- Cada pregunta **múltiple opción** contestada correctamente tiene un valor de 2 puntos. Cada pregunta incorrecta de la múltiple opción resta 0,66 puntos.
- **El puntaje total del parcial es 50 puntos.**

Múltiple Opción

1. Los cambios en el software son inevitables. En cuanto a la evolución de software y al proceso de mantenimiento. Puede decirse que:

   a) El mantenimiento de software ocurre cuando el software ya está operativo, mientras que la evolución de software se refiere a los cambios durante el proceso de desarrollo, antes de ser liberado para su uso.
   b) La evolución del software habla de la naturaleza dinámica del software, mientras que el proceso de mantenimiento comprende las acciones a realizar para gestionar dicha naturaleza, mediante herramientas y procedimientos.
   c) b) y la etapa de servicio es la etapa en la que el software está operativo, pero no puede sufrir modificaciones producto de nuevos requerimientos.
   d) a) y el mantenimiento de software abarca la gestión de cambios y la gestión de casos de prueba de funcionalidades nominales.

2. Las gestión de ambientes está relacionada a la gestión de configuración del software e implica:

   a) Contar con herramientas de gestión de versiones distribuida basadas en delta para cada ambiente.
   b) Que el equipo de desarrollo prepare y mantenga un ambiente de pre-producción, igual al ambiente de testing. En este caso, el ambiente de pre-producción es usado para que el usuario final pruebe el sistema antes de pasar la nueva versión a producción.
   c) El esquema esperado presenta un ambiente de desarrollo, un ambiente de testing, un ambiente de pre-producción idéntico al ambiente de producción y, finalmente, un ambiente de producción.
   d) c) y cuando se encuentra un error en el ambiente de testing, por parte del equipo de pruebas, se regresa el cambio a desarrollo para su ajuste.

3. Con respecto a la arquitectura y diseño de software:

   a) La arquitectura de software busca expresar la estructura global de una aplicación. El objetivo de la arquitectura no es detallar la solución adoptada sino que es proveer una visión global de la misma, que permita, por ejemplo, simplificar su comprensión,
   b) (a) donde el concepto de abstracción refiere a identificar los elementos principales e ignorar los detalles no esenciales.
   c) (b) y el modelo 4+1 está compuesto por: la vista lógica, la vista de despliegue, la vista de desarrollo y la vista física, relacionados con los casos de uso o escenarios (+1).
   d) (c) y los patrones de diseño son soluciones a problemas detallados de diseño de software.

4. En cuanto a la mejora de procesos.

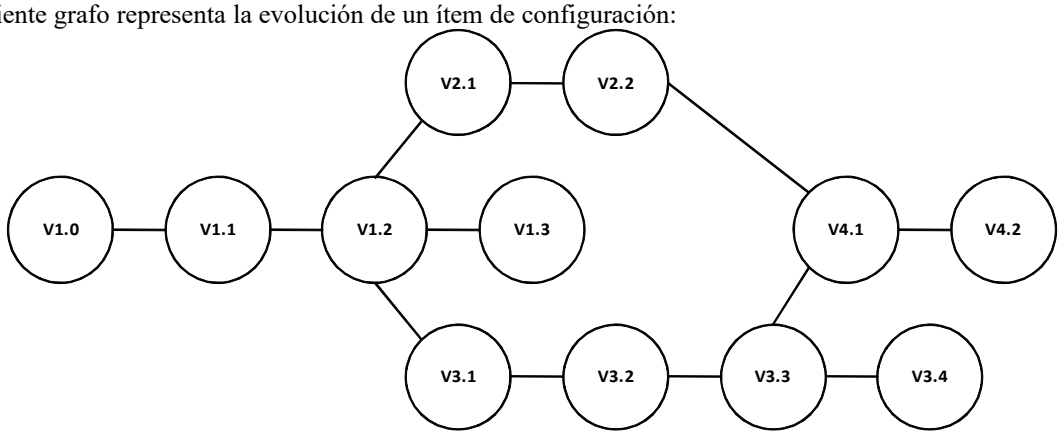
   a) El análisis de procesos comprende el estudio de los procesos para ayudar a entender sus características clave y cómo las personas implicadas realizan en la práctica dichos procesos.
   b) a) y posibles técnicas a utilizar son: encuestas, entrevistas, estudios etnográficos.
   c) b) y en general el análisis de procesos es más usado o conveniente en procesos que recién se han comenzado a utilizar.
   d) b) y conviene identificar las excepciones que hayan ocurrido y evaluar si no se trata de necesidades recurrentes que requieran cambios sobre el proceso.

5. Dada las siguientes afirmaciones sobre Modelo 4+1:

   I. El modelo describe la arquitectura del sistema a través de distintas vistas desde el punto de vista de diferentes interesados, como por ejemplo usuarios finales y desarrolladores.
   II. Las vistas del modelo 4+1 son: vista lógica, vista de proceso, vista de desarrollo y vista física. Además se considera una quinta vista, que describe la arquitectura a través de una selección de casos de usos o escenarios.
   III. La vista lógica soporta principalmente los requisitos funcionales.
   IV. El diagrama de despliegue forma parte de la vista física.

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

6. El siguiente grafo representa la evolución de un ítem de configuración:



a) En general la gestión de las versiones incluye la identificación de las versiones y de las liberaciones.
b) (a). Ejemplos de versiones representadas en el grafo son: 1.1 y 4.1
c) (b) y a partir de la versión 4.1 se genera una nueva codeline,
d) (c) que es un ejemplo de branching.
e) Ninguna afirmación es correcta.

……...

EJERCICIO – Liberación y Mantenimiento

La aplicación *HealthyFing* tiene como objetivo la venta de comida sana, en modalidad delivery, con beneficios especiales para estudiantes de la FING. Hoy en día ya existe una versión web de la aplicación y se busca sustituir la misma por una nueva versión mejorada y con mayor cantidad de funcionalidades. A su vez, se desea lanzar por primera ver una versión móvil de la aplicación.

Esta aplicación cuenta con 3 tipos de usuarios: los usuarios web y móvil que accederán a la misma para comprar la comida sana ofrecida por el sistema y los administradores web que gestionarán las comidas y los deliverys.

7. En relación al entrenamiento que se deberá brindar para la aplicación:
   a) Se debe entrenar tanto a usuarios finales como a administradores.
   b) (a) y los usuarios finales se les debe presentar cómo usar el sistema y las diferencias con respecto a la versión anterior. El entrenamiento de los administradores se puede hacer de la misma forma que el de los usuarios finales, ya que ambos entrenamientos tienen el mismo objetivo.
   c) (a) y dado el tipo de aplicación, como soporte para el entrenamiento de los usuarios finales es recomendable contar con ayuda en línea y con videos demostrativos (también en línea).
   d) (b) y es importante que esté pronta la documentación para dar soporte al entrenamiento antes de la liberación de la aplicación.

8. Con respecto a la liberación de la nueva versión de *HealthyFing*:
   a) Es necesario planear la estrategia de conversión ya que se debe sustituir un sistema por otro.
   b) (a) y antes de liberar la aplicación móvil al mercado se podría hacer una versión beta de la misma para ser utilizada por un conjunto inicial de usuarios de forma de obtener *feedback* y corregir los errores encontrados por los mismos.
   c) (b) y dado que la nueva versión tiene nuevas funcionalidades es posible que se tenga que hacer una conversión de datos.
   d) (c) y la estrategia de conversión paulatina se refiere a ejecutar el sistema antiguo y el sistema nuevo al mismo tiempo, uno en producción y el otro en prueba/control.

9. En relación a la versión que se encuentra en el mercado con referencia al mantenimiento, y en un contexto ideal y completo:
   a) es necesario tener definido la clasificación de incidentes/requisitos.
   b) (a) y conformado el Comité de Control de Cambios.
   c) (b) y el procedimiento de aceptación o rechazo del cambio.
   d) (c) y una herramienta para reporte de bugs y nuevos requisitos.

10. Se tiene una solicitud de cambio, seleccione la opción correcta
    a) En el caso que la misma sea realizada por el interesado principal del sistema, esta no se debe analizar, se debe realizar y liberar lo antes posible
    b) En el caso que la solicitud de cambio se clasifique como crítico o de emergencia, se prioriza realizar una reparación de emergencia al código y liberar el cambio, sobre la modificación del documento de requisitos y diseño.
    c) En el caso que la solicitud de cambio sea preventivo la aprobación del cambio será de forma automática.
    d) Todas las afirmaciones son correctas.

.....

11. Respecto a gestión de la configuración:
    a) La línea base está compuesta por toda la documentación del proyecto y el código del proyecto.
    b) a) y el alcance y grado de formalismo depende de los requerimientos.
    c) b) y un posible uso del ambiente de pre-producción (idéntico a producción), es la ejecución de pruebas de performance.
    d) Ninguna de las anteriores.

.....

EJERCICIO – Verificación y Validación

Dada la siguiente tabla que representa los plazos fijo de un banco. En las columnas se muestran los distintos intervalos de montos, los cuales deben ser como mínimo de $10.000. En las filas se encuentras los 4 plazos posibles, 31, 91, 181 o 367 días.

Depósito Plazo Fijo - mínimo $10.000		
Plazo	**Tasa de interés**	
	hasta $150.000	más de $150.000
31 días	6%	6,75%
91 días	6,5%	7,25%
181 días	7%	7,75%
367 días	8%	8,75%

```
Funcion tasainteres(monto, plazo){
        if( monto < 10.000 or plazo < 31 or plazo > 367 )
                error;
        else if (plazo = 31 or plazo = 91 or plazo = 181 or plazo = 367) {
                if (plazo = 31)
                        tasa = 6;
                else if (plazo = 91)
                        tasa = 6,5;
                else if (plazo = 181)
                        tasa = 6,5;
                else if (plazo = 367)
                        tasa=8;
        }

}
```

12. La cantidad de clases de equivalencia, considerando que todos los errores se consideran como una misma clase son:
    a) 12
    b) 8
    c) 7
    d) Ninguna de las anteriores

13. Dadas las siguientes afirmaciones:

    i. Si estoy aplicando clases de equivalencia, tendría al menos un caso de prueba por clase de equivalencia.
    ii. Si estoy aplicando clases de equivalencia, podría ocurrir que con un caso de prueba cubra más de una clase de equivalencia.
    iii. Si quisiera agregar la técnica de valores limites, estaría incorporando más clases de equivalencia.
    iv. Aplicando clases de equivalencia y valores limites juntos tengo dos ventajas, voy a detectar más fallas y ejecutar menos casos de prueba que solo haciendo clases de equivalencia.
    v. Un ejemplo de caso de prueba de valor limite seria: deposito=10.000, plazo=31.

Seleccione la opción correcta:

    a) Las afirmaciones i), ii), iii) son correctas
    b) Las afirmaciones i), ii) y iv) son correctas.
    c) Las afirmaciones i), ii), iii), iv) y v) son correctas.
    d) Ninguna de las anteriores.

14. Dado el pseudocódigo de la función *tasainteres*, seleccione la opción correcta.

    a) Si aplico cubrimiento de sentencias, necesito al menos 5 casos de prueba.
    b) a) y si aplico cubrimiento de decisión, también necesito al menos 5 casos de prueba.
    c) c) y si aplico cubrimiento de condiciones, también necesito al menos 5 casos de prueba.
    d) No puedo aplicar las técnicas de cubrimiento en este caso.

15. Si estamos escribiendo casos de prueba utilizando la técnica de valores límites:

    a) ($150.000, 91) y ($149.999, 91) son valores límites de entrada.
    b) ($9.500, 181) es un valor límite de entrada.
    c) Valores límites es una técnica de caja blanca.
    d) Dos de las anteriores son correctas

.....

16. Sobre estilos de arquitectura:

I. Un estilo Shared-data donde el consumidor sea notificado sobre cambios de interés es un pizarrón (blackboard)
II. Para aumentar la escalabilidad de una arquitectura cliente-servidor solo hace falta agregar más clientes
III. Una de las ventajas del estilo tubos y filtros es la reutilización de los filtros
IV. En un modelo en capas estricto una capa solo utiliza servicios de la capa inmediata inferior
V. En el único estilo donde cambiar la arquitectura de un sistema no es costoso es si partimos de un modelo cliente servidor en 3 niveles.

Selecciona la opción correcta:

a) IV, V.
b) III,V.
c) I,II, III.
d) I, III, IV.

17. Sobre la calidad del producto de software. Suponga que forma parte de un proyecto de mantenimiento de software con liberaciones del producto mensuales y con cientos de funcionalidades. Actualmente, están teniendo problemas de calidad pero no saben a ciencia cierta si son fallas en el producto o si también hay problemas en la instalación y configuración en el ambiente del cliente. ¿Qué actividades de aseguramiento y control de la calidad le parece que convendrían más?
    Tenga en cuenta que un mes es poco tiempo y las actividades que usted agregará al proceso de desarrollo tienen también un costo asociado:

    a) Comenzaría con un enfoque de encontrar todos los defectos del producto, para eso usaría Revisiones e Inspecciones. Que luego podría complementar con Inventario de Pruebas
    b) Comenzaría tratando de identificar si hay errores de instalación y en caso afirmativo propondría Filtros de Defectos. Luego podría comenzar a construir un Inventario de Pruebas y utilizaría un enfoque de Pruebas basadas en Riesgos para priorizar las pruebas antes de cada entrega
    c) Comenzaría tratando de identificar si hay errores de instalación y en caso afirmativo propondría Filtros de Defectos. Luego propondría un enfoque de encontrar todos los defectos del producto, para eso usaría Revisiones e Inspecciones.
    d) Ninguna de las anteriores es adecuada para la realidad planteada.

18. Respecto a diseño de sistema:

    a) El proceso de definición de la arquitectura es un proceso creativo que varía para diferentes sistemas.
    b) a) donde los requerimientos no funcionales son elementos relevantes para dicho proceso.
    c) b) por lo que deben ser relevados al inicio del proceso de relevamiento de requisitos.
    d) Ninguna de las anteriores es correcta.

19. Sobre la calidad del producto de software, dadas las siguientes técnicas y sus definiciones, indique qué definición corresponde a cada técnica (utilizando P(I) si la definición P corresponde a la técnica I):

    I. Revisiones
    II. Inspecciones
    III. Pruebas basadas en riesgos
    IV. Inventarios de pruebas
    V. Filtros de defectos
    VI. Análisis de tendencias de defectos

O. Se construye una lista con los ítems a ser probados (en general se hace una lista de funcionalidades) se describe la complejidad (naturaleza del ítem, cómo es su complejidad estructural o de algoritmos) y su criticidad (qué tan crítico es para el cliente). Luego se gestionan las pruebas utilizando la gestión de riesgos y también la complejidad y la criticidad. Si hubiera poco tiempo para las pruebas, por ejemplo, se propondría probar los ítems más críticos y complejos primero.

P. Tienen como objetivo encontrar bugs en el programa en desarrollo. Incluye una cuidadosa revisión, línea por línea, del código fuente del programa. Usualmente se usan checklists y se realizan de forma individual sobre código fuente propio o ajeno.

Q. Luego de encontrar defectos se puede tratar de evitarlos. Esto se puede hacer por ejemplo utilizando checklists en pasos esenciales del proceso de construcción o en puntos dónde se encontraron muchos defectos previamente.

R. Un grupo de personas examina el software y su documentación asociada en busca de problemas potenciales y la falta de conformidad con los estándares. En este sentido, es un proceso público de detección de errores, comparado con el proceso más privado de prueba de componentes.

S. Al realizar el seguimiento periódico de los riesgos se debe evaluar la necesidad de realizar pruebas al software. Esto puede ocurrir ante cambios en el software, cambios en el ambiente, cambios en los datos u otros tipos de riesgos.

T. Se analizan los últimos reportes de incidentes y se toman medidas, por ejemplo, no liberar una versión si los últimos testeos encuentran cada vez más errores (hay una pendiente positiva).

a) O(IV), P(II), Q(V), R(I), S(III), T(VI).
b) O(IV), P(I), Q(V), R(II), S(III), T(VI).
c) O(IV), P(II), Q(VI), R(I), S(III), T(V).
d) O(IV), P(II), Q(V), R(III), S(I), T(VI).

20. Un buen diseño de software debe presentar la siguientes características:

    a) Ser completo
    b) a), rastreable
    c) b), abstracto
    d) c) y razonable.

21. Sobre verificación y validación:

    i. Verificar implica comprobar que el sistema cumple con los requerimientos especificados, mientras que validar busca comprobar que el software hace lo que el usuario espera.
    ii. Se deben considerar riesgos y analizar el costo-beneficio de realizarla.
    iii. Desempeño, instalación, aceptación, integración son tipos de pruebas
    iv. Las técnicas estáticas permiten verificar productos como el código fuente y documentos
    v. El plan de V&V debe indicar que tipos de pruebas se ejecutaran, así como la planificación de las mismas.

    a) Todas las opciones son correctas
    b) No hay opciones correctas
    c) Cuatro opciones son correctas
    d) Tres opciones son correctas

22. Con respecto a la construcción de software:

I. Una cantidad significativa del trabajo de diseño se realiza en construcción. A su vez, tanto las pruebas unitarias como parte de las pruebas de integración se realizan durante la construcción.
II. La anticipación al cambio afecta como el software es construido. En este sentido, una de las tácticas que se puede utilizar es, para valores que pueden cambiar, registrarlos en la base de datos o en un archivo de configuración en lugar de ponerlos fijos.
III. La elección del modelo de desarrollo a utilizar no afecta la construcción.
IV. La reutilización, por lo general, permite reducir los costos del desarrollo.
V. Las APIs tienen como desventaja que aumentan el acoplamiento.

Seleccione la opción correcta:
a) Todas son correctas.
b) Sólo 2 opciones son correctas.
c) Sólo (I), (II) y (III) son correctas.
d) Sólo (I), (II) y (IV) son correctas.

23. Respecto a Construcción:

    a) Refiere a la creación detallada del software mediante una combinación de codificación, verificación, pruebas unitarias, pruebas de integración y debugging.
    b) (a) y la entrada de dicho proceso es la salida del proceso de diseño, aunque algunas tareas de diseño pueden realizarse durante el proceso de construcción.
    c) Refiere a la creación detallada del software pero no involucra actividades de verificación.
    d) Ninguna opción es correcta.

24. Sobre una evaluación de mejora:
    a) Si la misma se realiza previo a la puesta en producción habrá retroalimentación de la entrega y se contará con todo el equipo.
    b) Si estamos buscando buena retroalimentación un buen momento para realizarla será poco tiempo después de la puesta en producción.
    c) a) y b) son correctas, ya que lo recomendable es hacer una evaluación antes de la puesta en producción y otra evaluación complementaria después. Esto no involucra demasiado esfuerzo adicional.
    d) Es de especial valor en una evaluación centrarse en los aspectos negativos, ya que los positivos es esperable que se repitan.

25. Respecto a Arquitectura y diseño
    a) Tres conceptos claves para la calidad de un diseño son: bajo acoplamiento, alta cohesión y principio abierto-cerrado.
    b) a) y el acoplamiento refiere a qué tanto se conocen entre los módulos.
    c) b) y la cohesión refiere a que tan vinculados están los elementos que se encuentran en un mismo módulo.
    d) c) y el principio de abierto-cerrado refiere a que debe ser abierto con la extensión y cerrado con la modificación.