Ejercicios de repaso N° 1

Preguntas - Verdadero o Falso (Justificar las falsas)

1. (___) Las pruebas de caja negra se realizan sin necesidad de conocer el código fuente del software.
2. (___) Las métricas del software solo se aplican después de que el producto ha sido liberado.
3. (___) La complejidad ciclomática se calcula sumando la cantidad de líneas de código.
4. (___) En la relación de composición, los objetos pueden seguir existiendo aunque el objeto principal desaparezca.
5. (___) El encapsulamiento permite ocultar la implementación interna de una clase, exponiendo solo lo necesario.
6. (___) La prueba unitaria busca evaluar cómo interactúan distintos módulos entre sí.
7. (___) Una interfaz en UML puede ser instanciada directamente como una clase concreta.
8. (___) El análisis orientado a objetos se utiliza para identificar clases y relaciones en base a los requisitos.
9. (___) La prueba del sistema se realiza antes de la prueba unitaria.
10. (___) El uso de patrones de diseño ayuda a evitar la reutilización de soluciones previas.

Completa los espacios en blanco

1. La técnica de prueba que analiza el flujo interno del código se llama _____.
2. En la programación orientada a objetos, una _____ es una plantilla para crear objetos, mientras que un _____ es una instancia de ella.
3. La complejidad ciclomática permite medir la cantidad de _____ independientes en un programa.
4. Una agregación representa una relación _____ entre objetos, mientras que una composición representa una relación _____.
5. La prueba que se realiza para verificar que el producto cumple con lo que el cliente espera se llama prueba de _____.
6. En UML, el símbolo (+) indica un atributo o método de tipo _____.
7. Las métricas del modelo de análisis permiten verificar la _____ y _____ de los requisitos.
8. En una relación de herencia, una clase hija hereda los _____ y _____ de su clase padre.
9. La cobertura de código es una métrica que indica el porcentaje de _____ que fue ejecutado durante las pruebas.
10. Las pruebas funcionales y no funcionales forman parte de la prueba del _____.

Cuestionario

Tema 1: Técnicas de Prueba del Software

1. ¿Por qué es importante realizar pruebas en cada etapa del desarrollo de software?
2. Explica con tus palabras la diferencia entre una prueba de caja blanca y una de caja negra. ¿En qué casos se usaría cada una?
3. ¿Cómo se construye un camino básico en una prueba estructural? ¿Qué ventajas ofrece?
4. ¿Qué aspectos debe tener en cuenta un analista al diseñar casos de prueba efectivos?
5. ¿Qué riesgos podrían presentarse si no se realizan pruebas especializadas en aplicaciones móviles o distribuidas?

Tema 2: Estrategias de Prueba del Software

1. ¿Cuál es la diferencia entre validar un producto y verificar un módulo?
2. ¿Qué papel juega la planificación dentro de una estrategia de pruebas? ¿Qué elementos no pueden faltar?
3. Explica la diferencia entre una prueba unitaria y una prueba de integración, y da un ejemplo práctico de cada una.
4. ¿Qué beneficios tiene aplicar un enfoque iterativo en la ejecución de pruebas?
5. ¿Qué importancia tiene la depuración en el desarrollo de software y qué habilidades debe tener quien la realiza?

Tema 3: Métricas del Software

1. ¿Qué son las métricas del software y por qué son necesarias en un proyecto?
2. ¿Qué información puede brindarte la complejidad ciclomática sobre un módulo de código?
3. ¿Cómo podrías interpretar una cobertura de código del 60% en una aplicación próxima a su entrega?
4. ¿Qué diferencia hay entre una métrica de producto y una de proyecto? Da un ejemplo concreto.
5. ¿Cómo ayudan las métricas a detectar problemas en la etapa de mantenimiento de un software?

Tema 4: Conceptos y Principios Orientados a Objetos

1. ¿Qué ventajas ofrece el encapsulamiento para el desarrollo de software de gran escala?
2. Explica con tus palabras qué significa polimorfismo y cómo se aplica en un lenguaje de programación.
3. ¿Cuál es la diferencia entre una clase y un objeto? Da un ejemplo de cada uno.
4. ¿Cómo influye el uso de herencia en la calidad y mantenimiento de un sistema?

5. ¿Qué beneficios tiene usar programación orientada a objetos en comparación con la programación estructurada?

Tema 5: Análisis Orientado a Objetos

1. ¿Qué papel cumple el análisis orientado a objetos dentro del proceso de desarrollo?
2. ¿Cómo se identifican las clases y relaciones durante el análisis de un sistema?
3. ¿Por qué es importante modelar tanto el comportamiento como las relaciones entre objetos?
4. ¿Qué elementos debe incluir un buen modelo objeto-relación?
5. ¿Cuál es el valor de validar un modelo de análisis con usuarios reales?

Tema 6: Diseño Orientado a Objetos

1. ¿Qué decisiones deben tomarse al transformar un modelo de análisis en un diseño orientado a objetos?
2. Explica el concepto de patrón de diseño. ¿Por qué son útiles? Da un ejemplo.
3. ¿Qué elementos visuales se incluyen en un diagrama de clases UML y qué representan?
4. ¿Cuál es la diferencia entre una agregación y una composición en el diseño OO?
5. ¿Por qué es útil utilizar diagramas y notación UML durante el diseño de un sistema?