2do Parcial - Ingenieria de Software II

1. De las siguientes afirmaciones, seleccione aquellas que considere verdaderas sobre la arquitectura de software: **[8 pts]**

a. Señala la estructura de control de flujo de la ejecución de un sector de código.
b. Inhibe o habilita la implementación de los atributos de calidad.
c. Es un artefacto clave que permite elaborar una agenda de desarrollo.
d. Se define en etapas tardías del desarrollo.
e. Sirve de base para la formación de un nuevo miembro de equipo.
f. Establece las estructuras de datos importantes que serán encapsuladas de manera privada en componentes.
g. Establece un diseño de bajo nivel de abstracción para los programadores.
h. Su documentación mejora la comunicación entre distintos stakeholders.

2. Teniendo en cuenta la relación entre arquitectura y atributos de calidad señale las afirmaciones que son verdaderas: **[8 pts]**

a. Los atributos de calidad son irrelevantes en las decisiones arquitectónicas.
b. La elección de una arquitectura puede mejorar la mantenibilidad y la testeabilidad de un sistema.
c. La portabilidad no está influenciada por la arquitectura del sistema.
d. La selección de un estilo o patrón de arquitecturas conocido depende principalmente de los requerimentos no funcionales.

3. Identifique si cada frase esta relacionada con medidas de punto de función (PF), medidas de líneas de código de software (SLOC), o ambas (PF-SLOC): **[8 pts]**

a. Medida directa, menos costosa de implementar. **SLOC**
b. Más subjetiva, no tiene un significado físico directo. **PF**
c. Más costosa de implementar. **PF**
d. Método simple y con un alto grado de objetividad. **SLOC**
e. La medición puede comenzar antes de la implementación. **PF**
f. Su unidad física de medida puede resultar ambigua. **SLOC**
g. No se puede empezar a medir hasta la implementación. **SLOC**
h. Medida indirecta basada en información del dominio y complejidad del software. **PF**
i. Necesita una definición operacional. **PF-SLOC**

4. En el contexto de métricas, marque cuales de las siguientes afirmaciones sobre "baseline" son verdaderas: **[8 pts]**

a. Establecer baselines de un proyecto es útil sólo para comparaciones con evaluaciones futuras del mismo proyecto.
b. Las baselines son útiles para desarrollar estimaciones significativas, producir sistemas de mayor calidad y entregar el producto a tiempo.
c. El baseline es un concepto exclusivo de gestión de configuración de software para ayudar a controlar el cambio.
d. Las aplicaciones de las "baselines" mejoran cuando las usamos para estimar proyectos similares.
e. Las "baselines" pueden servir para comparar y obtener información sobre el proceso, el proyecto y el producto.
f. No es posible armar baselines reconstruyendo los datos históricos requeridos por las métricas que la componen.

2do Parcial - Ingenieria de Software II

5. Al aplicar las siguientes métricas, qué combinación de tres métricas con valores altos puede servir para detectar clases que potencialmente deban ser divididas. Enumere de la más relevante a la menos relevante: **[10 pts]**

a. Métodos Ponderados por Clases (MPC) (Weighted Methods per Class WMC). **2**
b. Profundidad del Árbol de Herencia (Depth of Inheritance Tree DIT).
c. Número de Hijos (Number of Children NOC).
d. Acoplamiento entre Objetos (Coupling Between Object classes CBO).
e. Respuesta Para una Clase (Response For a class RFC). **3**
f. Carencia de Cohesión en Métodos (Lack of Cohesion in Methods LCOM). **1**

6. De los siguientes resultados de métricas, señale aquellos que sugieren las propiedades necesarias para disponer de un testeo sencillo: **[8 pts]**

a. RFC bajo.
b. LCOM alto.
c. DIT bajo.
d. WMC bajo.
e. CBO alto.

7. De las siguientes afirmaciones sobre el costo de la corrección de un defecto, seleccione las afirmaciones que considera verdaderas: **[8 pts] (selección única)**

a. La corrección de un defecto tiene un mismo costo sin importar en qué etapa se corrige.
b. Cuanto más tarde la corrección menos costosa es porque más se conoce sobre el problema.
c. La corrección de un defecto es más costosa porque hay una amplificación del error en etapas anteriores.
d. La corrección de un defecto es más costosa en la etapa de mantenimiento.
e. La corrección de un defecto es costosa en etapas tempranas del desarrollo.

8. ¿Cuál de las siguientes afirmaciones describe correctamente el concepto de amplificación de error en ingeniería de software? **[8 pts] (selección única)**

a. La amplificación de error ocurre cuando actividades prescriptas reducen progresivamente el impacto de un defecto a medida que se avanza por las etapas del desarrollo.
b. La amplificación de error es el proceso de corregir automáticamente errores detectados durante la fase de pruebas, para evitar que lleguen a la implementación.
c. La amplificación de error se refiere al fenómeno en el que un error introducido en una etapa temprana del desarrollo no se detecta y genera defectos adicionales en etapas posteriores, aumentando su impacto y costo.
d. La amplificación de error ocurre únicamente durante la fase de mantenimiento, cuando los errores son provocados por el uso incorrecto del software por parte de los usuarios.

2do Parcial - Ingenieria de Software II

9. ¿Cuáles de las siguientes son actividades preventivas que ayudan a evitar la amplificación de defectos en el desarrollo de software? **[8 pts]**

- **a. Revisiones de código.**
- b. Pruebas en las etapas avanzadas antes del pasaje a producción.
- **c. Análisis de requisitos para detectar anomalías en lo que hay que desarrollar.**
- d. Refactorización del código después del lanzamiento.
- **e. Incluir pair programming en el desarrollo de features complejas.**
- f. Monitoreo del sistema andando en producción.
- **g. Capacitación del equipo en áreas claves.**

10. De cada actividad del proceso de gestión de riesgo, nombre y describa los productos (o resultados) que se generan: **[10 pts]**

- a. Identificación de riesgos: **Lista de riesgos potenciales**
- b. Análisis de riesgos: **Lista de riesgos priorizada**
- c. Planificación de riesgos: **Planes de anulación y contingencia**
- d. Supervisión de riesgos: **Evaluación de riesgos** (**Risk Assessment**)

11. Generar codebases y codelines en el contexto de SCM es una actividad de: **[8 pts]** (selección única)

- a. Gestión del cambio.
- **b. Gestión de versiones.**
- c. Construcción del sistema.
- d. Gestión de entregas.

12. En el contexto de SCM, lea las siguientes afirmaciones y diga cuales son verdaderas: **[8 pts]**

- **a. Un codeline es una secuencia de versiones relevantes de un item de configuración del software.**
- b. Un codeline es una colección de versiones de items de configuración que hacen a una versión del sistema.
- c. Cualquier commit de "git" representa un codebase para SCM.
- **d. Un codeline en "git" es la progresión de commits relevantes de un item de configuración del software.**
- **e. Un codebase se puede gestionar usando "git" con un label.**
- f. Un codebase no necesita una aprobación formal dentro del proceso de SCM.