**9 de diciembre de 2014**

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

**NOMBRE:**

**Se exige una nota mínima de 5/3=1.7 puntos (sobre 5) para aprobar la asignatura.**

**Cuestión 1. (0.5 puntos)**

Añadir las sentencias oportunas al código Java mostrado a continuación para que el programa imprima siempre por pantalla el mensaje "Fin de la ejecución", independientemente del valor de los numeradores y denominadores utilizados en la división. **La solución planteada debe respetar las siguientes restricciones:**

**(i) No se pueden añadir código en los bloques try y catch existentes.**
**(ii) No se pueden añadir nuevos bloques catch.**

```
public class Test{
        public static void main (String[] args){
                int num = Integer.parseInt(args[0]);
                int denom = Integer.parseInt(args[1]);

                int result;

                try{
                        result = num/denom;
                }
                catch(java.lang.ClassCastException e){
                        System.out.println("Excepción!");
                }
        }
}
```

**Respuesta:**

**Cuestión 2. (0.5 puntos)**

Indicar el resultado de ejecutar el siguiente código Java, asumiendo que el método **clone()** está implementado en la clase **Test**.

```
public class Test {
	private int i;
	private String s;
	private double d;

	public Test (int i,String s, double d){
		this.i=i;
		this.d=d;
		this.s=s;
	}

	public Test() {
		this(1,"mi cadena",2.3);
	}

	public Test clone (){
		//Asumir que la implementación de esté método está disponible.
	}

	public static void metodo1(Test obj){
		obj.i=200;
	}

	public static void metodo2(Test obj){
		obj = new Test();
		obj.i += 400;
		System.out.println("El valor del entero dentro de metodo2() es "+obj.i);
	}

	public static void main (String[] args){
		Test obj = new Test();
		Test obj2 = obj.clone();
		System.out.println("El atributo entero de obj2 es "+obj2.i);
		metodo1(obj);
		System.out.println("El valor del entero tras metodo1() es "+ obj.i);
		metodo2(obj);
		System.out.println("El valor del entero tras metodo2() es "+ obj.i);
	}

}
```

**Respuesta:**

**Cuestión 3. (0.75 puntos)**

Asumiendo la siguiente jerarquía de clases:

```
public abstract class Animal {
private String nombre;
        public Animal(String n) {
                nombre=n;
        }
}

public class Caballo extends Animal{
        public Caballo(String n) {
                super (n);
        }
}

public abstract class Humano {
        private String name;
        public Deportista(String n) {
                name=n;
        }
}
public class Futbolista extends Humano {
        public Futbolista(String n) {
                super(n);
        }
}
public class Tenista extends Humano{
        public Tenista(String n){
                super(n);
        }
}
```

Indicar **de forma razonada** si el código Java mostrado a continuación es correcto **(0.5 puntos).**

```
import java.util.ArrayList;
public class Test {
        public static void main(String[] args){

        ArrayList lista = new ArrayList();

        lista.add(new Tenista("Rafa"));
        lista.add(new Caballo("Rocinante"));
        lista.add(new Animal("Copito de nieve"));
        lista.add(new Futbolista ("Nolito"));
        Tenista t = (Tenista)lista.get(0);
        }
}
```

**Respuesta:**

Declare la variable **lista** de forma que esa colección sólo pueda incluir objetos de las clases **Futbolista** y **Tenista (0.25 puntos)**

**Respuesta:**

**Cuestión 4**. **(0.75 puntos)**

Asumiendo el código Java mostrado a continuación:

```
public class Clase1{
        public void metodo (Object o){
                System.out.println(o);
        }
        public String toString(){
                return ("Esto es un objeto de Clase1");
        }
}
public class Clase2 extends Clase1 {
        public String metodo (Clase2 o){
                return ("Ejecutando método en Clase2");
        }
        public String toString(){
                return ("Esto es un objeto de Clase2");
        }
}

public class Test {
        public static void main(String[] args){
                Clase1 obj = new Clase2();
                obj.metodo(new Clase2());
        }
}
```

Indique si las siguientes afirmaciones son verdaderas o falsas.

**NOTA: Fallar alguna de las afirmaciones supone invalidar la cuestión entera.**

1. El método toString() está sobrecargado en Clase2. _____
2. **Clase1 obj = new Clase2()** no es una sentencia correcta porque no podemos asignar dos tipos diferentes sin hacer un *cast*. _____
3. El resultado de la ejecución es "**Esto es un objeto de Clase1**" _____
4. El resultado de la ejecución es "**Ejecutando método en Clase2**" _____
5. Podríamos usar **Clase1.metodo(new Clase2())** en lugar de **obj.metodo(new Clase2())** _____

**Problema 1. (1.5 puntos)**

Supongamos la jerarquía de clases Java indicada a continuación, en la que identificamos a los **Profesores** y **Alumnos** de una Universidad. Entre los **Alumnos** se cuentan los llamados **Becarios**, que perciben un salario por parte de la institución por tareas de mantenimiento informático de los laboratorios de las diferentes escuelas del campus.

```
public abstract class Personas {
        private String nombre;
        public Personas(String n) {
                nombre=n;
        }
        public String toString(){
                return "Esto es un objeto Persona";
        }
}


public class Profesores extends Personas implements Pagable{
        private String despacho;
        public Profesores(String n, String d) {
                super(n);
                despacho=d;
        }
        public String getDespacho(){
                return despacho;
        }
        public double getSueldo(){
                return 2000;
        }
        public String toString(){
                return "Esto es un objeto Profesor";
        }
}
```

```
public class Alumnos extends Personas{
private double notaMedia;
        public Alumnos (String n, double nota) {
                super(n);
                notaMedia=nota;
        }
        public double getNotaMedia(){
                return notaMedia;
        }
        public String toString(){
                return "Esto es un objeto Alumno";
        }
}

public class Becarios extends Alumnos implements Pagable{
private String escuela;
        public Becarios(String n,double nota,String esc) {
                super (n,nota);
                escuela = esc;
        }
        public double getSueldo(){
                return 500;
        }
}
```

1. **(0.25 puntos)** Definir la interfaz **Pagable** que implementan las clases **Profesores** y **Becarios**.

2. **(0.75 puntos)** Complete el código del método Java mostrado a continuación que debe mostrar por pantalla el sueldo de aquellas personas a las que la Universidad paga un salario (i.e., **Profesores** y **Becarios**). El argumento de entrada de dicho método es un *array* en el que se incluyen todos los **Alumnos** y **Profesores** de la institución. Dicho método deberá imprimir además la nota media de los **Becarios** y el número de despacho de los **Profesores**.

**Respuesta:**

```
public void imprimeSueldo (ArrayList<Personas> lista){









}
```

3. **(0.5 puntos)** ¿Cuál es el resultado de la ejecución del siguiente código Java?

```
public class Test{
        public static void main (String[] args){
                Personas obj = new Becarios ("Pepe Castro", 4.5, "Teleco");
                System.out.println(obj);
        }
}
```

**Respuesta:**

**Problema 2. (1 punto)**

Identifique los posibles errores del siguiente fragmento de código Java y explíquelos de forma razonada (de ser el caso).

```
public class Padre {
	private int i;
	private String cad;

	public Padre (int ent, String cadena){
		i=ent;
		cad=cadena;
	}
}

public class Hijo extends Padre{
private double d;
	public Hijo(double d, int ent, String cad) {
		this.d=d;
		super(ent,cad);
	}

	public Hijo(){

	}
}

public class Test {

	public static void main (String[] args){
		Hijo obj = new Padre(2,"cadena");
	}
}
```

**Respuesta:**