar la cantidad de goles obtenida por ambos equipos y sumar los puntos correspondientes (usar el método **sumarPuntos**) de acuerdo a los resultados:
   - Equipo ganador: **3 puntos**.
   - Empate: **1 punto** para cada equipo.
   - Equipo perdedor: **0 puntos**.
7. Mostrar el resultado final del encuentro. (Ver ejemplo de salida)
8. La función debe retornar el equipo ganador, si hubo un empate debe retornar null.

4. Clase Main – OPCIONAL (5 puntos)

A. Implemente el método estático jugarFecha(ArrayList<Partido> partidos) que **recibe** una lista de tipo Partido y **devuelve** una lista con los equipos que ganaron los partidos. **Recuerde que el método jugarPartido devuelve el equipo ganador o null si fue empate**.

Ejemplo de salida al ejecutar jugarFecha

```
============= PRIMERA FECHA DEL CAMPEONATO =============

Partido: Tigres FC vs Halcones FC

¡Gol de Tigres FC!
Anotador: Carlos Pérez

¡Gol de Tigres FC!
Anotador: Andrés López
Capitán desde hace 4 años.

¡Gol de Halcones FC!
Anotador: Diego Herrera

Resultado:
Tigres FC 2 - 1 Halcones FC
Ganador: Tigres FC


Partido: Leones FC vs Delfines FC

¡Gol de Leones FC!
Anotador: Juan Torres

¡Gol de Delfines FC!
Anotador: Marco Silva
Capitán desde hace 2 años.

Resultado:
Leones FC 1 - 1 Delfines FC
Empate
…
```