28 de junio de 2013

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

NOMBRE:

Cuestión 1. **(0.5 puntos)**

Dada la siguiente implementación:

```
public class FigGeo {
       private String color = "Blanco";
        private boolean relleno;

        public String toString() {
          return "Color de la figura " + color + " relleno = " + relleno;
        }

        protected String aCadena() {
          return "Color de la figura " + color + " relleno = " + relleno;
        }
}
```

¿Qué nombre técnico recibe la implementación del método `toString()` de la clase `FigGeo`?

Respuesta:

Cuestión 2. **(0.5 puntos)**

Implemente los métodos de la clase `Circulo` invocando para ello los métodos de igual nombre, si los hubiese, de la clase `FigGeo` (especificados en la cuestión anterior)

```
public class Circulo extends FigGeo{
       private double radio = 1.0;

        private String aCadena() {



       }
        public void setRadio(int radio){



        }
}
```

¿Detecta algún error en el código proporcionado?

Respuesta:

Cuestión 3. **(0.5 puntos)**

¿Cuál es el resultado de la ejecución del siguiente código? (Asuma que el código facilitado e implementado en las cuestiones 1 y 2 es correcto)

```
public class Prueba {
	private double radio = 2.0;

	public void metodoPrueba1 () {
	Circulo miCirculo;
	miCirculo.setRadio(radio);
	miCirculo = new Circulo();
	System.out.println("El Radio es:  " + radio);
	}
}
```

Respuesta:

Cuestión 4. **(0.5 puntos)**

Cuando se realiza el tratamiento de excepciones el bloque de instrucciones incluido en la cláusula `finally`:

- Se ejecuta siempre, se haya producido o no alguna excepción y se haya capturado o no la excepción que se haya podido producir
- Se ejecuta siempre que se haya producido alguna excepción aunque independientemente de que se haya capturado o no
- Se ejecuta sólo si no se ha producido ninguna excepción
- Se ejecuta sólo si se ha producido alguna excepción y si dicha excepción ha sido capturada en un bloque `catch` ()

Cuestión 5. **(0.5 puntos)**

Dado el siguiente código de un método, indique qué tipo de método es y qué hace:

```
	public static boolean XXX(String s) {
	  if (s.length() <= 1)
	    return true;
	  else if (s.charAt(0) != s.charAt(s.length() - 1))
	    return false;
	  else
	    return XXX(s.substring(1, s.length() - 1));
	}
```

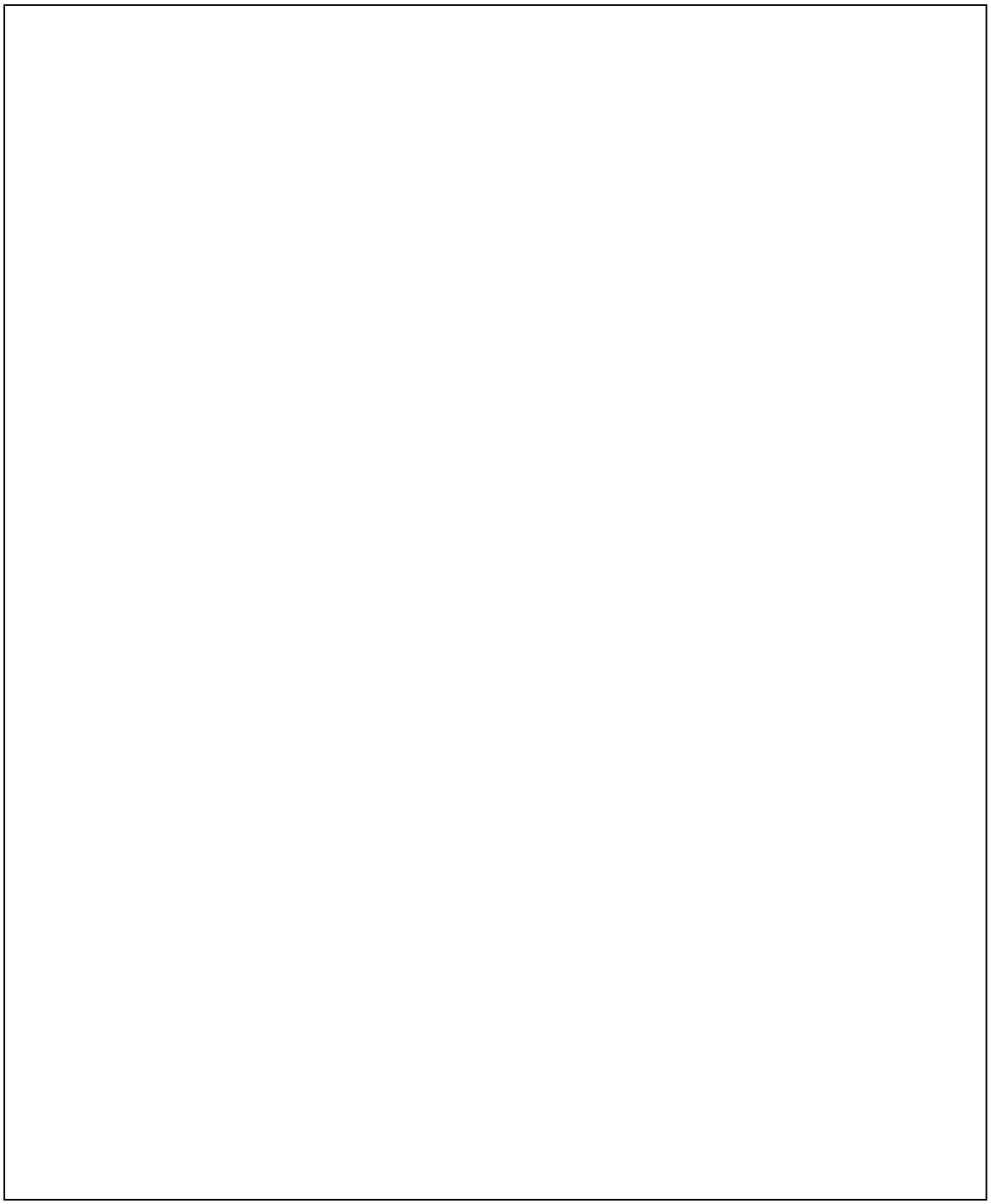
Respuesta:

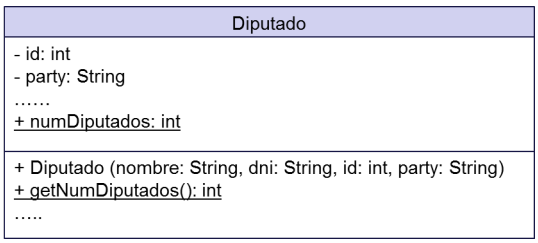
Problema 1. **(5.5 puntos)**

Los diputados, siendo personas y miembros del Congreso, tienen libertad para dar su apoyo mediante su firma a los documentos de queja que son elaborados, a su vez, por otros diputados, para que éstos sean presentados en el registro de entrada ubicado en el Congreso.

1. Elabore el diagrama de clases que represente **todas** las relaciones descritas entre las entidades: `Diputado`, `Persona`, `Queja`, `Congreso` y `Registro`. **(1 punto)**

   Respuesta:





2. Dada la representación UML de la clase `Diputado` (en la figura de la izquierda), implemente las clases `Persona` y `Diputado`. Es imprescindible que el constructor de ésta última invoque al constructor de la primera. **(0.5 puntos)**

Respuesta:

3. **(2 puntos)** Implemente la clase `Queja` teniendo en cuenta que:

   a. Debe tener los siguientes atributos: título de la queja, contenido de la queja, autor de la queja, el número de registro asignado, en el caso en que la queja ya haya sido registrada en el Congreso, número de firmantes y, finalmente, el conjunto de diputados que la firman. El conjunto de firmantes no está acotado previamente en número y podrá variar durante el tiempo de ejecución. Seleccione una estructura que no requiera del almacenamiento de datos en una zona de memoria consecutiva.

   b. Debe ofrecer dos métodos: uno denominado `AltaFirma` y otro denominado `BajaFirma` que, respectivamente, ofrecerán la funcionalidad de agregar y retirar la firma de un diputado a la queja. Ambos métodos deberán garantizar: (i) que la lista de firma no se falsee con adhesiones replicadas y (ii) que, una vez registrada, no sea factible agregar ni retirar ninguna adhesión.

Respuesta:

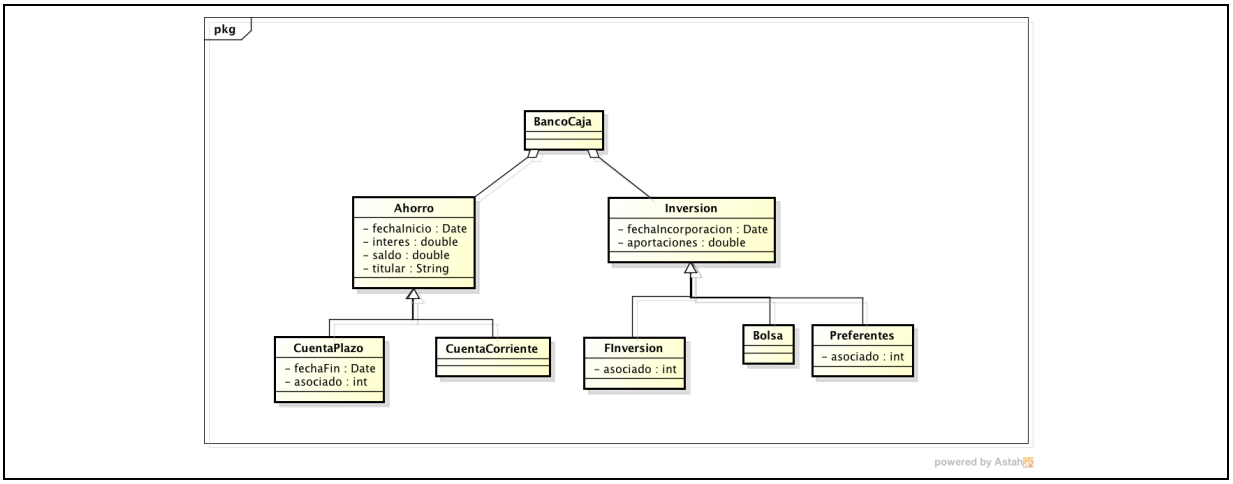
4. **(2 puntos)** Implemente las clases `Congreso` y `Registro` (manteniendo la relación entre ambas que haya definido en el apartado 1) teniendo en cuenta que adicionalmente:
   La clase `Congreso` deberá tener:
   a. Un atributo que almacene el conjunto de diputados del congreso. Dado que la identidad y número de éstos es constante en tiempo de ejecución, la estructura utilizada deberá almacenar la información en una zona de memoria consecutiva
   b. Un método denominado `localizarDiputado` que reciba como parámetro un diputado y compruebe si, efectivamente, está registrado como tal en el congreso.

   La clase `Registro` deberá tener:
   a. Un atributo que almacene el conjunto de quejas procesadas. Seleccione una estructura que no requiera del almacenamiento de datos en una zona de memoria consecutiva.
   b. Un método denominado `registrarQueja` que (i) comprueba, antes de registrar ninguna queja, que todos los firmantes son efectivamente diputados del congreso y (ii) asigna a la queja tramitada un número de registro (a la primera queja recibida el número de registro 324, numerando las siguientes de forma consecutiva).

   Respuesta:

Problema 2. (2 puntos)

Una aplicación informática para la gestión de cuentas de una institución financiera maneja dos jerarquías de clases distintas asociadas respectivamente a productos de ahorro y a productos de inversión. En la siguiente figura se muestra la estructura básica de clases de la aplicación.



En estos momentos de crisis los directivos del banco quieren lanzar un nuevo tipo de productos en los que se puedan combinar dos de los siguientes: `CuentaPlazo`, `FInversion` y `Preferentes`. Para implementar esta solución, el desarrollador de la aplicación considera que puede incluir un método "asociar" en cada una de estas clases que le permita establecer una relación entre uno de esos productos con otro pasado como parámetro y guardar ese otro producto como un atributo.

1. **(0.5 puntos)** Se pide en primer lugar plantear esta solución ampliando el diagrama UML con el mínimo número de cambios posible en las clases y métodos (Esta solución se puede hacer sobre el diagrama anterior)

Respuesta:

2. **(0.5 puntos)** Realizar la codificación en Java del diagrama de clases desarrollado en el punto anterior.

   Respuesta:

3. **(1 punto)** Sobre esta propuesta de productos asociados los directivos introducen como restricción que un mismo producto sólo pueda asociarse con otro. Por ejemplo, una `Cuenta a Plazo` sólo puede asociarse con un único `Fondo de Inversión` o `preferentes`. Para controlar esta restricción el desarrollador de la aplicación se plantea hacer un método estático que dado un producto le indique si ya está asociado con otro o no, devolviendo respectivamente un entero 1 ó 0 según el caso. Sabiendo que las clases `CuentaPlazo`, `FInversion` y `Preferentes` tienen implementado un método `getAsociado` que devuelve como resultado `NULL` si el producto no tiene otro producto asociado y en otro caso la referencia a la instancia del producto asociado, desarrollar el método estático indicado en el programa principal.
   Nota: todos los productos de ahorro se guardan en una variable `ArrayList<Ahorro>` y todos los productos de inversión en un `ArrayList<Inversion>`

   Respuesta: