Universidade de Vigo

ESCOLA SUPERIOR DE ENXEÑERÍA INFORMÁTICA

Programación Avanzada
– Septiembre de 2004 –

Examen de Teoría

NOTA

Apellidos _____

Nombre _____

PLAN: ☐ **Nuevo** ☐ **Viejo** **DNI** _____

IMPORTANTE:

- El tiempo para la realización del examen es de 2 horas.
- Se deben cubrir los datos del encabezado (incluyendo PLAN: [ ]Nuevo, [ ]Viejo) de esta hoja, que será grapada junto con las hojas de respuestas que deberán ir numeradas.
- En el momento de la entrega del examen se deberá firmar una hoja de entrega, que estará a disposición del alumno.
- Las calificaciones de teoría y práctica se harán públicas a última hora del viernes 10 de septiembre en el tablón y en Internet (http://proa.ei.uvigo.es/).

REVISIÓN DE CALIFICACIONES:
PRÁCTICA: **13 de Septiembre (lunes) de [8:30-13:00]**.
TEORÍA: **13 de Septiembre (lunes) de [15:30-19:30]**.

☐ **1.-** ***Herencia en Java versus C++*:**

1.1.- ¿Qué tipo de herencia está permitida en C++ y no en Java?. ¿Qué mecanismo proporciona Java para aportar una funcionalidad parecida?.

1.2.- ¿Qué problemas técnicos o inconvenientes acarrearía implementar en Java el tipo de herencia al que se hace referencia en la cuestión anterior?.

☐ **2.-** ***Gestión de excepciones*:**

2.1.- ¿Qué sentido tiene utilizar cláusulas *catch* múltiples en el código Java?. A este respecto, ¿qué precaución se debe tomar cuando se programan dichas cláusulas?.

2.2.- ¿Es posible utilizar sentencias *try* anidadas en el código Java?, si es posible, pon algún ejemplo de su utilización. ¿Cuántas sentencias *finally* deben aparecer por cada bloque *try*/*catch*?.

☐ **3.-** ***Trabajo con la red*:**

3.1.- ¿Para qué se utiliza en Java la clase *InetAddress*?.

3.2.- ¿Qué es un método de fábrica?.

3.3.- ¿Para qué se utilizan los métodos de fábrica en el caso de la clase *InetAddress*?.

3.4.- ¿Qué clases se crean a partir de un *Socket* Java y permiten enviar y recibir información a través de ese socket?.

4.- ***Generación de archivos de documentación***:

4.1.- ¿Para qué se utilizan las etiquedas *@exception* y *@deprecated*?. ¿A qué entidades del lenguaje se aplican cada una de ellas?.

4.2.- ¿Cuál es la característica más destacable de la etiqueta *@deprecated*?.

5.- ***Seguridad en Java***:

5.1.- Esquema de bloques de la arquitectura nuclear de seguridad de Java.

6.- ***Desarrollo de aplicaciones distribuidas***:

6.1.- ¿Para qué se utilizan los métodos *bind*() y *lookup*() de la clase *Naming* de Java?.

6.2.- ¿Qué ficheros genera la utilidad *rmic* de Java?, ¿para qué se utilizan?.

7.- ***Applets***:

7.1.- ¿Qué métodos forman la estructura básica de un applet Java?. ¿Cuál es la característica común en cuanto a su invocación?.

7.2.- ¿De qué forma se le pueden pasar parámetros a un applet?. ¿Qué método se utiliza en la clase *Applet* para su recepción?.

8.- ***Programación multihilo***:

8.1.- En Java existen dos formas de crear un hilo: heredando de la clase *Thread* o implementando la interfaz *Runnable*, ¿cuándo se utiliza una u otra?.

8.2.- Los hilos hijo en Java siempre deben haber terminado antes de que finalice el hilo padre, ¿de qué dos formas se puede garantizar que esto ocurra?.

9.- ***Gestión de entrada/salida***:

9.1.- ¿Qué capacidad tienen las clases que implementan la interfaz *Serializable*?.

9.2.- Relaciona serialización y RMI (*Remote Method Invocation*).

10.- ***Gestión de bases de datos con JDBC***:

10.1.- ¿Cuáles son los 4 pasos básicos que se deben seguir para lograr ejecutar una consulta SQL sobre un SGBD y obtener sus resultados?.

10.2.- Identifica los nombres de las clases y los métodos más importantes que entran en juego en la cuestión anterior.