EXAMEN DISEÑO SOFTWARE 2017

1.- ¿Qué elementos de UML pueden ser instanciados?

a. Solo las clases
b. Clases y paquetes
c. Casos de uso, actores, clases, componentes y nodos entre otros
d. Absolutamente todos los elementos

2.- Aparte de capturar requisitos, los modelos de casos de uso también sirven para:

a. Establecer la frontera entre el sistema y su entorno
b. Representar la línea base de arquitectura
c. Mostrar el flujo de ejecución a alto nivel
d. Ilustrar el ensamblado y la topología del sistema

3.- Un actor deberá asociarse a un caso de uso:

a. Si se encuentra dentro del mismo paquete
b. Sólo si desencadena la ejecución del escenario principal
c. Sólo si es el receptor de la información producida por ese caso de uso
d. Si obtiene algún beneficio de la ejecución de ese caso de uso

4.- Si se necesita ampliar de forma controlada la funcionalidad descrita por un caso de uso, ¿qué mecanismo debería ponerse en práctica?

a. Realización
b. Inclusión
c. Extensión
d. Generalización

5.- ¿Qué se entiende por ingeniería inversa?

a. Ciclo de vida que antepone el diseño de las pruebas a las tareas de diseño y codificación
b. Técnica de modelado para deducir clases a partir de los objetos previamente diseñados
c. Reconstrucción de un modelo a partir de código
d. Método de construcción de software basado en el paradigma orientado a objetos

6.- Los modelos de casos de uso recogen propiamente el siguiente tipo de requisitos:

a. Funcionales
b. De evolución
c. De interfaz
d. De calidad

7.- La siguiente relación entre casos de uso no está contemplada:

a. Asociación

b. Inclusión

c. Extensión

d.Generalización

8.- Un extremo navegable en una asociación entre dos clases indica:

a. Que la responsabilidad de navegar recaerá sobre las instancias de la clase del extremo opuesto.

b. El sentido en que fluye la información entre objetos en tiempo de ejecución

c. Asimetría en cuanto a las multiplicidades con las que participa cada una de las dos clases

d. El sentido de lectura de la etiqueta de la asociación

9.- La generalización es un tipo de relación…

a. Estructural entre elementos

b. Que resalta que los cambios en un elemento pueden afectar a otro que lo usa

c. En virtud de la cual las instancias del padre pueden ser sustituidas por las instancias del hijo

d. A dos bandas en la cual uno de ellos especifica un contrato que el otro garantiza cumplir

10.- En la tarjeta CRC de una clase se harán constar:

a. Su clasificador y sus relaciones

b. Su nombre cualificado y los retornos de sus operaciones

c. Los casos de uso que la originan y los roles que desempeña

d. Sus responsabilidades y colaboradores

11.- ¿Qué tipos de mensajes pueden aparecer en un diagrama de interacción?

a. Eventos de disparo y acciones

b. Llamadas, retornos y señales

c. Llamadas, señales, excepciones e interrupciones

d. Llamadas, señales, retornos, creaciones y destrucciones

12.- ¿Cuál de las siguientes afirmaciones es verdadera respecto a los diagramas de colaboración y los diagramas de objetos?

a. Ambos visualizan instancias, enlaces y mensajes, pero los diagramas de objetos no son capaces de ofrecer el orden temporal de los mensajes

b. Ambos muestran objetos y enlaces, pero los diagramas de objetos no permiten especificar el estado en el que se encuentran dichos objetos

c. Los diagramas de objetos vienen a ser diagramas de colaboración sin mensajes

d. Los diagramas de colaboración no representan comportamiento

13.- En el contexto de los diagramas de interacción, ¿en qué se distingue una llamada del envío de una señal?

a. Las llamadas son síncronas, mientras que las señales son asíncronas

b. Las llamadas no tienen valores de retorno

c. Las llamadas son asíncronas, mientras que las señales son síncronas

d. Las señales no admiten el paso de parámetros

14.- El mecanismo de extensibilidad que UML contempla para crear nuevos bloques de construcción se denomina:

a. Valor etiquetado

b. Estereotipo

c. Patrón de construcción

d. Restricción

15.- Dado que todo patrón describe una estructura y un comportamiento, en UML se representará como…

a. Escenario

b. Colaboración

c. Interacción

d. Secuencia

16.- ¿Cómo se refleja en un diagrama estructural que una clase sea responsable de crear instancias de otra?

a. Colocando una realización dirigida desde la primera hasta la segunda

b. No se puede hacer, pues esto tiene lugar en tiempo de ejecución

c. Se puede mostrar con una restricción expresada en OCL

d. Mediante una dependencia adecuadamente estereotipada

17.- Lo que diferencia a una agregación de una composición es que…

a. La agregación es una asociación

b. La composición es una relación entre un todo y sus partes

c. En una composición la vida de las partes está fuertemente ligada a la del todo

d. En una agregación las partes no se pueden compartir entre varios todos

18.- ¿Qué ventaja presenta el paradigma orientado a objetos frente al clásico?

a. Al propiciar un diseño centrado en la estructura, facilita la evolución de los sistemas

b. La elección de uno u otro es una mera cuestión de preferencias

c. Por fomentar un diseño que gira en torno a la funcionalidad, simplifica el mantenimiento

d. Ninguna, ya que todo ingeniero sabe que el paradigma procedimental es netamente superior a él

19.- Si una operación se maneja a nivel de clase, en UML se dice que…

a. Está sobrecargada

b. Es estática

c. Es abstracta

d. Tiene alcance de clasificador

20.- ¿Cuál de las siguientes afirmaciones sobre interfaces es falsa?

a. Fijan una frontera entre el comportamiento deseado para una abstracción y su implementación

b. No especifican estructura ni implementación

c. Pueden generalizar a clases o a otras interfaces

d. Especifican contratos que deben ser satisfechos por clases o componentes

21.- ¿Cuál de los siguientes no es un uso previsto para los diagramas de clases?

a. Modelado de vocabulario de un dominio de aplicación

b. Modelado de un flujo de trabajo (“workflow”)

c. Modelado de la parte estructural de una colaboración

d. Modelado del esquema conceptual de una base de datos

22.- ¿Cuál de las siguientes afirmaciones acerca de los patrones de diseño es falsa?

a. Mejoran apreciablemente la documentación de los sistemas

b. Reducen significativamente la complejidad del sistema

c. Hacen a los sistemas más fáciles de ampliar y modificar en el futuro

d. Habitualmente sirven para desacoplar diferentes partes de un sistema para que puedan evolucionar independientemente

23.- Dado que, en cierto modo, tanto paquetes como componentes encapsulan clases, una diferencia fundamental entre ambos es que…

a. Los componentes no se pueden instanciar

b. La misma clase puede aparecer en varios componentes

c. En un paquete no puede haber interfaces

d. Los paquetes soportan modelado físico

24.- ¿Qué representa un modo en un diagrama de despliegue?

a. Un mecanismo de propósito general para agrupación lógica de elementos de modelado

b. Un recurso físico dotado habitualmente de capacidad de cómputo y de almacenamiento

c. Una parte reemplazable de un sistema que se ajusta e implementa un conjunto de interfaces

d. Una colección de operaciones que define un determinado servicio

25.- ¿Para qué sirven las calles o carriles (“swimlanes”) en los diagramas de actividades?

a. Para visualizar la realización de actividades en paralelo mostrando quién se responsabiliza de la ejecución de cada una de ellas

b. Para manejar subestados concurrentes

c. Para bloquear transiciones prohibidas entre actividades

d. Para manejar comportamiento opcional

26.- ¿Qué patrones de diseño constituyen la columna vertebral de la arquitectura MVC?

a. Composite, Strategy y Observer

b. Facade, DAO y Abstract Factory

c. Composite, Mediator y Strategy

d. Mediator, Visitor y Comand

27.- Si se aplica Abstract Factory…

a. Habrá tantas fábricas como familias y una operación en ella por cada producto

b. Habrá tantas fábricas como productos y tantas operaciones en ellas como familias

c. Habrá una fábrica por cada producto

d. Cada fábrica se especializará en el subconjunto de productos de esa familia

28.- Dada la similitud entre Adapter y Bridge, la diferencia esencial entre ambos patrones radica en que:

a. Adapter es un patrón de comportamiento

b. Bridge hace que las cosas antes de ser diseñadas y Adapter lo hace después

c. Adapter hace que las cosas funcionen antes de ser diseñadas y Bridge lo hace después

d. Bridge es un patrón estructural

29.- ¿Cuántos niveles de agrupación admite Composite en tiempo de ejecución?

a. Dos

b. Tantos como niveles se hayan establecido estructuralmente vía generalización

c. Su número es ilimitado

d. Vendrá determinado por el número de hijos que admita un grupo

30.- Teniendo en cuenta que un decorador soporta la misma interfaz que los objetos "decorables", ¿qué implementación ofrecerá una clase decorador concreto para una operación ajena a su responsabilidad?

a. Sobrescritura del método de su padre para evitar que esa petición se propague

b. Lanzamiento de una excepción

c. Invertirá sus acciones, ejecutando el comportamiento añadido antes de propagar la petición

d. Ninguna

31.- ¿Propagará siempre un *proxy* las peticiones a su representado?

a. Siempre, y cuando vuelva le flujo aprovechará para hacer su trabajo

b. Dependerá del tipo de proxy y del origen y naturaleza de la petición

c. Sí, para lo cual traducirá la sintaxis del cliente a la interfaz del representado

d. Nunca, ya que precisamente es el representado quien delega comportamiento en su proxy

32.- En UML es admisible que una clase…

a. Especialice a varias clases y realice varias interfaces

b. Especialice a varias clases y realice una sola interfaz

c. Especialice a una clase como mucho y realice varias interfaces

d. Especialice a una clase como mucho y realice una sola interfaz

33.- Al aplicar el patrón *Observer*…

a. Se aíslan sujeto y suscriptores

b. Se desacoplan sujeto y suscriptores

c. Los sujetos delegan en los observadores

d. Los sujetos se sincronizan con los observadores

34.- Si se incorpora el patrón *State*, ¿podrían compartirse los objetos estado entre diferentes contextos?

a. Sí, sobre todo si esos objetos carecen de “estado”

b. No, porque los estados son ayudantes de visibilidad privada

c. Sí, porque la idea es que un objeto estado reparta las peticiones entre varios contextos

d. No, porque el contexto delega toda petición en su estado y lo necesita en exclusiva

35.- Si al aplicar *Strategy* no todos los algoritmos necesitan la misma información de su contexto…

a. Será mejor no adoptar ese patrón sino *Template Method*

b. El contexto necesitará mayor acoplamiento con las estrategias

c. Es preferible que cada estrategia navegue al contexto para acceder a la información que requiera

d. Es probable que algunos algoritmos reciban información superflua.

36.- Si se aplica el patrón *Decorator*…

a. Se añadirá una operación por cada responsabilidad a la clase que se quiere decorar

b. Bastará con dotar al modelo de una nueva clase que generalice a la clase “decorable”

c. La clase a decorar recibirá un “hermano” capaz de navegar directamente a él

d. se incorporará una nueva clase por cada responsabilidad considerada

37.- Si se implementa el patrón *State*…

a. El objeto de la clase Contexto trasladará peticiones a un objeto de la clase Estado y éste, a su vez, le pedirá a una instancia de alguna de las clases EstadoConcreto que las ejecute

b. Todas las peticiones dirigidas al Contexto pasarán primero por el estado que lo representa

c. El Contexto ejecutará algunos pasos del algoritmo y del resto se ocuparán los objetos estado

d. El objeto Contexto delegará en su estado la ejecución de algunas operaciones

38.- ¿Se pueden mezclar patrones de diseño?

a. Se puede, pero es poco recomendable dada la complejidad que da lugar esta combinación

b. Sí, de hecho, los mejores diseños suelen integrar varios patrones que se superponen para lograr un resultado superior en cuanto a extensibilidad

c. Bajo ningún concepto, ya que cada patrón excluye al resto por ser sus competidores

d. Sí, y además es la clave para que el rendimiento del sistema sea óptimo

39.- Dadas las clases y los objetos respectivamente representados en los Diagramas 1a y 1b, la secuencia que refleja lo que sucede en caso de que la instancia de la clase *Render* le envíe un mensaje *dibujar*() al objeto d1 es la mostrada en el…

a. Diagrama 1c
b. Diagrama 1d
c. Diagrama 1e
d. Diagrama 1f

40.- Dadas las clases y los objetos de los Diagramas 2ª y 2b, si el objeto CNMV manda una petición *fijarValor*() al objeto IBEX35, se ejecutará la secuencia del…

a. Diagrama 2c
b. Diagrama 2d
c. Diagrama 2e
d. Diagrama 2