Prueba 2 de Evaluación Continua

Análisis y Diseño de Software (2010/2011)

Contesta en esta misma hoja

Apellidos: **Nombre:**

Apartado 1 (1.5 puntos)

Señala los errores de compilación (si los hubiera), indica por qué son errores, y cómo los corregirías:

```
class CuentaPrincipal {
	private double saldo;
	public CuentaPrincipal (int saldo) {
		this.saldo = saldo;
	}

	public abstract void liquidar();
}

class Subcuenta extends CuentaPrincipal {
	String saldo;

	public Subcuenta() {
		saldo = "Cancelado";
	}

	public void liquidar() {
		System.out.println("Saldos = " + this.saldo + " : " + super.saldo);
	}
}

public class Problema1 {
	public static void main(String[] args) {
		CuentaPrincipal cp = new Subcuenta();
		cp.liquidar();
	}
}
```

Errores y Corrección:

Prueba 2 de Evaluación Continua

Análisis y Diseño de Software (2010/2011)

Contesta en esta misma hoja

Apellidos:
Nombre:

Apartado 2 (2.5 puntos)

Señala los errores de compilación del método *main* (si los hubiera). Después de eliminar las líneas que provocan los errores, indica los errores de ejecución. Para cada error – de compilación o ejecución –escribe una línea que justifique por qué se produce. Después de eliminar las líneas que los provocan, indica la salida del programa:

```
class ClaseA {
	double z = 6.9;
	protected String titulo = "ClaseA";
	public double rastrear(int alpha) {
		System.out.println("rastreando en Clase A");
		return z * alpha;
	}
}
class ClaseB extends ClaseA {
	private static long id = 0;
	public ClaseB(double d) {
		ClaseB.id++;
		this.z = d;
	}
	@Override
	 public double rastrear(int beta) {
		System.out.println("rastreando en Clase B");
		return super.z + beta;
	}
	public void test() { System.out.println("Clase B valores:" + z + " : " + id);}
}
public class Problema2 {
	public static void main(String[] args) {
	  ClaseA aa = new ClaseA();              // linea 1
	  ClaseA ab = new ClaseB(7.5);           // linea 2
	  ClaseB ba1 = new ClaseA();             // linea 3
	  ClaseB ba2 = (ClaseB) new ClaseA();    // linea 4
	  ClaseB bb1 = new ClaseB();             // linea 5
	  ab.rastrear(8);                        // linea 6
	  ClaseB bb2 = new ClaseB(9);            // linea 7
	  bb2.rastrear(5);                       // linea 8
	  ab.z = 3;                              // linea 9
	  aa.titulo = "Variable aa";             // linea 10
	  ab.test();                             // linea 11
	  bb2.test();                            // linea 12
	}
}
```

```
Errores de compilación en líneas:




Errores de ejecución en líneas:



Salida:
```

Prueba 2 de Evaluación Continua

Análisis y Diseño de Software (2010/2011)

CONTESTA AL DORSO de esta misma hoja. Usa hojas adicionales en caso necesario

Apellidos:
Nombre:

Apartado 3 (6 puntos)

Vamos a gestionar pólizas de seguro sobre obras de arte que pueden ser pinturas o esculturas. Todas las obras de arte tienen título y precio de catálogo, y las esculturas tienen además un atributo que indica si están hechas principalmente de metal, madera o piedra. Cada póliza de seguro tiene un identificador y un límite máximo del valor total asegurado. Dicho valor total se obtiene sumando los valores asegurados de las obras que se vayan incluyendo en la póliza. Al añadir una obra de arte a una póliza de seguros, se considera asegurada por su precio de catálogo, a menos que se indique un valor asegurado concreto, en cuyo caso no se asegurará la obra si dicho valor es superior al precio de catálogo. Tampoco se asegurará una obra si haciéndolo se llega a superar el límite del valor asegurado de la póliza, ni se asegurará dos veces la misma obra en una misma póliza. El coste de asegurar cada obra de arte en una póliza se calcula como un 0,005% de su valor asegurado, pero para las esculturas de madera hay que añadir además un 0,001% de su precio de catálogo. El coste total de una póliza es la suma de los costes de asegurar todas las obras incluidas en dicha póliza.

Escribe las clases necesarias para que la salida del siguiente programa sea la que se muestra más abajo.

```
public class SegurosDeArte {
  public static void main(String[] argc) {
                                          // Nombre y limite máximo de valor asegurado
      PolizaSeguro poliza1 = new PolizaSeguro("Poliza1", 600000);
      PolizaSeguro poliza2 = new PolizaSeguro("Poliza2", 700000);
      PolizaSeguro poliza3 = new PolizaSeguro("Poliza3", 800000);
                                 // Titulo, precio de catalogo
      Pintura   p1 = new Pintura("Barcazas", 100000);
      Pintura   p2 = new Pintura("PzaMayor", 200000);
                                       // Titulo, precio de catalogo, material
      Escultura e1 = new Escultura("Mi Libro", 300000, ObraDeArte.METAL);
      Escultura e2 = new Escultura("El Banco", 400000, ObraDeArte.MADERA);
      Escultura e3 = new Escultura("Mesa dos", 500000, ObraDeArte.PIEDRA);

      poliza1.añadirObraDeArte(p1, 100001); // Valor a asegurar > precio de catalogo. No se asegura
      poliza1.añadirObraDeArte(p2);         // OK Se asegura por valor de precio de catalogo
      poliza1.añadirObraDeArte(e3, 450000); // Valor total de seguro > limite. No se asegura

      poliza2.añadirObraDeArte(e1, 300000); // OK Se asegura por valor indicado
      poliza2.añadirObraDeArte(e1);         // Ya está asegurada. No se vuelve a incluir en seguro
      poliza2.añadirObraDeArte(p2, 100000); // OK Se asegura por valor indicadoa

      poliza3.añadirObraDeArte(p2, 150000); // OK Se asegura por valor indicado
      poliza3.añadirObraDeArte(e2, 300000); // OK Se asegura por valor indicado

      System.out.println("Obras aseguradas en "+poliza1+" = "+poliza1.getObrasAseguradas()
                         +" coste seguro: "+poliza1.costeTotal());
      System.out.println("Obras aseguradas en "+poliza2+" = "+poliza2.getObrasAseguradas()
                         +" coste seguro: "+poliza2.costeTotal());
      System.out.println("Obras aseguradas en "+poliza3+" = "+poliza3.getObrasAseguradas()
                         +" coste seguro: "+poliza3.costeTotal());
      poliza3.imprimeDetalles();
   }
 }
```

Salida:

```
Obras aseguradas en Poliza1 = [PzaMayor] coste seguro: 10
Obras aseguradas en Poliza2 = [Mi Libro, PzaMayor] coste seguro: 20
Obras aseguradas en Poliza3 = [PzaMayor, El Banco] coste seguro: 26
Poliza3 en detalle:
  PzaMayor asegurada por: 150000 con coste 7.5
  El Banco asegurada por: 300000 con coste 19.0
```

Nota: Aunque en el programa no se ha usado, `añadirObraDeArte` debe devolver `true` o `false` dependiendo de si la obra se ha podido incluir en la póliza de seguro o no, según las condiciones expuesta arriba.