1. La ingeniería de software:

a) está relacionada con la ingeniería de sistemas en la medida que cuando construimos software estamos también construyendo un sistema compuesto al menos por hardware, software y personas
b) (a) y se ha visto impactada por la necesidad de reducir los tiempos para salir al mercado, lo que ha llevado a buscar alternativas al modelo de proceso en cascada
c) (b) y se ha visto impactada por el aumento del costo del petróleo que ha generado un aumento significativo del costo relativo del hardware respecto al del software
d) (c) y consiste esencialmente en el desarrollo de software

---

2. Respecto a modelos de proceso

a) El modelo de proceso en espiral es especialmente adecuado para proyectos pequeños, con requerimientos estables.
b) Los modelos propuestos por Abdel-Hamid vistos en el curso (Factores que inciden en la productividad y Estructura de desarrollo de software) son de tipo prescriptivo, es decir, prescriben una forma de llevar a cabo el desarrollo
c) El desarrollo en fases tiene como principales ventajas el logro de resultados tempranos, con lo que por lo menos parte de los beneficios se obtienen antes, y el poder contar antes con retroalimentación de parte de cliente y usuarios respecto a los atributos externos de calidad del producto y su adecuación al uso.
d) Los modelos de proceso Especificación Operacional y Transformacional, son ampliamente utilizados en la industria, en proyectos de diverso porte.

---

3. Los métodos algorítmicos de estimación:

a) son modelos que reflejan relación entre esfuerzo y factores que lo influencian, y son de la forma E = (a+b*S^c) m(X)
b) a) y el tamaño es el factor más influyente
c) intuitivamente podemos afirmar que c debería ser >1, con lo cuál la productividad decrecería con el tamaño.
d) todas las anteriores.

---

4. Dada una consulta que despliega datos del cliente. Y dada otra consulta que despliega los mismos datos más la edad que calcula a partir de la fecha de nacimiento, ¿cuántas y qué tipo de transacciones se deben contar?

a) una sola EQ
b) dos EQ
c) una sola EO
d) una EQ y una EO

---

5. La obtención y análisis de requerimientos:

a) involucra solo al equipo de desarrollo
b) debe registrar únicamente lo que el cliente pide explícitamente
c) puede ser influenciada por factores políticos
d) da como salida un conjunto de requerimientos que quedarán fijos de ahí en adelante para el proyecto.

---

6. Técnicas de obtención de requerimientos son:

a) la investigación de antecedentes
b) a) y las entrevistas, que deben estar en todo proyecto y se deben hacer con todos los usuarios que van a utilizar el sistema
c) b) y una encuesta o cuestionario, que me permite detectar nuevos requerimientos.
d) c) y los casos de uso que permiten detectar aspectos no funcionales.

---

7. La planificación del proceso de administración de los requerimientos implica:

a) decidir la forma de identificación de los requerimientos, que debe ser nemotécnica.
b) definir las actividades necesarias para evaluar el impacto y el costo de un cambio propuestos
c) definir políticas de rastreo para registrar y mantener información del dueño del requerimiento, la dependencia entre ellos y la trazabilidad al diseño.
d) definir la utilización de herramientas case para relevar y priorizar los requerimientos, administrar el cambio y administrar el rastreo.

---

8. la separación de los documentos de diseño en diseño conceptual y diseño técnico

a) facilita la comunicación del diseño al cliente cuando esta se realiza en base a ambos documentos
b) facilita la comunicación del diseño a los constructores del sistema cuando esta se realiza en base al documento conceptual
c) agrega la tarea de mantener la trazabilidad entre ambos documentos cuando alguno de los dos es modificado
d) ninguna de las anteriores

---

9. Entre los beneficios de realizar una definición explicita de la Arquitectura de Software se encuentran

a) mejorar la comunicación entre los interesados: cliente – diseñadores, diseñadores – desarrolladores

b) clarificar las intenciones de diseño evitando que la Arquitectura concebida se pierda, aportando al mantenimiento del sistema
c) proporcionar bases para el análisis del diseño en cuanto a evaluación de características requeridas o deseables como el desempeño
d) todas las anteriores

---

10. Un patrón de diseño plantea una solución a un problema en un contexto determinado
a) por ejemplo, el patrón Proxy se plantea como una solución al problema de cambiar la forma de distribución del software
b) a) y el patrón model-view-controller se plantea como una solución al problema de mantener la funcionalidad del núcleo independiente de la interfaz de usuario
c) b) y las consecuencias de las decisiones de implementación que se plantean en el patrón son aspectos esenciales de la definición del mismo
d) c) y parte importante de la definición del patrón consiste en establecer también para que problemas no aplica

---

11. Entre las características de un buen diseño se encuentra la independencia de los elementos de software definidos donde
a) la medida de esta independencia está dada por la cohesión y acoplamiento entre módulos, donde lo que se busca obtener es baja cohesión y alto acoplamiento
b) la cohesión de un módulo refiere a cuan conectado está el módulo con el resto de los módulos definidos
c) el acoplamiento de un módulo refiere a cuan focalizado está el módulo en resolver determinado aspecto del problema
d) cuanto mayor es la independencia se facilitan aspectos como la comprensión y el mantenimiento del diseño

---

12. Dadas las siguientes afirmaciones indique cuál respuesta es correcta
1) Descubrir defectos es uno de los objetivos de la verificación
2) Evaluar la calidad de los productos es uno de los objetivos de la verificación
3) La corrección de los defectos detectados es parte de la verificación
a) Se cumple 1 pero no se cumple 2
b) Se cumplen 1 y 2
c) Se cumple 2 pero no se cumple 1
d) Se cumplen 1, 2 y 3

---

13. Dadas las siguientes afirmaciones indique cuál respuesta es correcta
1) Es normal en la industria de software que las pruebas unitarias las realice un equipo especializado de testing y no el equipo de desarrollo
2) El conocimiento de las interfaces y la forma en la que interactúan las componentes de un sistema es importante para realizar las pruebas de integración
3) Según Myers el autor de un programa tiende a cometer los mismos errores al construirlo que al verificarlo
a) Se cumplen 1, 2 y 3
b) Se cumple 1 y 2
c) Se cumple 2 y no se cumple 3
d) Se cumplen 2 y 3 y no se cumple 1

---

14. Dadas las siguientes afirmaciones indique cuál respuesta es correcta
1) El testing no puede demostrar la ausencia de defectos
2) Las técnicas estáticas de verificación son efectivas en la detección temprana de defectos
3) El plan del proceso de verificación para un proyecto de desarrollo de software se debe realizar al principio del mismo. Este plan no se modifica durante el proyecto ya que al modificarlo se pueden sacar recursos o tiempo de dedicación al testing debido a los plazos de entrega del producto de software.
a) Se cumplen 1 y 2
b) Se cumple 1 y no se cumple 2
c) Se cumple 2 y no se cumple 1
d) Se cumplen 1, 2 y 3

15.
a) Cumplir con cierto criterio de cubrimiento de código al ejecutar las pruebas (y que todas las pruebas ejecuten correctamente) puede ser adecuado como criterio de terminación de las pruebas unitarias.
b) (a) y la *siembra de defectos* es una técnica para estimar la cantidad remanente de defectos
c) La terminación de las pruebas basada en la cantidad de defectos detectados por unidad de tiempo establece que si se detectan menos de 5 defectos por semana durante el testing de sistema las pruebas deben culminar.
d) Ninguna de las anteriores.

---

16. La planificación de la liberación de un sistema de software

a) Conviene comenzarla al comienzo del proyecto, para incluir sus particularidades en el alcance del proyecto
b) (a) y completarla y ajustarla antes de proceder a la liberación en sí, incorporando lo aprendido en el proyecto
c) Conviene completarla junto con la especificación de requerimientos
d) Conviene comenzarla cuando el producto está terminado

---

17. Respecto al mantenimiento
a) Conviene que los desarrolladores atiendan directamente los requerimientos de los usuarios, realicen los cambios y los pongan en explotación tan pronto como les resulte posible
b) (a) y frecuentemente la modificación del código insume la mayor parte del esfuerzo total aplicado a cada cambio
c) Conviene conformar un Comité de Control de Cambios que reúna la visón de Cliente, Usuarios y Desarrolladores, responsable por evaluar, priorizar y autorizar los cambios al sistema en explotación
d) (c) y frecuentemente el análisis de impacto insume buena parte del esfuerzo total aplicado a cada cambio

---

18. El modelo CMMI introduce las siguientes áreas de proceso de la categoría de ingeniería:
a) Análisis de Riesgo en el nivel 2
b) Verificación y Validación en el nivel 3
c) b y Desarrollo de los Requerimientos
d) Ninguna de las anteriores

---

19. Uno de los métodos de evaluación de procesos es el análisis post-mortem que se recomienda para:
a) facilitar el aprendizaje de la organización sobre la base de los aciertos, dificultades y sus causas
b) detectar los problemas y establecer los culpables
c) a y conocer el grado de satisfacción del cliente
d) c y estudiar la evolución de las estimaciones

---

20. La construcción de un modelo de calidad de un producto involucra los pasos siguientes:
a) identificar sus atributos de alto nivel y sus componentes
b) identificar y clasificar las propiedades de calidad más significativas de sus componentes
c) establecer la relación entre las propiedades y los atributos definidos, evaluar y refinar el modelo
d) todas las anteriores

---

21) Ejercicio Verificación (25 puntos)

Se define una clase Dimensión con los siguientes métodos (la clase contiene otros métodos que no se van a testear, por ejemplo, el ingreso de elementos a un objeto Dimension:

```
public class Dimension {
    public Dimension(int[] tamanioDimensiones)
    public int cantidadDimensiones()
    public int tamanioDimension(int nroDimension)
    public Object elementoAt(int[] posicion)
}
```

`public Dimension(int[] tamanioDimensiones)`
Crea un objeto de la clase Dimension. El objeto va a tener tantas dimensiones como el tamaño del array pasado como parámetro. El tamaño de cada dimensión está determinado por los valores enteros contenidos en el array. Tanto la cantidad de dimensiones como el tamaño de cada una son mayores o iguales que uno.
Por ejemplo, Dimension({2,3}), crea un objeto de dos dimensiones. La primera dimensión es de tamaño 2 y la segunda dimensión es de tamaño 3. En definitiva es una matriz de 2x3.

`public int cantidadDimensiones()`
Devuelve la cantidad de dimensiones que tiene el objeto. Para el ejemplo este método devuelve el valor 2.

`public int tamanioDimension(int nroDimension)`
Devuelve el tamaño de la dimensión pasada como parámetro. Las dimensiones se numeran de 1 a n. La dimensión número 1 es la que aparece en el lugar 0 del array pasado como parámetro al construir el objeto. La dimensión número 2 es la del lugar 1 y así sucesivamente. Para el ejemplo, tamanioDimension(1) devuelve el valor 2. tamanioDimension(2) devuelve el valor 3.

`public Object elementoAt(int[] posicion)`
Dada una posición, pasada como parámetro como un array de enteros, devuelve el elemento que contiene el objeto Dimensión en esa posición.

Para el ejemplo, elementoAt({2,2}), devuelve el objeto que se encuentra en la posición 2,2 de la matriz. Las posiciones en cada dimensión van de 1 a n, siendo n el tamaño de esa dimensión.

Parte a)
En caso de que algún método esté especificado de forma incompleta complemente la especificación de forma clara y concisa.

Parte b)
Escriba casos de prueba interesantes (probar cada *parte* de la especificación) para cada uno de los métodos presentados. Debe considerarse lo agregado en la parte a). Se sugiere tener presente la técnica de partición en clases de equivalencia y valores límite.

---

22) Ejercicio de WBS (15 puntos)

Una empresa de software maneja una aplicación Clipper para el registro de horas de sus empleados, y desea cambiarlo por una aplicación web, que desarrollará ella misma.
Construya un WBS para planificar este proyecto.