CESUR
Tu Centro Oficial de FP

EXAMEN PARCIAL – UNIDADES 3 Y 4		
TRIMESTRE: PRIMERO		**Fecha:**
CICLO: Desarrollo de Aplicaciones Web. *MÓDULO: Programación*	*CURSO: 1º* *Turno: Mañana*	*CALIFICACIÓN:*
Nombre:	***Apellidos:***	

Instrucciones: *Esta prueba tiene como finalidad evaluar los aprendizajes de Programación. Lee atentamente y responde escribiendo el código más adecuado.*

Si las instrucciones no se siguen como se especifican el examen no será evaluado

PARTE PRÁCTICA. TIPO C.

- El examen práctico tiene una puntuación máxima de 10 puntos.
- Para superar la parte práctica se requiere alcanzar un mínimo de 5 puntos.

CESUR
Tu Centro Oficial de FP

1. **(3 puntos)** Diseñar un sistema en Java para gestionar diferentes tipos de empleados dentro de una empresa. El sistema debe manejar tres modalidades de contrato distintas: empleado a tiempo completo, empleado por hora, y empleado freelancer.

   Clase Empleado

   - Crea una clase base llamada Empleado con los siguientes atributos:
     - nombre (de tipo String): el nombre del empleado.
     - salarioBase (de tipo double): el salario base del empleado.
   - Proporciona **getters** y **setters** para ambos atributos.
   - Implementa los siguientes métodos:
     - calcularSalario(): calcula y devuelve el salario del empleado basado en el salarioBase.
     - trabajar(): imprime un mensaje genérico como "El empleado está trabajando.".
     - infoEmpleado(): imprime la información básica del empleado, incluyendo el nombre y el salario base.

   Subclases

   - **EmpleadoTiempoCompleto**:
     - Sobrescribe calcularSalario() para incluir un bono fijo de $500 al salario base.
     - Sobrescribe trabajar() para imprimir "El empleado de tiempo completo está trabajando en un horario regular.".
   - **EmpleadoPorHora**:
     - Sobrescribe calcularSalario() para calcular el salario basado en una tarifa por hora y el número de horas trabajadas.
     - Sobrecarga calcularSalario(int horasTrabajadas, double tarifaPorHora) para aceptar estos parámetros y calcular el salario correspondiente.
     - Sobrescribe trabajar() para imprimir "El empleado por hora está trabajando según las horas asignadas.".
   - **EmpleadoFreelancer**:
     - Agrega un atributo numeroDeProyectos (de tipo int) y un método **getter** y **setter** para este atributo.
     - Sobrescribe calcularSalario() para calcular el salario basado en una tarifa fija por proyecto (por ejemplo, $200 por proyecto).
     - Sobrescribe trabajar() para imprimir "El freelancer está trabajando en proyectos específicos.".

   En el método principal, realiza las siguientes acciones:

- Crea instancias de cada tipo de empleado (EmpleadoTiempoCompleto, EmpleadoPorHora, EmpleadoFreelancer) y asigna valores específicos para cada atributo utilizando los **setters**.
- Muestra la información básica de cada empleado utilizando el método infoEmpleado().
- Llama al método trabajar() en cada empleado para mostrar el comportamiento sobrescrito.
- Llama al método calcularSalario() en cada empleado para calcular y mostrar su salario total.
- Para el caso del empleado por hora, utiliza la sobrecarga de calcularSalario(int horasTrabajadas, double tarifaPorHora) para calcular su salario basado en diferentes escenarios de horas y tarifas.

Probar con este main:

```
public static void main(String[] args) {
    // Crear instancias de empleados
    EmpleadoTiempoCompleto empleadoTiempoCompleto = new
     EmpleadoTiempoCompleto("Juan Pérez", 2000);
    EmpleadoPorHora empleadoPorHora = new EmpleadoPorHora("María
                                                    Gómez");
    EmpleadoFreelancer empleadoFreelancer = new
     EmpleadoFreelancer("Pedro López", 4);

    // Mostrar información y comportamiento
    empleadoTiempoCompleto.infoEmpleado();
    empleadoTiempoCompleto.trabajar();
    System.out.println("Salario total: $" +
     empleadoTiempoCompleto.calcularSalario());
    System.out.println();

    empleadoPorHora.infoEmpleado();
    empleadoPorHora.trabajar();
    System.out.println("Salario total (tarifa de $20.0/hora, 40 horas):
                    €" + empleadoPorHora.calcularSalario(40, 20.0));
    System.out.println("Salario total (tarifa de $25.0/hora, 30 horas):
                    €" + empleadoPorHora.calcularSalario(30, 25.0));
    System.out.println();

    empleadoFreelancer.infoEmpleado();
    empleadoFreelancer.trabajar();
    System.out.println("Salario total (4
proyectos): $" +
     empleadoFreelancer.calcularSalario());
}
```

Ejemplo:

```
Información del empleado:
Nombre: Juan Pérez
Salario base: 2000.0€
El empleado de tiempo completo está trabajando en un horario regular.
Salario total: 2500.0€

Información del empleado:
Nombre: María Gómez
Salario base: 0.0€
El empleado por hora está trabajando según las horas asignadas.
Salario total (tarifa de $20.0/hora, 40 horas): 800.0€
Salario total (tarifa de $25.0/hora, 30 horas): 750.0€

Información del empleado:
Nombre: Pedro López
Salario base: 0.0€
El freelancer está trabajando en proyectos específicos.
Salario total (4 proyectos): 800.0€
```

2. **(3 puntos)** Desarrolla un sistema en Java para gestionar productos de diferentes categorías (electrónicos, ropa y alimentos). El sistema debe permitir calcular los precios finales de los productos según características específicas de cada tipo, como garantía, tamaño o fecha de vencimiento.

Clase Producto:

- **Atributos**:
  - nombre (String): El nombre del producto (ej. "Televisor", "Camiseta").
  - precio (double): El precio base del producto.
  - cantidadStock (int): La cantidad disponible de ese producto en el inventario.
- **Métodos**:
  - **calcularPrecio()**: Devuelve el precio del producto, que por ahora es igual al precio base.
  - **mostrarDetalles()**: Muestra todos los detalles del producto: nombre, precio y cantidad en stock.
  - **disponibilidad()**: Imprime si el producto está disponible en stock o no (si la cantidad en stock es mayor que 0, está disponible).

ProductoElectronico:

- **Atributo adicional**:
  - garantia (int): La garantía del producto, expresada en años.

- **Métodos**:
  - **calcularPrecio()**: Sobrescribe el método calcularPrecio() para aplicar un descuento del 10% si la garantía del producto es mayor a 2 años. Si no, se deja el precio tal cual.
  - **mostrarDetalles()**: Sobrescribe el método mostrarDetalles() para incluir también el valor de la garantía.

ProductoRopa:

- **Atributo adicional**:
  - tamaño (String): El tamaño de la ropa, que puede ser "S", "M", "L", o "XL".
- **Métodos**:
  - **calcularPrecio()**: Sobrescribe el método calcularPrecio() para aplicar un aumento del 10% al precio si el tamaño del producto es "L" o "XL". Si el tamaño es "S" o "M", el precio permanece igual.

- **mostrarDetalles()**: Sobrescribe el método mostrarDetalles() para incluir también el tamaño del producto.

ProductoAlimenticio:

- **Atributo adicional**:
  - fechaVencimiento (String): La fecha de vencimiento del producto. Puede ser una cadena como "próximo mes" para indicar productos cercanos a la fecha de vencimiento.
- **Métodos**:
  - **calcularPrecio()**: Sobrescribe el método calcularPrecio() para aplicar un descuento del 20% al precio si el producto está cerca de su fecha de vencimiento (por ejemplo, si fechaVencimiento es igual a "próximo mes").
  - **mostrarDetalles()**: Sobrescribe el método mostrarDetalles() para incluir también la fecha de vencimiento.

Probar con este main:

```
public static void main(String[] args) {
    // Crear productos
    Producto producto1 = new ProductoElectronico("Televisor", 300.0, 10, 3);
    Producto producto2 = new ProductoRopa("Camiseta", 20.0, 15, "L");
    Producto producto3 = new ProductoAlimenticio("Leche", 2.5, 50,
                                                 "próximo mes");

    // Mostrar detalles y calcular precios
    System.out.println("Detalles del Producto 1 (Electrónico):");
    producto1.mostrarDetalles();
    System.out.println("Precio final: $" + producto1.calcularPrecio());
    producto1.disponibilidad();

    System.out.println("\nDetalles del Producto 2 (Ropa):");
    producto2.mostrarDetalles();
    System.out.println("Precio final: $" +
producto2.calcularPrecio());
    producto2.disponibilidad();

    System.out.println("\nDetalles del Producto 3
(Alimenticio):");
    producto3.mostrarDetalles();
    System.out.println("Precio final: $" +
producto3.calcularPrecio());
    producto3.disponibilidad();
}
```

Ejemplo:

```
Detalles del Producto 1 (Electrónico):
Producto: Televisor
Precio: $300.0
Cantidad en Stock: 10
Garantía: 3 años
Precio final: $270.0
El producto está disponible.

Detalles del Producto 2 (Ropa):
Producto: Camiseta
Precio: $20.0
Cantidad en Stock: 15
Tamaño: L
Precio final: $22.0
El producto está disponible.

Detalles del Producto 3 (Alimenticio):
Producto: Leche
Precio: $2.5
Cantidad en Stock: 50
Fecha de Vencimiento: próximo mes
Precio final: $2.0
El producto está disponible.
```

3. **(4 puntos)** Desarrolla un sistema en Java para gestionar diferentes tipos de material bibliográfico (libros, revistas y tesis) en una biblioteca. Cada tipo de material debe tener características y comportamientos específicos. El sistema debe permitir realizar operaciones sobre el material bibliográfico, como mostrar detalles, prestar y devolver los elementos, y manejar condiciones especiales como la disponibilidad y los requisitos de préstamo.

MaterialBibliografico:

- **Atributos**:
  - titulo (String): Título del material.
  - autor (String): Autor del material.
  - añoPublicacion (int): Año de publicación.
  - disponible (boolean): Indica si el material está disponible para préstamo.
- **Métodos**:
  - prestar(): Método que intenta prestar el material. Si no está disponible, debe mostrar un mensaje: *"Material no disponible para préstamo."*.
  - devolver(): Método para registrar la devolución del material.
  - mostrarDetalles(): Método para imprimir los detalles básicos del material.
  - disponibilidad(): Imprime si el material está disponible para préstamo o no, según el valor del atributo disponible.

Libro:

- **Atributos adicionales**:
  - genero (String): Género del libro (e.g., "Ficción", "Educativo").
  - numPaginas (int): Número de páginas del libro.
- **Métodos**:
  - prestar(): Sobrescribe el método para permitir un préstamo máximo de 14 días. Si el préstamo no es posible, debe mostrar: *"El libro no puede ser prestado por más de 14 días."*.
  - mostrarDetalles(): Sobrescribe el método para incluir género y número de páginas.

Revista:

- **Atributos adicionales**:
  - numeroEdicion (int): Número de edición.
  - mesPublicacion (String): Mes de publicación.
- **Métodos**:
  - prestar(): Sobrescribe el método para permitir un préstamo máximo de 7 días. Si la revista es de consulta interna (disponible = false), debe mostrar: *"Esta*

revista solo es para consulta interna.".

- mostrarDetalles(): Sobrescribe el método para incluir número de edición y mes de publicación.

Tesis:

- **Atributos adicionales**:
  - universidad (String): Universidad que respalda la tesis.
  - campoEstudio (String): Campo de estudio de la tesis.
- **Métodos**:
  - prestar(): Sobrescribe el método para permitir préstamos solo si se pasa un permiso explícito como parámetro (ej. boolean tienePermiso). Si no tiene permiso, debe mostrar: *"El préstamo de esta tesis requiere permisos especiales."*.
  - prestar(int dias): Permite especificar la duración del préstamo en días.
  - prestar(String usuario): Registra el préstamo indicando el nombre del usuario que lo solicita.
  - mostrarDetalles(): Sobrescribe el método para incluir universidad y campo de estudio.

Probar con este main:

```
public static void main(String[] args) {
    Libro libro = new Libro("El Quijote", "Miguel de Cervantes", 1605,
                                                    true, "Ficción", 1000);
    Revista revista = new Revista("National Geographic", "Varios", 2024,
                                                    false, 10, "Noviembre");
    Tesis tesis = new Tesis("Inteligencia
Artificial", "Juan Pérez",
          2020, true, "MIT", "Computación");
    // Operaciones con el libro
    libro.mostrarDetalles();
    libro.prestar(14);
    libro.devolver();
    // Operaciones con la revista
    revista.mostrarDetalles();
    revista.prestar();
    // Operaciones con la tesis
    tesis.mostrarDetalles();
    tesis.prestar(false);
    tesis.prestar(true);
}
```

```
Título: El Quijote
Autor: Miguel de Cervantes
Año de Publicación: 1605
Disponible: Sí
Género: Ficción
Número de Páginas: 1000
Material prestado por 14 días.
Material devuelto exitosamente.
Título: National Geographic
Autor: Varios
Año de Publicación: 2024
Disponible: No
Número de Edición: 10
Mes de Publicación: Noviembre
Esta revista solo es para consulta interna.
Título: Inteligencia Artificial
Autor: Juan Pérez
Año de Publicación: 2020
Disponible: Sí
Universidad: MIT
Campo de Estudio: Computación
El préstamo de esta tesis requiere permisos especiales.
Tesis prestada exitosamente.
Material prestado exitosamente.
```