LETRA A

1. La búsqueda y recuperación de componentes reutilizables de software plantea problemas:
   a) similares a las búsquedas de artículos sobre un tema para estudiar
   b) de evaluación de la calidad de los componentes
   c) de costos y tiempo de salida al mercado
   d) todas las anteriores

2. Pruebas de integración
   I) Con la estrategia big-bang es difícil ver cuál es el origen de una falla encontrada en la integración
   II) Normalmente los drivers son más complejos de desarrollar que los stubs
   III) En la estrategia top-down puede ser muy complejo (e incluso imposible) generar determinados casos de pruebas para algunos módulos.
   a) Se cumplen I, II y III
   b) Se cumplen I y III. No se cumple II
   c) Se cumple I. No se cumple II y no se cumple III
   d) Se cumple II. No se cumple I y no se cumple III.

3. Para el diseño de la interfaz de usuario es importante tener en cuenta que
   a) los términos utilizados en las pantallas sean los definidos en el desarrollo para los conceptos identificados para el sistema
   b) los menúes y comandos que se utilizan en las pantallas se comporten de forma adecuada aunque el comportamiento sea distinto en cada pantalla
   c) la apariencia definida para las pantallas sea aprobada por los integrantes del equipo de desarrollo no por el cliente
   d) el código de colores sea utilizado para apoyar las tareas en las pantallas, por ejemplo resaltar similitudes o diferencias

4. Una empresa que desarrolla un producto desde hace 5 años, se plantea cambiar de plataforma y por lo tanto desea evaluar distintas alternativas, A y B. La hipótesis es que la alternativa A es mejor en términos de costo beneficio pero igual se decide hacer una prueba con dos grupos trabajando en paralelo en condiciones similares pero uno en cada una de las plataformas evaluadas.
   a) Esta estrategia corresponde a un experimento formal.
   b) Se trata de una evaluación de características
   c) a) y se definieron un conjunto de variables que serán controladas durante todo el desarrollo
   d) b) y también incluye la comparación con la línea base de desarrollo de la empresa

5. Respecto al mantenimiento de software:
   a) Enfrenta la disyuntiva entre la elegancia y principios de diseño y la necesidad de urgencia de la solución
   b) (a), lo que puede ser visto como el conflicto entre las necesidades de largo plazo y las de corto plazo
   c) (b) y suele resultar difícil realizar pruebas que permitan verificar de forma adecuada los cambios, debido al esfuerzo necesario para crear y mantener ambientes de prueba
   d) (c) y su gestión debiera quedar en manos de un Comité de Control de Cambios que integre las visiones de Cliente, Usuarios y Desarrolladores

6. Entre los principios de diseño vistos en el curso
   a) el diseño para el cambio implica tener en cuenta cambios futuros a los requerimientos actuales para que sean fácilmente incluidos en el sistema
   b) la separación de intereses consiste en asignar distintas tareas a distintos módulos de forma de que cada uno se ocupa de aspectos específicos
   c) la alta cohesión de cada módulo y el bajo acoplamiento entre módulos permite obtener una alta modularización del sistema que aporta a la localización e impacto de los cambios
   d) se cumplen todas las anteriores

7. Se pueden utilizar como técnicas y herramientas de diseño
   a) las tarjetas CRC donde la información en las tarjetas se define por el equipo de desarrollo para la verificación del sistema
   b) los diagramas y notación UML que permiten modelar un sistema desde los requerimientos hasta el deployment
   c) los patrones de diseño que plantean para un problema determinado una solución particular sin tener en cuenta el contexto del problema
   d) los frameworks que como los COTS constituyen reutilización con enfoque de caja negra tanto de diseño como de implementación

8. Para mejorar el diseño es posible utilizar la técnica de análisis del árbol de faltas

a) donde los nodos definidos en el árbol representan cada uno una causa posible que podría llevar a la falla que está siendo analizada
b) a) y estos nodos pueden ser eventos básicos o composición de eventos básicos con operadores lógicos según la combinación que produce la falla
c) b) y el conjunto de corte del árbol de faltas definido permite identificar las combinaciones de eventos básicos que al ocurrir producen la falla
d) no se cumple ninguna de las anteriores

9.
a) El proceso de verificación puede comenzar a la vez que el proceso de desarrollo de un producto de software
b) (a) El modelo V, entre otras cosas, muestra cómo se pueden realizar actividades de verificación en paralelo con las actividades de construcción del producto
c) El proceso de verificación no tiene relación con el proceso de desarrollo
d) (c) y es conveniente que las actividades del proceso de verificación sean ejecutadas por los líderes del proyecto de desarrollo.

10. En la empresa "Cero defecto software" se ha llevado a cabo el desarrollo de un producto de software. Durante las pruebas de sistema se han provocado 125 fallas. El producto vuelve a desarrollo y se encuentran y corrigen 93 defectos. Se realizan pruebas de regresión de sistema ejecutando exactamente los mismos casos de prueba anteriormente ejecutados. Se tendrá en cuenta una ejecución de casos de prueba perfecta en el sentido que: realmente se ejecutaron los mismos casos ejecutados anteriormente y se distingue de forma perfecta un resultado correcto de uno incorrecto. Las pruebas de regresión ejecutan todas correctamente. Se pasa a ejecutar nuevas pruebas de sistema y también ejecutan correctamente. El porcentaje total de tiempo usado durante las pruebas de sistema es el 60% del tiempo total de todo el proyecto.
a) El producto que se obtiene es un producto cero defecto
b) A partir de lo descrito no se puede conocer la cantidad de defectos remanentes del producto
c) Nunca puede pasar que se detecten y corrijan menos defectos (93) que las fallas (125) y que luego las pruebas de regresión no encuentren fallas. Se está considerando pruebas de regresión completas y perfectas (con la misma definición brindada más arriba)
d) 125 fallas son pocas para una primera ejecución de los casos de prueba de un producto a nivel de sistema

11. I) Las recorridas y las inspecciones son: técnicas estáticas de verificación
II) Las recorridas y las inspecciones son: técnicas dinámicas de verificación
III) En las reuniones de inspección no se debe criticar a la persona que desarrolló el producto
IV) En las recorridas se usa una lista de defectos comunes (check-list)
V) En las inspecciones se usa una lista de defectos comunes (check-list)
Cuál de las siguientes opciones es verdadera.
a) Se cumplen I, III y IV
b) Se cumplen I y IV pero no se cumple III
c) Se cumplen I, III y V
d) Se cumplen II, III y V

12. Respecto a los criterios de terminación de las pruebas presentados en el curso.
a) Los criterios de terminación basados en lograr cumplir (sin fallas en los casos ejecutados) con un cubrimiento de código pueden ser apropiados para las pruebas unitarias.
b) Es apropiado terminar las pruebas de sistema cuando el tiempo establecido para las mismas ha terminado.
c) Es apropiado basar la terminación de las pruebas únicamente en el resultado de la ejecución de los casos de prueba; si todos los casos de prueba ejecutan correctamente entonces se detienen las pruebas.
d) Es aconsejable detener las pruebas de integración si se han detectado más del doble de los defectos detectados durante las pruebas unitarias.

13. La prueba de función del sistema tal cual se vio en el curso, incluye:
a) Prueba de las funcionalidades del sistema de forma individual
b) (a) y distintas combinaciones de funcionalidades, tales como ciclos de vida de entidades y procesos de la organización
c) (b) y es común utilizar técnicas de caja negra para las pruebas de sistema
d) Ninguna de las anteriores

14. Los modelos de calidad de los productos de software se basan fundamentalmente:
a) la identificación de características, subcaracterísticas e indicadores que permiten medirlas
b) a) y son de alcance universal
c) en el tiempo medio entre fallas y la intensidad de las mismas
d) en el proceso utilizado para su desarrollo

15. Se recomienda realizar el análisis post-mortem de un proyecto:
    a) a los seis meses de su culminación
    b) a) y se considera esencial para evaluar la productividad de cada participante
    c) para evaluar la satisfacción del cliente y aprender de los errores y aciertos
    d) c) y mejorar las estimaciones

16. El modelo de madurez de la capacidad de los procesos CMMI está pensado fundamentalmente para:
    a) Mejorar los procesos de desarrollo de software de las empresas pequeñas y medianas
    b) Incluir los procesos de testing unitario, de integración y de sistemas
    c) b) y Dar una mayor flexibilidad para ordenar los procesos a mejorar e integrar los procesos de ingeniería de sistemas y de software
    d) Ninguno de los anteriores

17. La conversión de datos de un sistema viejo a un sistema nuevo
    a) Conviene considerarla una vez que el sistema nuevo ya está construido y en funcionamiento, para evitar retrabajo proveniente de cambios en las estructuras de datos.
    b) Conviene considerarla al comienzo del proyecto que involucra la sustitución, para tomarla en cuenta en la planificación general del proyecto.
    c) Conviene definirla por completo al comienzo del proyecto para poder determinar los recursos necesarios, por más que la estrategia de implantación no esté definida, ya que se trata de aspectos del proyecto completamente independientes.
    d) En el caso de sistemas de información normalmente es un aspecto que no requiere mayor esfuerzo.

18. La capacitación y el entrenamiento a usuarios
    a) En proyectos en los que la población a capacitar es muy grande, conviene comenzar la capacitación tan pronto como los requerimientos se vuelven estables, para poder culminarla junto con la fecha prevista de comienzo de la puesta en explotación.
    b) Son tareas que se pueden planificar y ejecutar de forma completamente independiente de las tareas de documentación del software.
    c) Son tareas que dado un determinado producto a construir, pueden estar, o no, comprendidas en el alcance de un proyecto, por lo que deben ser consideradas en el momento de definir ese alcance.
    d) Debe abarcar tan solo a los usuarios más importantes y que hacen un uso frecuente del sistema.

19. Respecto a la estrategia de implantación de un producto de software:
    a) Conviene definirla tan pronto como sea posible, ya que ciertas estrategias pueden requerir desarrollar software adicional específico, por lo que mientras no esté definida existe incertidumbre respecto al tamaño total del software necesario.
    b) En el caso de sistemas de información complejos, vitales para el funcionamiento de una organización, que involucren múltiples tipos de usuarios y con una población usuaria numerosa, conviene adoptar la estrategia big-bang ya que permite detectar de una vez todos los problemas relacionados con los procedimientos.
    c) Que sustituye a otro, si incluye el procesamiento en paralelo del sistema nuevo en modalidad de prueba por un período de al menos un mes permite verificar de forma adecuada el correcto funcionamiento de la totalidad del sistema.
    d) La ventaja más importante de la implantación paulatina es que evita la necesidad de convivencia entre el sistema nuevo y el viejo.

20. El mantenimiento de software es importante entre otras razones debido a que:
    a) Un producto de software en general cambia o se vuelve menos útil
    b) Si se realiza de forma adecuada, un producto de software puede vivir muchos años
    c) La calidad del mantenimiento puede afectar positiva o negativamente diversos atributos de calidad del software, entre ellos su mantenibilidad
    d) Todos los anteriores son válidos, a lo que se agrega que existe una tendencia al crecimiento de la participación de los costos del mantenimiento en los costos de la industria de software

21. Ejercicio de Liberación y Mantenimiento (8 puntos)

La empresa encargada del servicio de distribución de agua potable de la ciudad de Santa María cuenta con un portafolio de aplicaciones para soportar el negocio que totalizan 1.400.000 líneas de código de las cuales 1.000.000 son en lenguaje Cobol y y 400.000 en lenguaje C, en ambos casos con SQL embebido.

El mantenimiento de las aplicaciones es llevado a cabo por un equipo de 17 personas. Las aplicaciones existentes cuentan con muy poca documentación y fueron desarrolladas en su mayoría hace más de 16 años. Las reglas del negocio están en las cabezas del personal técnico y de los usuarios, y en los programas. La cantidad de solicitudes de cambio pendientes de atención es creciente y la distribución del esfuerzo muestra un peso creciente del mantenimiento y en particular del correctivo. Más abajo se presentan cuadros con el detalle de la evolución de estos datos.

El 80% del esfuerzo de mantenimiento se concentra en las aplicaciones relacionadas con la gestión de los clientes, consumos y facturación que corresponden a poco menos de la mitad de las líneas de código de la instalación.
El tratamiento del mantenimiento difiere de acuerdo al tipo. El mantenimiento correctivo recibe la máxima prioridad y cada problema se trata de atender enseguida que es reportado. Las correcciones se ponen en explotación tan pronto como sea posible. Los otros tipos de mantenimiento reciben menor prioridad y se tratan de forma planificada, con un calendario de liberación de nuevas versiones.

Año	Solicitudes pendientes al inicio	Solicits. nuevas	Solicits. cumplidas	% Esf. Desarrollo	%Esfuerzo Mantenim. Correctivo	% Esfuerzo Mantenim. Preventivo	%Esfuerzo Mantenim. Perfectivo	%Esf. M.Adap-tativo
2002	350	330	120	9	56	5	7	23
2003	560	280	110	7	62	4	4	23
2004	730	230	95	5	66	3	3	23
2005	865	198	89	4	70	2	2	22
2006	974	91	31	3	75	0	1	21

a) Discuta las diferentes estrategias que se pueden adoptar para reducir la carga de trabajo del mantenimiento correctivo y mejorar el servicio de mantenimiento, excluyendo la alternativa de sustituir el software por otro nuevo.

La empresa administra del orden de 400000 conexiones al servicio de agua potable. El consumo de agua lo registra en un PC de mano un operario que recorre un circuito de conexiones. La información del PC la transfiere por Internet a un nodo central al fin del día. A partir de los consumos se generan las facturas, las que se distribuyen a través de un servicio tercerizado. Entre la toma del consumo y la entrega de la factura pasan al menos 5 días hábiles. El cliente dispone de 4 días hábiles adicionales para pagar.
La empresa está evaluando sustituir el sistema actual de toma de consumo y facturación por uno nuevo, con el objetivo de reducir costos. Por un lado se plantea acortar los tiempos de facturación para reducir costos financieros y por otro, evitar el costo asociado a la distribución de las facturas.
En el nuevo sistema el cliente podrá imprimir su factura accediendo por Internet, o pagar directamente en un local de cobranza descentralizada mencionando el número de conexión.

b) Construya un WBS para las actividades de liberación de este producto. Discuta las posibles estrategias de liberación y de qué forma afectarían el proyecto.

---

22. Ejercicio Verificación (12 puntos)
El siguiente programa recorre una lista ordenada y elimina los elementos repetidos

```
public void eliminarRepConDefectos(Vector lista) {
    if (lista = = null || lista.size() <= 1) return;
    Object ant = lista.get(0);
    int index  = 1;
    do {
        Object actual = lista.get(index);
        if (actual.equals(ant)) lista.remove(index);
        else {
            ant = actual;
             index++;
        }
    }while (index < lista.size());
}
```

Parte a) (3 puntos) : Dibuje el grafo de flujo de control del método presentado

Parte b) (3 puntos) : Un camino es una secuencia única de aristas desde la entrada a la función (método) hasta su salida.
Un **camino simple** es un camino que no repite aristas.
Como en los loops es necesario repetir aristas se genera otro tipo de criterio de cubrimiento. El criterio de **cubrimiento de loops** establece que los loops del tipo while deben ser ejecutados cero veces y al menos una vez y los del tipo do-while deben ser ejecutados una vez y más de una vez.

Se pide: Identificar el mínimo de caminos necesarios para cumplir con los dos criterios a la vez (criterio de camino simple y criterio de cubrimiento de loops) para el método presentado.

Parte c) (3 puntos) : Generar un caso de prueba para cada uno de los caminos identificados en la parte b)

Parte d) (3 puntos) : Si ahora se considera el **criterio de condición/decisión**, ¿los casos generados en la parte c) cumplen con este criterio?
En caso que la respuesta sea negativa brinde un conjunto mínimo de casos de prueba que complementen a los casos de la parte c) de forma de cumplir con el criterio de condición/decisión.