27 de junio de 2014

PROGRAMACIÓN II **(Duración del examen: 2,5 horas)**

NOMBRE:

Cuestión 1. (0.5 puntos)

¿Es correcto el código Java indicado a continuación? En caso afirmativo, indique el resultado de la ejecución. En caso contrario, explique el error y describa una posible solución (en sintaxis Java).

```
package p1;
public class C1{
        public int entero = 0;
        String cadena = "Hola";
}

package p1;
public class C2{
        private double d;
        public Clase2(){

        }
public static void main (String[] args){
        C1 o = new C1();
        System.out.println("El atributo entero vale "+o.entero);
        System.out.println("El atributo String es "+o.cadena);
}
}
```

Respuesta:

Cuestión 2. (0.5 puntos)

¿Es correcto el código Java indicado a continuación? En caso afirmativo, indique el resultado de la ejecución. En caso contrario, explique el error y describa una posible solución (en sintaxis Java).

```
public class Prueba{
	private int entero;
	private String cadena;

	public Prueba (int i,String cadena){
		entero=i;
		this.cadena = cadena;
	}
	public int getEntero(){
		return entero;
	}

	public static void main (String[] args){
		System.out.println("El entero es "+Prueba.getEntero());
	}
}
```

Respuesta:

Cuestión 3. (0.5 puntos)

¿Es correcto el código Java indicado a continuación? ¿Por qué?

```
public class Padre{
	int entero = 5;

	public int metodo(){
		return entero;
	}
}
public class Hijo extends Padre{
	protected int metodo (int factor){
		return factor*entero;
	}
}
public class Nieto extends Hijo{
	int metodo (int num){
		return num;
	}
}
```

Respuesta:

Cuestión 4. (0.5 puntos)

¿Cuál es el resultado de ejecutar el siguiente código Java?

```
public class Test {
	public static void main (String[] args){
		try{
			metodo();
			System.out.println("Mensaje 1");
		}
		catch(java.lang.ArithmeticException ex){
			System.out.println("Mensaje 2");
		}
		catch(java.lang.Exception ex){
			System.out.println("Mensaje 3");
		}
		finally{
			System.out.println("Mensaje 4");
		}
		System.out.println("Mensaje 5");
	}
	public static void metodo (){
		try{
			Object o = new Object();
			Integer i = (Integer) o;
			System.out.println("Mensaje 6");

		}
		catch(java.lang.NullPointerException ex){
			System.out.println("Mensaje 7");
		}
		catch(java.lang.ClassCastException ex){
			System.out.println("Mensaje 8");
		}
		System.out.println("Mensaje 9");
	}
}
```

Respuesta:

Cuestión 5. **(0.5 puntos)**

Indique si las siguientes afirmaciones sobre características de Java son verdaderas o falsas. **NOTA: Fallar alguna de las afirmaciones supone invalidar la cuestión entera.**

1. Una clase concreta puede incluir métodos abstractos y concretos _____
2. Una interfaz puede incluir métodos concretos _____
3. Las clases concretas que implementan una interfaz deben implementar necesariamente todos los métodos de dicha interfaz _____
4. Las clases abstractas que implementan una interfaz deben implementar necesariamente todos los métodos de dicha interfaz _____
5. Los métodos constructores de las interfaces deben ser siempre públicos (*public*) _____

Problema 1. (1.75 puntos)

El objetivo de este problema es desarrollar un programa de gestión que permita consultar los ingresos anuales del personal que trabaja en una Universidad. Dicho personal incluye las siguiente figuras:

- El Rector, con unos ingresos anuales de 60.000 euros y exento de tareas docentes debido a sus labores de gestión.
- Una plantilla con dos tipos de profesores: Asociados y Titulares, con unos ingresos anuales de 18.000 y 40.000 euros, respectivamente.
- Un conjunto de Becarios que no perciben ningún salario a cambio de su trabajo.

Además de registrar el nombre y los apellidos de cada trabajador, el programa de gestión deberá guardar (i) la carga docente (número de horas de clase semanales) de cada tipo de profesor y (ii) el nombre de la facultad donde trabaja cada becario.

Se pide:

1. **(0.5 puntos)** Definir la interfaz Asalariados que permita consular el salario anual del personal de la Universidad en nómina (i.e. rector y profesores).

2. **(0.25 puntos)** A partir de una clase `Personal` que engloba a todos los empleados de la Universidad (i.e. `rector`, `profesores` y `becarios`), definir la jerarquía de clases que utilizará el programa de gestión de esta institución. En la solución deben incluirse en cada clase los métodos y atributos que se consideren oportunos.

Respuesta:

3. **(0.75 puntos)** Implementar el método Java ImprimirIngresosProfesores (Personal[] trabajadores) que muestra por pantalla el sueldo anual de los profesores de la Universidad, asumiendo que el array trabajadores incluye toda la plantilla de la institución. **NOTA: Dicho método debe poder ser íntegramente reutilizado cuando el rector contrate a dos nuevos tipos de profesores: Asistentes y Ayudantes.**

Respuesta:

```
public class Gestor {

	public void ImprimirIngresosProfesores (Personal[] trabajadores){









	}
}
```

4. **(0.25 puntos)** Implementar las clases Java correspondientes a las dos nuevas figuras docentes, asumiendo la jerarquía de clases definida en el apartado 2 y suponiendo que profesores asistentes y ayudantes perciben 20.000 y 25.000 euros anuales, respectivamente.

Respuesta:

Problema 2. (0.75 puntos)

Considerando la jerarquía de clases mostrada a continuación, se pide:

1. **(0.25 puntos)** Completar el código del constructor con argumentos de las clases Deportista y PilotoF1.
2. **(0.25 puntos)** Implementar el constructor de copia de ambas clases.
3. **(0.25 puntos)** ¿Es correcto el constructor sin argumentos de la clase PilotoF1?¿Por qué?

Respuesta del apartado 3:

```
public class Deportista {
  private double peso;
  private int edad;

  public Deportista (int edad, double peso){




  }
  // Añadir aquí el constructor de copia de Deportista (signatura + código)




}

public class PilotoF1 extends Deportista{
  private String escuderia;

  public PilotoF1(int edad, double peso, String escuderia){




  }

  public PilotoF1(){
  }

  //Añadir aquí el constructor de copia de PilotoF1 (signature + código)







}
```