Examen de Programación 3

23 de ENERO de 2015 (MODALIDAD 2)

1. Una colaboración describe como un grupo de objetos trabaja conjuntamente para realizar una tarea.
2. En el diseño mediante tarjetas CRC, utilizamos una tarjeta por cada clase.
3. La sobrescritura es una forma de polimorfismo.
4. Un buen diseño orientado a objetos se caracteriza por un alto acoplamiento y una baja cohesión.
5. El principio de segregación de interfaz indica que el código cliente no debe ser forzado a depender de interfaces que no utiliza.
6. La inversión de control en los frameworks es posible gracias al enlace dinámico de métodos.
7. El usuario de un framework implementa el comportamiento declarado en los interfaces del framework mediante herencia de implantación.
8. Para poder utilizar un framework, es necesario crear clases que implementen todas las interfaces declaradas en el framework.
9. El polimorfismo es una forma de reflexión.
10. La instanciación mediante reflexión de un objeto de la clase Rectángulo del paquete pack se realiza así: Class.forName("pack", "Rectangulo");
11. En java el enlace por defecto de métodos de instancia es estático.
12. En Java no se pueden derivar clases genéricas de otras clases genéricas.
13. En Java las sentencias de un bloque 'finally' solamente se ejecutan cuando se ha producido una excepción y no la hemos capturado en un bloque 'catch'.
14. En el paradigma orientado a objetos, un objeto siempre es instancia de alguna clase.
15. El método invocado por un objeto en respuesta a un mensaje viene siempre determinado, entre otras cosas, por la clase del objeto receptor en tiempo de compilación.
16. En el paradigma orientado a objetos, un programa es un conjunto de objetos que se comunican mediante el paso de mensajes.
17. La robustez de un sistema software es un parámetro de calidad intrínseco.
18. El siguiente código en Java define una interfaz: interface S{}
19. Sea un método llamado glue(), sin argumentos, implementado en una superclase y sobrescrito en una de sus subclases. Siempre podremos invocar a la implementación del método en la superclase desde la implantación del método en la subclase usando la instrucción super.glue();
20. Los métodos abstractos siempre tienen enlace dinámico.
21. Dado un objeto, mediante reflexión no se puede obtener la lista de todos los métodos declarados en su clase, tanto públicos como privados.
22. Cuando diseñamos sistemas orientados a objetos las interfaces de las clases que diseñamos deberían estar abiertas a la extensión y cerradas a la a modificación.
23. La refactorización nunca produce cambios en las interfaces de las clases.
24. Las sentencias 'switch' son un caso de código sospechoso (código de mal olor).
25. El código duplicado es un caso de código sospechoso en el que se aconseja el uso de técnicas de refactorización para eliminarlo.
26. La existencia de una sólida colección de pruebas unitarias es una precondición fundamental para la refactorización.
27. "Los métodos que usan referencias a clases bases deben ser capaces de usar objetos de clase derivadas sin saberlo" es una posible formulación del principio de inversión de dependencias.
28. El enlace de la invocación a un método sobrescrito se produce en tiempo de ejecución, en función del tipo en tiempo de ejecución del receptor del mensaje.
29. This es un ejemplo de variable polimórfica en Java.
30. En Java el downcasting siempre se realiza en tiempo de ejecución.
31. En Java, los métodos de instancia con polimorfismo puro, pero no abstractos tienen enlace dinámico por defecto.

32. En Java, un atributo de clase debe declararse dentro de la clase con el modificador static.
33. En Java, gracias a loa sobrecarga de operadores podemos crear operadores en el lenguaje.
34. Si en una clase no se declara, implícita o explícitamente, un constructor por defecto, no se pueden crear instancias de esa clase.
35. Una de las características básicas de un lenguaje orientado a objetos es que todos los objetos de la misma clase pueden recibir los mismos mensajes.
36. La instrucción throw en Java sólo permite lanzar objetos que son instancias de la clase java.lang.Throwable o de clases derivadas de ésta.
37. En Java, la instrucción throw no se puede usar dentro de un bloque catch.
38. Los métodos genéricos no se pueden sobrecargar so sobrescribir.
39. Una clase abstracta siempre tiene como clase base una clase interfaz.
40. De una clase abstracta no se pueden crear instancias, excepto si se declara explícitamente algún constructor.