Departamento de Ingeniería Telemática
Grado en Ingeniería de Sistemas de Comunicaciones
Programación de Sistemas
Leganés, 9 de marzo de 2018

Examen Parcial I (teoría)
Modelo A

Duración de la prueba: 10 minutos
Puntuación: 3 puntos sobre 10 del examen

Nombre:	
Grupo (61/62):	

NIA:

Firma

En cada pregunta, sólo una opción es correcta. Cada respuesta correcta suma 0.3 puntos. Cada respuesta incorrecta resta 0.1 puntos. Las preguntas sin contestar no suman ni restan puntos.

- Marca la respuesta a cada pregunta con una equis (“X”) en la tabla de abajo.
- Si marcas más de una opción o ninguna opción, la pregunta se considera no contestada.

	A	B	C	D
1				
2				
3				
4				
5				

	A	B	C	D
6				
7				
8				
9				
10				

Las preguntas (1) y (2) se basan en el esqueleto de código que se presenta a continuación:

```
01 public class Point {
02      private double x, y;
03      public Point(double x, double y) {  /* POR HACER */ }
04      public Point() {  /* POR HACER */ }
05      public double getX() {  /* POR HACER */ }
06      public void setX(double x){  /* POR HACER */ }
07      public double getY() {  /* POR HACER */ }
08      public void setY(double y){  /* POR HACER */ }
09      public double distance() {/* POR HACER */ }
10      public double distance(Point otroPunto) {/* POR HACER */ }
11  }
```

1. ¿Cuál de las afirmaciones es correcta?
   (a) Línea 02: atributos; línea 03 constructor; línea 04 es un error; líneas 05-08 métodos de acceso; línea 09 método y línea 10 es un error.
   (b) Línea 02: atributos; líneas 03 y 04 constructores; líneas 05-08 métodos de acceso; líneas 09 y 10 métodos. (***)
   (c) Líneas 02-10 definen los atributos de la clase Point.
   (d) Líneas 02, 05, 07, 09 y 10: atributos; líneas 03, 04, 06 y 08 métodos de la clase.

2. Supongamos que en la clase `Plano` hemos creado el punto "p" así: "Point p = new Point();" y queremos imprimir el valor de sus coordenadas "x" y "y", ¿cuál de las siguientes opciones es correcta?

   (a) System.out.println(p);
   (b) System.out.println(p.x + " " + p.y);
   (c) System.out.println(p.getX() + " " + p.getY()); (***)
   (d) System.out.println(p.setX(4.0) + " " + p.setY(2.0));

3. ¿Cuál de las siguientes anotaciones hace que el método de JUnit prefijado con dicha anotación, se ejecute antes de cada método JUnit prefijado con "@Test"?

   (a) @TestBefore
   (b) @Before (***)
   (c) @BeforeTest
   (d) @BeforeClass

4. Se tienen definidas las siguientes clases:

```
1.1 package p;
1.2 public class Padre {
1.3     public int publica;
1.4     private int privada;
1.5     protected int protegida;
1.6     public Padre(int a, int b, int c){
1.7         publica = a;
1.8         privada = b;
1.9         protegida = c;
1.10    }
1.11}

2.1 package p;
2.2 public class Hijo extends Padre{
2.3     public Hijo(int a, int b, int c){
2.4         super(a,b,c);
2.5     }
2.6     public void usaVarPadre(){
2.7         System.out.println("Hijo:: pública: " + publica);
2.8         System.out.println("Hijo:: privada: " + privada);
2.9         System.out.println("Hijo:: protegida: " + protegida);
2.10    }
2.11 }
```

¿Qué ocurre en este programa?:

(a) El compilador no da ningún error.
(b) La línea 2.8 origina un error de compilación (***)
(c) Las líneas 2.7 y 2.9 originan errores de compilación.
(d) Las líneas 2.7, 2.8 y 2.9 originan errores de compilación.

5. Indica qué valor tendrían los atributos s y t para los objetos a1 y a2 donde dice **"//aquí**":

```
public class A{
    public static int s = 0;
    public int t = 0;
    public A() {t=s; s++;}
}
public class Main {
    public static void main(String[] args) {
        A a1 = new A();
        A a2 = new A();
        // aquí
    }
}
```

(a) a1.s=2, a1.t = 0, a2.s = 2, a2.t = 1 (***)
(b) a1.s=1, a1.t = 2, a2.s = 2, a2.t = 2
(c) a1.s=2, a1.t = 2, a2.s = 2, a2.t = 1
(d) a1.s=1, a1.t = 0, a2.s = 2, a2.t = 2

6. ¿Qué imprime el siguiente programa?

```
public class Padre {
	public Padre(){
		System.out.print("constructorPadre ");
	}
	public void imprime(){
		System.out.print("imprimePadre ");
	}
}
public class Hijo extends Padre{
	public Hijo(){
		super();
		System.out.print("constructorHijo ");
	}
	public void imprime(){
		super.imprime();
		System.out.print("imprimeHijo ");
	}
}
public class MainPadreHijo {
	public static void main(String[] args) {
		Hijo hijo = new Hijo();
		hijo.imprime();
	}
}
```

(a) constructorHijo constructorPadre imprimeHijo imprimePadre
(b) imprimePadre imprimeHijo
(c) constructorHijo imprimeHijo
(d) constructorPadre constructorHijo imprimePadre imprimeHijo (***)

7. Dado el siguiente código y suponiendo que tenemos una clase "B" que hereda de la clase "A". Indica cuál de las siguientes afirmaciones es correcta si queremos asignar un valor al atributo "a" desde la clase "B".

```
public class A{
	private int a;
	public A(int a){
		this.a = a;
	}
}
```

(a) El único modo de hacerlo es haciendo una llamada al constructor de A desde un método de B. (***)
(b) No es posible hacerlo porque el atributo a es privado.
(c) Para hacerlo sería necesario crear un método public void setA(int a){this.a=a;} en la clase B .
(d) Ninguna de las otras opciones es correcta

8. Dado el siguiente programa -donde todas las clases están dentro del mismo paquete-.

```
1 public abstract class A {
2       private int i;
3       // Declarar entero j
4       public A(int i){ this.i = i;}
5       public abstract void metodo(int i);
6 }
7 public class B extends A{
8       public B(int entero){ super(entero); }
9       @Override public void metodo(int i){j = i;}
10 }
11 public class Main {
12      public static void main(String[] args) {
13              B b = new B(10);
14              b.metodo(20);
15       }
16 }
```

Indique qué afirmación es correcta:

(a) La línea 3 solo puede ser: “private int j;”
(b) La línea 3 solo puede ser: “protected int j;”
(c) La línea 3 debe ser: “public int j;” o “protected int j;” (***)
(d) La línea 3 debe ser: “private int j;” o “protected int j;”

9. Las siguientes clases están definidas dentro del mismo paquete:

```
1 public class Persona {
2       protected String nombre;
3       Persona(String s){ nombre = s;}
4 }
5 public class Alumno extends Persona {
6       private String id;
7       Alumno(String s, String id) {
8               super(s);  this.id = id;
9       }
10 }
11 public class Main {
12      public static void main(String[] args) {
13          Persona p1;
14          Alumno a1 = new Alumno("Hugo", "123");
15          p1 = a1;
16          Alumno a2;
17          a2 = p1;
18      }
19 }
```

Indica qué afirmación es correcta:

(a) En las líneas 15 y 17 se realizan conversiones implícitas de tipo;
(b) Las líneas 15 y 17 deben reescribirse como “p1 = (Persona) a1;” y “a2 = (Alumno) p1;” respectivamente.
(c) En la línea 15 se realiza una conversión implícita de tipos. La línea 17 debe reescribirse así: “a2 = (Alumno) p1;”. (***)
(d) La línea 15 debe reescribirse así: “p1 = (Persona) a1;”. En la línea 17 se realiza una conversión implícita de tipos.

10. La clase Fecha está implementada como sigue:

```
public class Fecha {
        private int dia, mes, anio;
        Fecha(int d, int m, int a){
                dia = d; mes = m; anio = a;
                if (!verificaFecha()){ System.err.println("ERROR"); }
        }
        protected static boolean enRango(int valor, int min, int max){
                return (valor >= min && valor <= max);
        }
        protected static boolean esBisiesto(int anio){
                return (anio%4==0 && anio%100 != 0 || anio%400 == 0);
        }
        private boolean verificaFecha(){
                if (mes == 4 || mes == 6 || mes == 9 || mes == 11)
                        return enRango(dia,1,30);
                else if (mes != 2)
                        return enRango(dia,1,31);
                    else if (esBisiesto(anio))
                                return enRango(dia,1,29);
                             else
                                return enRango(dia,1,28);
        }
}
```

Por otra parte, la clase TestEnRango que pretende verificar el método enRango() tiene la siguiente estructura:

```
1 import static org.junit.Assert.*;
2 import org.junit.BeforeClass;
3 import org.junit.Test;
4 public class TestEnRango  {
5       public static Fecha fecha;
6       @BeforeClass
7       public static void inicializacion(){
8               fecha = new Fecha(9, 3, 2018);
9       }
10      @Test public void testAntes() {…}
11      @Test public void testLimiteInferior() {…}
12      @Test
13      public void testEnMedio() {
14              // RELLENAR
15      }
16      @Test public void testLimiteSuperior() {…}
17      @Test public void testDespues() {…}
18 }
```

¿Qué instrucción debemos colocar en la línea 14?

(a) assertTrue (enRango(9, 1, 31));
(b) assertTrue (fecha.enRango(9, 1, 31), true);
(c) assertEquals(enRango(9, 1, 31), true);
(d) assertEquals(Fecha.enRango(9, 1, 31), true); (***)