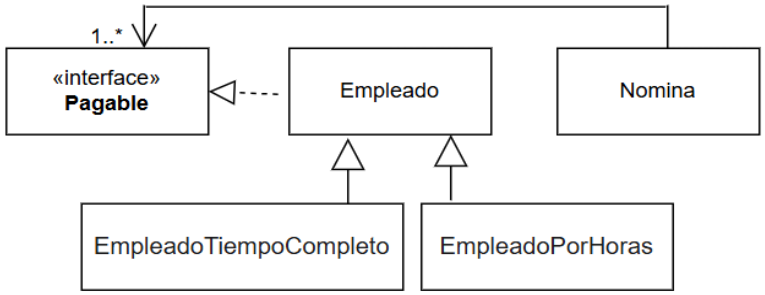
ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
PROGRAMACIÓN ORIENTADA A OBJETOS
EXAMEN TERCERA EVALUACIÓN - 2025 - 2T

NOMBRE: **PARALELO:**

TEMA 1. Seleccione la respuesta correcta. (18 puntos)

Se tiene un sistema que maneja el pago para los empleados de una empresa con el diagrama de clases:



El código ya existe y funciona, pero tiene decisiones de diseño que se analizan.

1. El método calcularSalario() está sobrescrito en las clases hijas. Si se invoca así:

```
Empleado e= new EmpleadoPorHoras();
e. calcularSalario()
```

¿Qué versión del método se ejecuta?

A. La versión definida en Empleado
B. La versión correspondiente al tipo real del objeto
C. La versión que se ejecuta primero en memoria
D. Ninguna, porque el método es abstracto en Empleado.

2. Si se modifica la definición del método calcularSalario() en la clase Empleado y el método es declarado como static ¿Cuál es la principal consecuencia de esta decisión?

A. Mejora el rendimiento del programa
B. Se pierde el polimorfismo
C. Se evita la herencia
D. Se mejora la encapsulación

3. La clase Empleado incluye implementación específica para el método calcularPagoPorHoras(). Desde el punto de vista de POO, esto implica que:

A. La abstracción es correcta
B. Se viola el principio de abstracción
C. Se mejora la reutilización
D. No tiene ningún impacto

4. Algunos atributos de Empleado son públicos. Seleccione todas las consecuencias de esta decisión.

☐ Se pierde control sobre el estado del objeto
☐ Se rompe la encapsulación
☐ El código deja de compilar
☐ Se dificulta el mantenimiento futuro

5. Usar herencia e interfaces al mismo tiempo siempre es una mala práctica. Verdadero Falso

6. Si un método no está sobrescrito, no puede existir polimorfismo. Verdadero Falso

TEMA 2. El siguiente código contiene 4 errores de compilación. Para cada error indique la línea, clase y explique el motivo. (12 puntos)

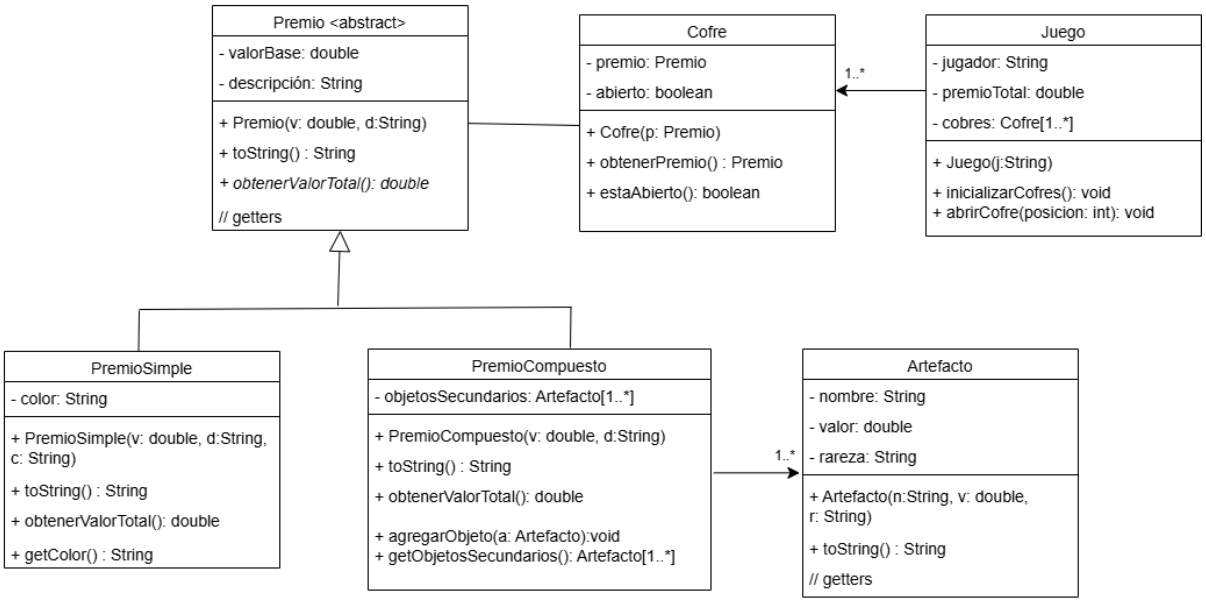
```
1  package com.example;
2
3  interface Prestamo {
4    int diasMaximos();
5
6    default boolean esUrgente() {
7        return diasMaximos() <= 3;
8    }
9
10   static String politicaGeneral() {
11       return "Renovación permitida una vez.";
12   }
13}
14
15 abstract class Material implements Prestamo {
16   protected String titulo;
17
18   public Material(String titulo) {
19       this.titulo = titulo;
20   }
21
22   public void mostrarFicha() {
23       System.out.println(titulo + " | urgente=" +
esUrgente());
24   }
25}
26
27 class Libro extends Material {
28   public Libro(String titulo) {
29       super(titulo);
30   }
31
32   @Override
33   public int diasMaximos(int extra) {
34       return 7;
35   }
36
37   protected boolean esUrgente() {
38       return false;
39   }
40 }
```

```
class Revista extends Material {
    public Revista(String titulo) {
        super(titulo);
    }

    @Override
    public int diasMaximos() {
        return 2;
    }

    boolean esUrgente(String valor) {
        return valor.equals("Si");
    }

    @Override
    public static String politicaGeneral() {
        return "Política especial de revista.";
    }
}

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

        Prestamo p = new Revista("Ciencia Hoy");

        System.out.println(p.esUrgente("Sí"));

        System.out.println(Prestamo.politicaGeneral());
    }
}
```

TEMA 3. DESARROLLO

Se requiere implementar un juego de consola en el que el jugador interactúa con 5 cofres numerados (1 a 5). Cada cofre contiene un premio generado aleatoriamente al inicio del juego. Los premios pueden ser simples (valor directo) y premios compuestos (valor base + colección de artefactos secundarios). El jugador acumula puntaje al abrir cofres, y al finalizar debe mostrar su ganancia total.

Se le proporciona el siguiente diagrama de clases:



De acuerdo al diagrama implementar lo solicitado a continuación:

1. Clase Premio. Esta clase está parcialmente implementada. Complete lo siguiente en esta clase:
    - Definición de la clase.
    - Método abstracto obtenerValorTotal().
    - Los demás miembros de la clase ya están implementados y los puede utilizar:
        - Atributos privados valorBase y descripcion.
        - Constructor para inicializar variables de instancia.
        - Getters para sus atributos.
        - método toString que retorna:

          Descripción [descripcion] | Valor base: [valorBase]

2. Clase PremioSimple (NO IMPLEMENTAR)
    - Atributo color
    - Constructor para inicializar las variables de instancia
    - Getter para color
    - obtenerValorTotal() retorna únicamente el valor de la variable valorBase.
    - Método toString() que retorna (implementa reutilización de código):
      Premio Simple: Descripción [descripcion] | Valor base: [valorBase] | Color: [color]
3. Clase PremioCompuesto (Implementar esta clase completamente)
    - Atributo: Lista privada objetosSecundarios de tipo ArrayList<Artefacto>.
    - Constructor que recibe valorBase, descripción e inicializa la lista de artefactos.
    - Getter para la lista objetosSecundarios.
    - método obtenerValorTotal() que retorna valorBase + la suma de los valores de los artefactos.
    - Método agregarObjeto(a: Artefacto) que recibe un objeto de tipo Artefacto para agregar a la lista objetosSecundarios.
    - Método toString() que retorna (implementar reutilización de código):

      Premio Compuesto: Descripción [descripcion] | Valor base: [valorBase] | Contiene [N] artefactos

4. Clase Artefacto (NO IMPLEMENTAR)
    - Atributos privados nombre, valor, rareza.
    - Constructor y getters.
    - toString que devuelve: [nombre] ([rareza]): [valor] Ejemplo:

      Varita Mágica (Épico): 60.0

5. Clase Cofre (NO IMPLEMENTAR)
    - Variable de instancia de tipo Premio.
    - Constructor para inicializar la variable de instancia.
    - obtenerPremio(), cambia la variable abierto a true y retorna la referencia al Premio almacenado en ese cofre.
    - estaAbierto() retorna true si el cofre ya está abierto (valor de variable abierto).

6. Clase Juego

Esta clase está parcialmente implementada. Desarrolle sólo los métodos abrirCofree y main.

Asuma que el constructor de la clase ya está implementado y asigna el contenido a la variable jugador. No escriba este método, sólo utilícelo.

El método inicializarCofres() también está implementado y llena la lista de cofres generando aleatoriamente 5 premios (mezcla de PremioSimple y PremioCompuesto). No escriba este método, sólo utilícelo.

Los getters de las variables de instancias también están implementados y disponibles para ser utilizados.

El método mostrarPremioTotal() también está implementado e imprime el premio acumulado con formato:

PREMIO TOTAL ACUMULADO: [premioTotal] monedas.

Implemente los siguientes métodos:

- abrirCofre(posicion: int)
    - Obtiene el premio del cofre que se encuentra en la posición recibida en el parámetro.
    - Muestra la información del premio contenido. No olvide el uso de toString().
    - Acumula el valor total del premio en premioTotal.
    - Si el premio es PremioCompuesto, realiza casting explícito para:
        - Acceder a la lista de artefactos.
        - Imprimir el detalle de los artefactos que contiene junto con el valor total del premio de ese cofre.

      - [nombre1] ([rareza1]): [valor1]

      - [nombre2] ([rareza2]): [valor2]

      ...

      Valor total del premio: [valorTotal]

- Método main en la clase Juego
- Solicitar el nombre del jugador
- Crear la instancia de Juego
- Llame al método inicializarCofres()
- Mientras el premioTotal no haya superado las 300 monedas
    - Mostrar los cofres numerados (los que no han sido abiertos).
    - Solicitar al usuario el número de cofre. Asuma que el usuario va a ingresar un número válido.
    - Abrir el cofre de acuerdo con la opción del usuario (usando el método ya implementado)
    - Mostrar el premio total acumulado (usando el método ya existente)
- Mostrar el mensaje “No puede abrir más cofres.” cuando haya acumulado, al menos, 300 monedas.
- Escribir en el archivo resultados.txt el nombre del jugador, total de cofres abiertos y total de premio acumulado con el siguiente formato:

```
Pedro,3,325.5
```

Ejemplo de ejecución:

```
¡Bienvenido a Cofres del Destino!
Ingrese nombre de jugador:  Gladys
Cofres disponibles: [1] [2] [3] [4] [5]
Elige un cofre para abrir (1-5): 3

Premio Compuesto: Cofre Ancestral | Valor base: 75.0 | Contiene 3 artefactos:
  - Amuleto de Fuego (Raro): 22.5
  - Capa de Sombras (Épico): 45.0
  - Daga Afilada (Común): 8.0
  Valor total del premio: 150.5

 PREMIO TOTAL ACUMULADO: 150.5 monedas

Cofres disponibles: [1] [2] [4] [5]
Elige un cofre para abrir (1-5): 1

Premio Simple: Bolsa de Monedas | Color: Dorado | Valor: 30.0

 PREMIO TOTAL ACUMULADO: 180.5 monedas

Cofres disponibles: [2] [4] [5]
Elige un cofre para abrir (1-5): 5

Premio Compuesto: Cofre Encantado | Valor base: 50.0 | Contiene 2 artefactos:
  - Varita Mágica (Épico): 60.0
  - Escudo Rúnico (Raro): 35.0
  Valor total del premio: 145.0

PREMIO TOTAL ACUMULADO: 325.5 monedas

No puede abrir más cofres.
```