- Cada pregunta **múltiple opción** contestada correctamente tiene un valor de 3,34 puntos. Cada pregunta incorrecta de la múltiple opción resta 1,11 puntos.
- **El puntaje total del examen es 100 puntos y se aprueba con 60 o más puntos.**

Múltiple Opción

1. Dado un escenario "real" de un proceso de desarrollo de software con restricciones de tiempo y recursos, frente a una liberación de una nueva versión de un producto existente, la cual incluye la corrección de defectos y funcionalidades nuevas, es importante tener en cuenta los siguientes aspectos:
   I. Priorización de las características/correcciones a probar de acuerdo al riesgo estimado (de no ser probada). Las más riesgosas deben probarse primero.
   II. Nivel de profundidad requerido para las pruebas (poca profundidad: solamente "casos felices", mucha profundidad: casos felices, casos inválidos, robustez, performance).
   III. Recursos disponibles (hardware/software, recursos humanos: habilidades, experiencia).
   IV. Ciclo de vida del software en el cual está inmersa esta liberación.
   V. Tipo de características/funcionalidades a probar y tipo de producto a verificar.

   Seleccione la opción correcta:
   a) Solamente I, II, y V son correctas
   b) Solamente I, y III son correctas
   c) Solamente II, III y IV son correctas
   d) Todas son correctas

---

2. Un EBP (Elementary Business Process / Proceso de Negocio Elemental):
   I. Es una tarea ejecutada por una persona en respuesta a un evento del negocio.
   II. Puede abarcar distintos lugares y momentos.
   III. Agrega valor de negocio mensurable.
   IV. Deja los datos en estado consistente.

   Seleccione la opción correcta:
   a) (I) y (II)
   b) (III) y (IV)
   c) (I), (III) y (IV)
   d) (I), (II), (III) y (IV)

---

3. El mantenimiento de software:
   a) Es una actividad que en general está bien considerada por los gerentes, lo que constituye un factor de motivación importante del personal que logra quedar asignado a este tipo de tareas.
   b) La evolución esperable de la calidad de un producto de software que está sometido a mantenimiento es que esta vaya mejorando de forma sostenida, de manera análoga a la evolución que experimenta la calidad de software durante el desarrollo.
   c) Conviene atender las solicitudes de modificación formuladas por los usuarios tan pronto como estas se reciben y liberar cada una tan pronto como sea posible. De esta manera se logra maximizar el nivel de satisfacción de los usuarios.
   d) Conviene establecer una política de versiones de forma tal que se liberen periódicamente versiones del producto, cada una de las cuales incorpora un conjunto de modificaciones debidamente aprobadas por el Comité de Control de Cambios.

---

4. Con respecto a la Arquitectura de Software:
   a) Diseñar y documentar explícitamente la arquitectura favorece la comunicación entre *stakeholders*.
   b) Los *idioms* son soluciones a problemas abstractos de diseño de software .
   c) El estilo arquitectónico a elegir depende más fuertemente de los requisitos funcionales que de los no funcionales.
   d) El estilo arquitectónico "*shared-data*" tiene como desventaja que resulta en una forma ineficiente de compartir grandes cantidades de datos.

---

5. Según Sommerville, ¿cuál de los siguientes requisitos NO es un requisito no funcional?
   a) La documentación elaborada durante el proceso de desarrollo de software debe ser entregada al cliente.
   b) El sistema debe permitir generar un conjunto de informes que contengan la lista de clientes y sus datos personales.
   c) Los informes generados deben estar en formato PDF.
   d) La generación de informes debe realizarse en un tiempo no mayor a 15 segundos.

6. Sobre los criterios (vistos en el curso) de terminación de las pruebas de software
   I. Finalizar las pruebas de sistemas cuando el tiempo asignado para las mismas se ha terminado es un criterio razonable de terminación de pruebas.
   II. Terminar las pruebas cuando todos los casos diseñados durante su planificación han sido ejecutados, es un criterio razonable de terminación de las pruebas..
   III. A nivel de pruebas unitarias un criterio razonable para culminar las pruebas, a veces combinado con otros criterios, es el criterio de terminación de pruebas basado en algún tipo de cubrimiento de código (p.e. cubrimiento de sentencias).
   IV. Una forma de estimar los defectos remanentes de un producto de software (para luego utilizar la estimación como dato para culminar las pruebas) es mediante la técnica de siembra de defectos (*fault seeding*).
   V. La detección de fallas por unidad de tiempo como criterio de terminación tiene el problema de agotamiento de casos de prueba. Al medir la cantidad de fallas que se detectan por unidad de tiempo (p.e. semanas) se puede detener las pruebas si ya no se están provocando una cantidad (y tipo) de fallas que justifique económicamente continuar con las pruebas. Sin embargo, si se están utilizando siempre los mismos casos de prueba, como puede ocurrir en las pruebas de regresión, lo que puede acontecer es que los casos ejecutados estén "agotados". Es decir, que pasan las pruebas porque se ha solucionado un problema concreto, pero al no variar el conjunto de casos de prueba la disminución de fallas por unidad de tiempo puede ser únicamente debido a dicho agotamiento de casos de prueba.

   Seleccione la opción correcta:
   a) I y II son las únicas correctas.
   b) II, III, IV, V son las únicas correctas.
   c) III, IV, V son las únicas correctas.
   d) IV, V son las únicas correctas.

---

7. Respecto a los atributos de calidad del software:
   a) Se clasifican de acuerdo a su visibilidad en Internos y Externos. Los Internos son visibles a todos y los Externos son solo visibles a los integrantes del equipo de desarrollo.
   b) Los atributos externos son visibles una vez que el producto ya está construido, lo que plantea el riesgo de que el cliente y/o usuario plantee recién en esa instancia que esos atributos no son adecuados.
   c) Los atributos de calidad internos: legibilidad del código y modularidad, permiten predecir de manera muy ajustada los atributos de calidad externos: adecuación al uso y tiempo medio entre fallas.
   d) Una ventaja del modelo de proceso en cascada es que permite evaluar de forma temprana los atributos de calidad externos, antes de que el producto esté construido de forma completa, lo que permite gestionar de una manera más adecuada los riesgos.

---

8. Es esencial que las comunicaciones dentro de un equipo de proyecto se puedan realizar de forma efectiva y eficiente. En este sentido, las comunicaciones son influenciadas por factores como:
   a) El tamaño del grupo: cuanto mayor sea la cantidad de integrantes, la efectividad y eficiencia de las comunicaciones disminuye. En este escenario, es posible que existan personas que sólo se comuniquen entre sí en pocas ocasiones.
   b) La estructura del grupo: en general, los integrantes de una estructura formal y jerárquica podrán comunicarse más eficazmente entre ellos que aquellos que se encuentran en una estructura informal. El jefe es el responsable de hacer que las comunicaciones fluyan debidamente en el grupo.
   c) La composición del grupo: los integrantes con personalidades similares tienden a llevarse mejor y tener menos conflictos, lo cual contribuye en la eficacia de las comunicaciones.
   d) (a). En este caso, la cantidad de enlaces de comunicación posibles es $n * (n+1)$.

---

9. Dados los siguientes proyectos y modelos de proceso:

Proyecto:
A - de alto riesgo, muy complejo, alcance no del todo definido, subconjuntos de funcionalidad definidos que resultan útiles para los usuarios, larga duración (dos años o más).
B – Riesgo medio, alcance definido, subconjunto de funcionalidad definido y útil para los usuarios, duración media (seis meses).
C – Riesgo bajo, simple, alcance definido, corta duración (seis semanas).

Proceso:
1 – Cascada
2 – En Fases con Evaluaciones Parciales Internas
3 – En Fases con Liberaciones Parciales en Producción
4 – De Prototipación

Marque la asignación que le parezca más adecuada. Si un mismo proyecto aparece asignado a más de un proceso, debe entenderse que cualquiera de los dos resultaría igualmente adecuado.
Para los procesos 2 y 3 se puede especificar además entre paréntesis el modelo de proceso a utilizar en cada fase. Por ejemplo 2(1) significa Proceso en Fases con Evaluaciones Parciales Internas y en cada fase se aplica Cascada. 3(2(1)) significa que en este caso en cada fase de 3 se aplica En Fases con Evaluaciones Parciales Internas y en cada fase de 2 se aplica Cascada.

a) A3(2(1)), A3(1), B3(2(1)), B3(1), C1, C4
b) A3(2(1)), A1, B2(1), B3(1), C1, C3(1)
c) A1, A4, B1, B4, C1, C4
d) A3(2(1)), A3(1), B3(2(1)), B3(1), C1, C2(3(1))

---

10. Con respecto a la estimación y planificación de un proyecto de software:
    a) A pesar de la complejidad de armar el cronograma, en la fase inicial del proyecto (antes de comenzar su desarrollo) es común que la estimación del esfuerzo requerido no tenga margen de error.
    b) Una de las ventajas de las técnicas de estimación es que alcanza con disponer de datos históricos sin importar si la tecnología utilizada en anteriores proyecto coincide con el proyecto actual.
    c) Una vez que se tienen los requisitos especificados es muy fácil estimar el tamaño del software a desarrollar.
    d) En general, es difícil obtener una estimación precisa del tamaño del código en etapas tempranas, ya que el tamaño final del software depende de decisiones de diseño, que probablemente no se hayan realizado en ese momento.

---

11. Acerca del mantenimiento de software, en el libro del curso se plantea que:
    a) Los defectos más costosos de corregir durante el mantenimiento son los errores introducidos al codificar.
    b) Existen tres tipos de mantenimiento de software: Corrección de defectos (donde los errores de requerimientos son los más difíciles de corregir), Adaptación al ambiente (puede cambiar el hardware o la plataforma), Nuevas funcionalidades o modificación de funcionalidades existentes (aparecen necesidades de la organización que deben considerarse en el software).
    c) Considerando toda la vida de un producto de software, el costo total del mantenimiento de software es generalmente considerablemente menor que el costo de desarrollo de ese producto
    d) Dado que el equipo que desarrolló el software es el mismo que lo mantiene, suele resultar poco costoso agregar funcionalidades luego de que el software está operativo.

---

12. Dado que las pruebas de unidad son costosas en esfuerzo se deben elegir casos de prueba que sean lo más efectivos posible. Por lo tanto:
    a) Ciertos casos de prueba tienen que mostrar que cuando se usa la unidad como se espera, la misma hace lo que se supone que debe hacer.
    b) La cantidad de casos de pruebas deben estar limitados antes de empezar el plan de verificación.
    c) Si hay defectos en la unidad, es seguro que los casos de prueba unitarios desarrollados siempre van a ejecutar el defecto y provocar fallas.
    d) Debido al alto costo las unidades no deberían probarse.

---

13. Sobre los niveles de madurez propuestos por el modelo CMMI, es interesante destacar que:
    a) En el nivel 2 los procesos están enfocados a proyectos mientras que en el nivel 3 lo están a la organización.
    b) En el nivel 3 los procesos están enfocados a proveer mediciones que serán optimizadas en el nivel 4.
    c) En el nivel 4 y 5 los procesos están enfocados a mejorar la relación costo/beneficio del desarrollo de software.
    d) Ninguna de las anteriores.

---

14. Con respecto al enfoque de desarrollo basado en la reutilización:
    a) Aumenta el margen de error en la estimación del costo del proyecto.
    b) Aumenta el tiempo de desarrollo y validación.
    c) Como el software reutilizado ha sido probado y testeado, debería ser mas confiable que el software nuevo.
    d) Los estándares de interfaz de usuario no pueden ser implementados como un conjunto de componentes reutilizables.

15. Con respecto al Diseño de la Arquitectura de Software:
    a) Es un proceso creativo por lo que el proceso puede variar según el sistema que va a ser desarrollado.
    b) (a) y la arquitectura de un sistema depende fuertemente de los requerimientos no funcionales.
    c) (b) y la arquitectura de un sistema puede ser diseñada en torno a un patrón de arquitectura o un "estilo", pero no ambos.
    d) Es una etapa temprana del proceso de diseño del sistema, y siempre debe llevarse a cabo en paralelo con actividades de especificación de requisitos.

16. Respecto a la liberación de software:
    a) Entre las ventajas de una estrategia de implantación paulatina se encuentra que se tarda más en obtener los beneficios del sistema nuevo.
    b) La principal ventaja de una estrategia de implantación paulatina es que se obtienen los beneficios completos del nuevo sistema desde el inicio mismo de la implantación.
    c) El momento más adecuado para comenzar la planificación de las actividades asociadas a la liberación de un producto de software es al inicio del proyecto, de forma de tomarlas en cuenta en el alcance del mismo.
    d) En aquellos casos en los que se optó por una estrategia de liberación paulatina, constituye una buena práctica desasignar los recursos asignados a la construcción de un producto de software al inicio del proceso de liberación, a los efectos disminuir los costos del proyecto.

17. El estilo y estructura de la arquitectura de un sistema depende los requerimientos no funcionales que se hayan especificado para ese sistema. ¿Cuáles de las siguientes afirmaciones son correctas?
    I. Si la *performance* es un requerimiento crítico, es recomendable que la arquitectura se componga de componentes grandes, de forma de minimizar su cantidad y las comunicaciones entre ellos.
    II. Si la *seguridad (security)* es un requerimiento crítico, la arquitectura debería tener una estructura en capas, de forma de localizar los activos más críticos en las capas más interiores.
    III. Si la *seguridad (safety)* es un requerimiento crítico, las operaciones relacionadas con la seguridad deben encontrarse distribuidas entre diferentes componentes. Esto reduce el costo de mantenimiento de estos componentes.
    IV. Si la *performance* es un requerimiento crítico, los componentes del sistema deberían estar distribuidos en la red, y no deployados en la misma computadora.
    V. Si la *disponibilidad* es un requerimiento crítico, la arquitectura debe incluir componentes redundantes, de forma que sea fácil reemplazarlos sin detener el sistema.
    VI. Si la *mantenibilidad* es un requerimiento crítico, es importante que los repositorios de datos se encuentren compartidos, principalmente entre consumidores y productores de datos.

    Seleccione la opción correcta:
    a) II, IV, V, VI.
    b) I, II, III, V.
    c) I, III, IV.
    d) I, II, V.

18. Sobre pruebas de sistemas con un enfoque incremental integrando componentes de diferentes equipos:
    a) Cuando se integran primero componentes que brindan servicios comunes, como acceso a red, a bases de datos y luego se agregan las componentes con funcionalidades específicas del sistema bajo prueba, el enfoque de integración incremental se llama *bottom-up*.
    b) (a) Cuando se integran primero las componentes que brindan las funcionalidades del sistema y sobre el final se integran las componentes que brindan servicios el enfoque se llama *top-down*.
    c) (b) Normalmente en este tipo de integraciones (*bottom-up* o *top-down*) hay que desarrollar código extra para simular ciertas componentes que aún no se hayan integrado.
    d) El desarrollo orientado a objetos y por componentes ha provocado que las pruebas de integración no sean necesarias. Por ende, luego de la prueba de componentes aisladas se pasa a probar el sistema completo con todas las componentes ya desarrolladas.

19. La Gestión de la Configuración:
    a) Es el arte de coordinar el desarrollo de software para minimizar la confusión.
    b) Es el arte de coordinar el desarrollo de software para maximizar la confusión.
    c) (a) y el propósito es maximizar la productividad minimizando errores.
    d) (a) y el propósito es minimizar la productividad minimizando errores.

20. Las métricas dinámicas de un producto de software se basan en:
    a) Mediciones sobre los recursos utilizados para su construcción como por ejemplo la cantidad de horas/persona de esfuerzo.
    b) Mediciones efectuadas durante la ejecución del producto como por ejemplo el tiempo de respuesta de una transacción.
    c) b) y en general existe una relación entre las métricas dinámicas y la calidad del producto.
    d) Mediciones sobre representantes del producto como el diseño, el código o la documentación, como por ejemplo la profundidad de los anidamientos.

21. Mantener la trazabilidad entre los requisitos, el diseño y el código que los implementa sirve para:
    a) Detectar errores en el relevamiento de los requisitos.
    b) Determinar que exista un proceso finito de costo accesible para determinar que el sistema cumple los requisitos.
    c) Analizar el impacto de los cambios en otras partes del sistema.
    d) Todas las anteriores.

22. En lo que refiere a las técnicas de registro y control de avance de un proyecto:
    I. Sirven para saber dónde está un proyecto, hacia dónde está yendo y poder comparar la situación real con la planificada.
    II. Uno de sus objetivos es informar el desempeño en costos y cronograma.
    III. Las técnicas de medición del avance de cada actividad se suelen clasificar según los criterios de tangiblidad del producto y duración del esfuerzo.
    IV. Todas las técnicas de control de avance del proyecto permiten hacer una predicción de cómo se comportaría este de mantenerse las tendencias observadas hasta el momento.

    Seleccione la opción correcta:
    a) Todas son correctas.
    b) Solo (I), (II) y (III) son correctas.
    c) Solo (I), y (IV) son correctas.
    d) Solo (II) y (III) son correctas.

23. Entre las cualidades más relevantes para un software de mensajería en Internet para dispositivos móviles están:
    a) Interfaz de usuario atractiva, Adecuación al uso, Tiempo de respuesta, Seguridad de la información (security).
    b) Disponibilidad, Adecuación al uso, Calidad de la documentación técnica, Calidad de la documentación de usuario.
    c) Seguridad de la integridad física (safety), Adecuación al uso, Tolerancia a fallas, Tiempo de respuesta.
    d) Modularidad, Facilidad de ser probado, Facilidad de mantenimiento, Legibilidad del código

24. Se entiende por Línea Base:
    a) El conjunto de elementos de configuración generados durante el ciclo de vida del producto.
    b) La última versión de los elementos de configuración que han sido liberados al cliente.
    c) Un conjunto de elementos de configuración, los cuales se encuentran controlados para una correcta gestión de sus cambios.
    d) Todos los activos críticos de la organización que son utilizados durante el proyecto de desarrollo.

25. Evaluar la factibilidad de un proyecto implica evaluar:
    I. La factibilidad técnica, es decir si puede construirse con las restricciones tecnológicas disponibles, y con el conocimiento y experiencia disponibles en mi equipo de proyecto, dentro de los plazos y el presupuesto establecido.
    II. La factibilidad operativa, es decir, analizar si habrá algo que haga que el sistema no funcione, como, por ejemplo, en la cultura organizacional, o, por ejemplo, si el sistema puede integrarse con otros sistemas en uso.
    III. La factibilidad institucional, que implica preguntarse si el sistema contribuye a los objetivos globales de la organización.
    IV. La factibilidad económica, es decir, si se dispone de los recursos financieros necesarios para su desarrollo.

    Seleccione la opción correcta:
    a) (I) y (IV)
    b) (I), (II) y (III)
    c) (II) y (III)
    d) Todas son correctas.

26. Respecto de los modelos y ciclos de vida del software:
    a) Un modelo de proceso en cascada en general es recomendable para proyectos largos (1 año o más) y complejos.
    b) Los modelos de proceso en fases en general no son compatibles con otros modelos de proceso, a excepción del modelo en cascada que es el modelo que se aplica en estos casos en cada una de las fases.
    c) Los modelos de proceso en general representan una guía de cómo llevar adelante el desarrollo/mantenimiento de un producto de software, que aplican mejor o peor dependiendo del tipo de proyecto. Los modelos pueden ser combinados y deben ser adaptados a la realidad del proyecto para el cual se aplican.
    d) Los ciclos de vida ágiles en general son recomendables para proyectos largos (1 año o más), con bajo riesgo y en los que se encuentra definido con precisión el alcance.

---

27. Con respecto a los requisitos:
    I. Los requisitos no funcionales tienen un alto impacto en la definición de la arquitectura del sistema.
    II. Los requisitos no funcionales incluidos en el alcance suelen ser críticos, ya que, si no se cumplen, puede que el sistema resulte inusable.
    III. El tiempo necesario de entrenamiento puede ser utilizado para medir la facilidad de uso.
    IV. El requisito: «El lenguaje de programación debe ser PHP», es un requisito funcional.
    V. La técnica de casos de uso resulta de gran ayuda para identificar requisitos no funcionales.

    Seleccione la opción correcta:
    a) Todas son correctas.
    b) Solo (I), (II) y (III) son correctas.
    c) Solo (I), (III) y (V) son correctas.
    d) Solo (II) y (V) son correctas.

---

28. Con respecto a las inspecciones de software:
    I. Son recomendadas para evaluar el desempeño de las personas que pertenecen al equipo de desarrollo.
    II. Las inspecciones de código sirven para detectar defectos y verificar el cumplimiento de los estándares. También ayudan a evaluar atributos de calidad tales como la mantenibilidad.
    III. Es recomendable que cada organización defina su propia *checklist* basándose en los estándares y prácticas que utiliza. Las *checklist* deberían actualizarse periódicamente, a medida que se encuentren nuevos tipos de defectos.
    IV. *Extreme Programming* (XP) exige utilizar inspecciones de código.
    V. Se ha mostrado que el testing detecta una mayor cantidad de defectos (efectividad) que las inspecciones.

    Seleccione la opción correcta:
    a) Todas son correctas.
    b) Solo (I) y (III) son correctas.
    c) Solo (II), (III) y (IV) y (V) son correctas.
    d) Solo (II) y (III) son correctas.

---

29. Para evaluar las características de un proceso pueden formularse, entre otras, las siguientes preguntas:
    a) El proceso está explícitamente definido y es comprensible para todos los actores?
    b) a) y ¿Son visibles los resultados del proceso durante su ejecución?
    c) b) y ¿Es medible y flexible?
    d) Todas las anteriores.

---

30. La técnica de entrevistas individuales o grupales para la obtención de requisitos:
    a) Tiene como ventaja que es una técnica orientada a las personas y es flexible.
    b) No es efectiva para obtener información respecto a los requisitos y restricciones de la organización.
    c) Es muy útil en caso de tener cientos de usuarios para relevar requisitos.
    d) Sirve para entender el problema del negocio, pero no el ambiente de operación.
