**UNIDAD 1**

1. *Enumere y explique los aspectos más relevantes que hacen que un software*

*de gran magnitud sea complejo.*

Los sistemas de software complejos tienen un ciclo de vida largo y a lo largo del

tiempo muchos usuarios llegan a depender de su funcionamiento correcto.

Resulta imposible para el desarrollador comprender todas las sutilidades de su

diseño.

La complejidad de estos sistemas excede la capacidad intelectual humana. Estos

sistemas son jerárquicos y los niveles de esta jerarquía representan diferentes

niveles de abstracción, cada uno de los cuales se construye sobre el otro, y cada

uno de los cuales es comprensible por sí mismo.

2. *¿Cuáles son los cinco atributos de un sistema complejo?*

Los atributos de los sistemas complejos son:

Tienen jerarquías: un sistema complejo se compone de subsistemas relacionados

que tienen a su vez sus propios subsistemas y así sucesivamente hasta que se

alcanza algún nivel ínfimo de componentes elementales

Componentes primitivos: La elección de que componentes de un sistema son

primitivos es relativamente arbitraria y queda en gran medida a decisión del

observador.

Los enlaces internos de los componentes suelen ser más fuerte que los enlaces

entere componentes. Esta diferencia entre interrelaciones intracomponentes e

intercomponentes proporciona una separación clara de intereses entre las

diferentes partes de un sistema, posibilitando el estudio de cada parte en forma

relativamente aislada

Los sistemas jerárquicos están compuestos usualmente de solo unas pocas clases

diferentes de subsistemas en varias combinaciones y disposiciones. Tienen

patrones comunes que pueden conllevar a la reutilización de componentes

pequeños.

Un sistema complejo que funciona ha evolucionado en un sistema simple que

funcionaba. Un sistema complejo diseñado desde cero nunca funciona y no puede

parchearse para conseguir que lo haga. Hay que volver a empezar partiendo de un

sistema simple que funcione. A medida que los sistemas evolucionan, objetos que

en su momento se consideraban complejos se convierten en primitivos sobre los

que se construyen sistemas más complejos aún.

*3.*

*¿Cuáles son las dos jerarquías más importantes que consideramos en la*

*Orientación a objetos para sistemas complejos?*

• Estructura de clases: “es un” jerarquía de generalización/especialización. Se

conoce como herencia. Define una relación entre clases, en donde una clase

comparta la estructura o comportamiento definido en una o más clases.

Herencia simple y múltiple.

• Estructura de objetos: “Parte de” jerarquía de agregación. Es el concepto

que permite el agrupamiento físico de estructuras relacionadas lógicamente.

Así un camión se compone de ruedas, motor, sistema de transmisión y

chasis. En consecuencia, camión es una agregación y ruedas es el agregado

de camión

*4.*

*¿Con qué podemos enfrentar a la complejidad para obtener partes cada vez*

*más pequeñas y simplificadas del dominio del problema?*

La técnica de dominar la complejidad se conoce como divide y vencerás.

Cuando se diseña un sistema de software complejo es esencial descomponerlo en

partes más y más pequeñas, cada una de las cuales se puede refinar entonces de

forma independiente. Para entender un nivel dado de un sistema basta con

comprender unas pocas partes a la vez.

*5. ¿Cuáles son las dos formas de descomposición más conocidas?*

Las dos formas de descomposición son la algorítmica y la orientada a

objetos.

*6.*

*¿Explique en qué se diferencia la descomposición algorítmica y la orientada a*

*objetos?*

• Descomposición algorítmica: Análisis y diseño estructurado. Se aplica

para descomponer un gran problema en pequeños problemas, la unidad

fundamental de este tipo de descomposición es el subprograma. El

programa resultante toma la forma de un árbol en el que cada

subprograma realiza su trabajo, llamando ocasionalmente a otro

subprograma

• Descomposición orientada a objetos: análisis y diseño orientado a

objetos. El mundo es visto como un conjunto de entidades autónomas.

Se descompone el sistema de acuerdo con las abstracciones clave del

dominio del problema

Diferencia: la visión algorítmica enfatiza el orden de los eventos, la visión orientada

a objetos resalta los agentes que, o bien causan acciones o bien son sujetos de

esas acciones.

*7. ¿Qué rol cumple la abstracción en la orientación a objetos?*

Cuando se adopta una visión del mundo orientada a objetos, se toman a los

objetos como abstracciones de entidades del mundo real. Representan un

agrupamiento de información particularmente denso y cohesivo.

A través de la abstracción se utilizan bloques de información de contenido

semántico cada vez mayor.

Incapaces es de dominar en su totalidad un objeto, decidimos ignorar sus

detalles no esenciales, tratando en su lugar con el modelo generalizado e idealizado

de objeto.

*8. ¿Qué rol cumple la jerarquía en la orientación a objetos?*

Otra forma de incrementar el contenido semántico de bloques de

información individuales es mediante el reconocimiento explícito de las jerarquías

de clases y objetos dentro de un sistema de software complejo.

La identificación de las jerarquías requiere que se descubran patrones entre

muchos objetos, cada uno de los cuales puede incluir algún comportamiento

tremendamente complicado. Una vez que se han expuesto estas jerarquías, la

estructura de un sistema complejo experimenta una gran simplificación.

*9.*

*¿Consideraría Ud. al diseño orientado a objetos un desarrollo evolutivo o*

*revolucionario? Justifique.*

Considero que es un desarrollo revolucionario porque cambia radicalmente la

forma de pensar un sistema.

*10.*

*¿Cuántos y cuáles son los modelos básicos que se manejan en el diseño orientad*

*o a objetos?*

El diseño orientado a objetos ofrece un rico conjunto de modelos que

reflejan la importancia de plasmar explícitamente las jerarquías de clases y de

objetos del sistema que se diseña:

Modelo dinámico

Modelo estático

Modelo lógico: estructura de clases; estructura de objetos

Modelo físico: arquitectura de módulos; arquitectura de procesos

11. *¿Qué es la programación orientada a objetos?*

La programación orientada a objetos es un método de implementación en el

que los programas se organizan como colecciones cooperativas de objetos, cada

uno de los cuales representa una instancia de alguna clase y cuyas clases son,

todas ellas, miembros de una jerarquía de clases unidas mediante relaciones de

herencia.

La programación orientada a objetos utiliza objetos, no algoritmos, como sus

bloques lógicos de construcción fundamentales (jerarquía “parte de”), cada objeto

es una instancia de alguna clase, las clases están relacionadas con otras clases por

medio de relaciones de herencia (jerarquía de clases).

Si falta alguno de estos elementos un programa no es orientado a objetos.

12. *¿Qué es el diseño orientado a objetos?*

El diseño orientado a objetos es un método de diseño que abarca el proceso

de descomposición orientada a objetos y una notación para describir los modelos

lógico y físico, así como los modelos estático y dinámico del sistema que se diseña.

Modelos lógicos (Estructura de clases y objetos)

Modelos físicos (Arquitectura de módulos y procesos)

13. *¿Qué es el análisis orientado a objetos?*

El análisis orientado a objetos es un método de análisis que examina los

requisitos desde la perspectiva de las clases y objetos que se encuentran en el

vocabulario del dominio del problema

14. *¿Cuáles son los elementos fundamentales en el modelo de objetos?*

Abstracción

Encapsulamiento

Modularidad

Jerarquía

15. *¿Cuáles son los elementos secundarios del modelo de objetos?*

Tipos

Concurrencia

Persistencia

16. *Explique el significado de la abstracción.*

Es una descripción simplificada o especificación de un sistema que enfatiza

algunos de los detalles o propiedades del mismo mientras suprime otros.

Una abstracción denota las características esenciales de un objeto que lo distingue

de todos los demás tipos de objeto y proporciona así fronteras conceptuales

nítidamente definidas respecto a la perspectiva del observador.

17. *Explique el significado del encapsulamiento.*

El encapsulamiento es el proceso de almacenar en un mismo

comportamiento los elementos de una abstracción que constituyen su estructura y

su comportamiento.

Sirve para separar la interface contractual de una abstracción y su implantación.

Se centra en la implementación que da lugar al comportamiento. Se

consigue mediante la ocultación de información, que es el proceso de ocultar todas

las propiedades de un objeto que no contribuyen a sus características esenciales. La

estructura de un objeto está oculta así como la implantación de sus métodos.

18. *Explique el significado de la modularidad***.**

La modularidad consiste en dividir un programa en módulos que pueden compilarse

por separado, pero que tienen conexiones con otros módulos.

La modularidad es la propiedad que tiene un sistema que ha sido descompuesto en

un conjunto de módulos cohesivos y débilmente acoplados.

Agrupa abstracciones relacionadas lógicamente.

19. *Explique el significado de la jerarquía.*

La jerarquía es una clasificación u ordenación de abstracciones.

Las dos jerarquías más importantes en un sistema complejo son:

Su estructura de clases (“de clases”) y su estructura de objetos (“de

partes”)

20. *Explique el significado de la tipificación.*

Un tipo es una caracterización precisa de propiedades estructurales o de

comportamiento que comparten una serie de entidades.

Los tipos son la puesta en vigor de la clase de los objetos, de modo que los

objetos de distintos tipos no pueden intercambiarse o como mucho pueden

intercambiarse solo de formas muy restringidas.

21. *Explique el significado de la concurrencia.*

La concurrencia es la propiedad que distingue un objeto activo de uno que no está

activo. Un sistema automatizado puede tener que manejar muchos eventos

diferentes simultáneamente. Otros problemas pueden implicar tantos cálculos que

excedan la capacidad de cualquier procesador individual. En ambos casos es natural

considerar el uso de un conjunto distribuido de computadoras para la implantación

que persigue utilizar procesadores de realizar multitareas.

La concurrencia permite a diferentes objetos actuar al mismo tiempo.

22. *Explique el significado de la persistencia.*

“La persistencia es la propiedad de un objeto por la que su existencia trasciende

el tiempo (es decir, el objeto continúa existiendo después de que su creador deja

de existir) y/o el espacio (es decir, la posición del objeto varía con respecto al

espacio de direcciones en el que fue creado)”. El tiempo de vida del objeto

trasciende al tiempo de vida del programa que los creó.

La persistencia conserva el estado de un objeto en el tiempo y en el espacio.

La persistencia abarca algo más que la mera duración de los datos. En las BD

orientadas a objetos, no sólo persiste el estado de un objeto, sino que su clase

debe trascender también a cualquier programa individual, de forma que todos los

programas interpreten de la misma manera el estado almacenado.

23. *¿Cómo se denotan las características esenciales de un objeto que lo*

*distinguen de todos los demás tipos de objetos y proporciona así fronteras*

*conceptuales nítidamente definidas respecto a la perspectiva del*

*observador?*

En términos generales, se define un objeto como cualquier cosa que tenga

una frontera definida con nitidez. Pero esto no es suficiente para servir de guía al distinguir un objeto de otro, ni permite juzgar la calidad de las abstracciones. Por lo

tanto surge la siguiente definición:

Un objeto tiene estado, comportamiento e identidad; la estructura y

comportamiento de objetos similares están definidos en su clase común, los

términos instancia y objeto son intercambiables. El de **estado** un objeto abarca

todas las propiedades del mismo (normalmente estáticas) más los valores actuales

(normalmente dinámicos) de cada una de esas propiedades.

Una propiedad es una característica inherente o distintiva, un rasgo o

cualidad que contribuye a hacer que un objeto sea ese objeto y no otro.

Las propiedades suelen ser estáticas, porque atributos como estos son inmutables y

fundamentales para la naturaleza del objeto.

El hecho de que todo objeto tiene estado implica que todo objeto toma cierta

cantidad de espacio.

El **comportamiento** es como actúa y reacciona un objeto en términos de sus

cambios de estado y paso de mensajes. Representa su actividad visible y

comprobable exteriormente.

24. *¿A qué denominamos un objeto cliente?*

Un cliente es cualquier objeto que utiliza los recursos de otro objeto

(denominado servidor)

25. *¿A qué denominamos un objeto servidor?*

Un servidor es el que otorga recursos al objeto sin consumirlos.

26. *¿A qué denomina Meyer el modelo contractual de programación?*

Meyer denomina modelo contractual de programación a al contrato definido

por la vista exterior de cada objeto que debe ser llevado a cabo por la vista interior

del mismo y a su vez pueden depender de otros objetos.

27. *¿Qué establece un contrato entre objetos?*

Un contrato entre objetos establece todas las suposiciones que puede

hacer un objeto cliente acerca del comportamiento de un objeto servidor. Abarca

las responsabilidades de un objeto, es decir el comportamiento del que se le

considera responsable.

28. *¿Cómo se denomina a las formas en que un objeto puede actuar y/o*

*reaccionar, constituyendo estas formas la visión externa completa, estática*

*y dinámica de la abstracción?*

A las formas en que un objeto puede actuar y/o reaccionar se lo denomina

comportamiento.

29. *¿Cómo se denomina al conjunto completo de operaciones que puede*

*realizar un cliente sobre un objeto, junto con las formas de invocación u*

*órdenes que admite?*

Al conjunto completo de operaciones que puede realizar un cliente sobre un

objeto, junto con las formas de invocación y órdenes que admite se le denomina

protocolo.

30. *¿A qué nos referimos cuando decimos que un concepto central de la idea de*

*abstracción es el de invariancia?*

Un invariante es una condición booleana (verdadera o falsa) cuyo valor de

verdad debe mantenerse. La violación de un invariante rompe el contrato asociado

con una abstracción.

31. *¿Qué se debe definir para cualquier operación asociada a un objeto?*

Para cualquier operación asociada con un objeto, es necesario definir

precondiciones (invariantes asumidos por la operación) y pos condiciones

(invariantes satisfechos por la operación)

32. *¿Qué es una precondición?*

Es una invariante asumida por una operación.

33. *¿Qué es una postcondición?*

Es una invariante satisfecha por una operación.

34. *¿A qué se denomina excepción?*

Una excepción es una indicación de que no se ha satisfecho (o no puede

satisfacerse) alguna invariante. Algunos lenguajes permiten a los objetos lanzar

excepciones, así como abandonar el procesamiento, avisando del problema a algún

otro objeto que puede a su vez aceptar la excepción y tratar dicho problema.

35. *¿A qué se denomina mensaje?*

Un mensaje es una operación que un objeto realiza sobre otro.

36. *¿El encapsulado es un concepto complementario a la abstracción?*

*Justifique.*

El encapsulado es un concepto complementario a la abstracción. Es el

proceso de almacenar en un mismo compartimento los elementos de una

abstracción que constituyen su estructura y su comportamiento. Sirve para separar

el interfaz contractual de una abstracción y su implantación.

37. *¿Cómo se denomina al elemento de un objeto que captura su vista externa*

El elemento de un objeto que captura su vista externa se llama interfaz. La

interfaz representa todo lo que un cliente necesita saber de una clase.

*38. ¿Cómo se denomina al elemento de un objeto que captura su vista interna la cu*

*al incluye los mecanismos que consiguen el comportamiento deseado?*

Se denomina implementación.

39. *¿El concepto de “ocultar los detalles de implementación” lo asociaría a*

*“esconder los detalles de implementación” o a “evitar el uso inapropiado de*

*los detalles de implementación”? Justifique.*

Dicho concepto lo asociaría a ambas frases. Lo principal es evitar el uso

inapropiado de detallas de implementación, como consecuencia de perseguir ese

objetivo se esconden los detalles de implementación.

40. *¿Cuáles son los dos aspectos que hacen importante considerar a la*

*modularidad?*

La modularidad es la propiedad que tiene un sistema que ha sido

descompuesto en un conjunto de módulos cohesivos y débilmente acoplados. Por lo

tanto los dos aspectos que hacen importante considerar la modularidad son bajo

acoplamiento y alta cohesión.

41. *¿Para qué se utiliza la jerarquía?*

Frecuentemente un conjunto de abstracciones forma una jerarquía, y la

identificación de esas jerarquías en el diseño simplifica en gran medida la

comprensión del problema.

Se define la jerarquía como una 3 de abstracciones.

Las dos jerarquías más importantes en un sistema complejo son su estructura de

clases (la jerarquía “de clases”) y su estructura de objetos (la jerarquía “de partes”)

*42.*

*¿Cómo denominamos a la caracterización precisa de propiedades estructurales*

*o de comportamiento que comparten una serie de entidades?*

Una caracterización precisa de propiedades estructurales o de

comportamiento que comparten una serie de entidades se denomina tipo. Se

utilizan los términos tipo y clase de manera intercambiable.

Los tipos son la puesta en vigor de la clase de los objetos, de modo que los objetos

de tipos distintos no pueden intercambiarse o pueden intercambiarse de forma muy

restringida.

43. *¿Las clases implementan tipos?*

Las clases contienen implementaciones de tipos

44. *¿Los tipos implementan clases?*

Los tipos no implementan clases

45. *¿Cómo denominamos a los lenguajes que hacen una comprobación de tipos*

*estricta?*

Lenguajes fuertemente tipados

*46.*

*¿Cómo denominamos a los lenguajes que no hacen una comprobación de tipos*

*estricta?*

Lenguajes débilmente tipados

47. *¿A qué se denomina ligadura estática (temprana)?*

La asignación estática de tipos, también conocida como ligadura estática o

ligadura temprana, se refiere al momento en el que los nombres se ligan con sus

tipos. La ligadura estática significa que se fijan los tipos de todas las variables y

expresiones en tiempo de compilación.

48. *¿A qué se denomina ligadura dinámica (tardía)?*

La ligadura dinámica, también llamada ligadura tardía, significa que los tipos

de las variables y expresiones no se conocen hasta el tiempo de ejecución.

49. *¿Es lo mismo la comprobación estricta de tipos y la ligadura dinámica?*

No es lo mismo, son conceptos independientes. Un lenguaje puede tener

comprobación estricta de tipos y tipos estáticos, puede tener comprobación estricta

de tipos pero soportar enlace dinámico, o no tener tipos y admitir la ligadura

dinámica.

*50.*

*¿Cómo se denomina la característica que permite a diferentes objetos actuar al*

*mismo tiempo?*

Se denomina concurrencia

51. *¿A qué se denomina concurrencia pesada?*

Un proceso pesado es aquel típicamente manejado de forma independiente

por el sistema operativo de destino y abarco su propio espacio de direcciones.

52. *¿A qué se denomina concurrencia ligera o liviana?*

Un proceso ligero suele existir dentro de un solo proceso del sistema

operativo en compañía de otros procesos ligeros, que comparten el mismo espacio

de direcciones.

*53.*

*¿La concurrencia es la propiedad que distingue un objeto activo de uno que no*

*l o está?*

Si, la concurrencia es la propiedad que distingue un objeto activo de uno que no

está activo.

*54.*

*¿Cómo se denomina la característica en orientación a objetos que permite cons*

*ervar el estado de un objeto en el tiempo y el espacio?*

Se denomina persistencia.

55. *¿Qué cosas se persisten?*

Se persisten:

- Resultados transitorios en la evaluación de expresiones

- Variables locales en la activación de procedimientos

- Variables propias, variables globales y elementos del heap cuya

duración difiere de su ámbito.

- Datos que existen entre ejecuciones de un programa

- Datos que existen entre varias versiones de un programa

- Datos que sobreviven al programa

56. *Defina qué es un objeto desde la perspectiva de la cognición humana.*

Desde la perspectiva de la cognición humana, un objeto es cualquiera de las

siguientes cosas:

- Una cosa tangible y/o visible

- Algo que puede comprenderse intelectualmente

- Algo hacia lo que se dirige un pensamiento o acción

57. *¿Un objeto es real o abstracto? Justifique.*

Un objeto representa un elemento, unidad o entidad individual e identificable, ya

sea real o abstracta, con un papel bien definido en el dominio del problema. Es

cualquier cosa real o abstracta que posee una estructura que lo define y acciones

que lo controlan.

58. *¿Los objetos poseen límites físicos precisos o imprecisos?*

Los objetos poseen limiten nítidamente definidos, es decir precisos.

59. *¿Cuáles son las tres cosas que debe tener un objeto?*

Para ser considerado objeto debe tener estado, comportamiento e identidad.

60. *¿Cuál es la palabra que se puede utilizar como sinónimo de objeto?*

Un sinónimo que se puede utilizar como objeto es instancia

61. *¿Cuál es la palabra que se puede utilizar como sinónimo de instancia?*

Objeto.

62. *¿Cómo definiría el estado de un objeto?*

Un objeto posee propiedades normalmente estáticas con valores

normalmente dinámicos, el estado de un objeto es el valor que tiene dichas

propiedades en un momento determinado

63. *¿A qué definimos propiedad de un objeto?*

Una propiedad es una característica inherente o distintiva, un rasgo o

cualidad que contribuye a hacer que un objeto sea ese objeto y no otro. Las

propiedades suelen ser estáticas, porque atributos como estos son inmutables y

fundamentales para la naturaleza del objeto.

64. *¿Qué es lo que distingue a “un objeto” de los “valores simples”?*

Las cantidades simples son temporales, inmutables y no instanciadas,

mientras que los objetos existen en el tiempo son modificables, tienen estado, son

instanciados y pueden crearse, destruirse y compartirse

65. *¿Cómo definiría el comportamiento de un objeto?*

Ningún objeto existe de forma aislada. En vez de eso, los objetos reciben

acciones y ellos mismos actúan sobre otros objetos. Así puede decirse que el

comportamiento es cómo actúa y reacciona un objeto en términos de sus cambios

de estado y paso de mensajes.

*66.*

*¿El comportamiento de un objeto se ve afectado por el estado del mismo o bien*

*que el comportamiento del objeto es función de su estado?*

El estado de un objeto representa los resultados acumulados de su

comportamiento.

El comportamiento de un objeto es función de su estado así como de la operación

que se realiza sobre él, teniendo algunas operaciones efecto lateral de modificar el

estado del objeto, entonces el estado de un objeto representa los resultados

acumulados de su comportamiento.

*67. ¿Algunos comportamientos pueden alterar el estado de un objeto?*

Si

*68.*

*Se puede afirmar que el estado de un objeto termina siendo los resultados acu*

*mulados de su comportamiento.*

Si, el estado de un objeto son los resultados acumulados de su

comportamiento.

69. *¿A qué definiría como operación (método/función miembro)?*

Una operación es una acción que un objeto efectúa sobre otro con el fin de

provocar una reacción. Las operaciones que los clientes pueden realizar sobre un

objeto suelen declararse como métodos o función miembro.

70. *¿Cuáles son las tres operaciones más comunes?*

Una operación denota un servicio que una clase ofrece a sus clientes. Los

tres tipos más comunes de operaciones son los siguientes:

• Modificador: Una operación que altera el estado de un objeto

• Selector: Una operación que accede al estado de un objeto pero no

altera ese estado

• Iterador: Una operación que permite acceder a todas las partes de un

objeto en algún orden perfectamente establecido.

*71.*

*¿Cuáles son las dos operaciones habituales que se utilizan para crear y destruir*

*instancias de clases?*

• Constructor: Una operación que crea un objeto y/o inicializa su estado

• Destructor: Una operación que libera el estado de un objeto y/o destruye

el propio objeto.

72. *¿Qué tipo de operación es el modificador?*

Una operación que altera el estado de un objeto

73. *¿Qué tipo de operación es el selector?*

Una operación que accede al estado de un objeto pero no altera ese estado

74. *¿Qué tipo de operación es el iterador?*

Una operación que permite acceder a todas las partes de un objeto en algún

orden perfectamente establecido.

75. *¿Qué tipo de operación es el constructor?*

Una operación que crea un objeto y/o inicializa su estado

76. *¿Qué tipo de operación es el destructor?*

Una operación que libera el estado de un objeto y/o destruye el propio

objeto.

*77.*

*¿Cómo denominamos operaciones fuera de las clases en aquellos programas*

*ori entados a objetos que permiten colocarlas (ej. C++)?*

C++ permite escribir operaciones como subprogramas libres, llamados función no

miembro. Los subprogramas libres son procedimientos o funciones que sirven

como operaciones no primitivas sobre un objeto u objetos de la misma o de

distintas clases.

78. *¿Todos los métodos son operaciones?*

Sí, todos los métodos son operaciones.

79. *¿Todas las operaciones son métodos?*

No. No todas las operaciones son métodos: algunas operaciones pueden

expresarse como subprogramas libres.

80. *Dado el protocolo de un objeto (todos sus métodos y subprogramas libres*

*asociados al objeto si el lenguaje lo permite) es conveniente dividirlo en*

*grupos lógicos más pequeños de comportamiento? ¿Por qué?*

Para la mayoría de las abstracciones no triviales, es útil dividir este protocolo mayor

en grupos lógicos de comportamiento. Estas colecciones, que constituyen una

partición del espacio de comportamiento de un objeto, denotan los papeles que un

objeto puede desempeñar y se definen los contratos entre las abstracciones y sus

clientes.

81. *¿Cómo denominamos a los grupos lógicos más pequeños de*

*comportamiento del protocolo total de un objeto?*

Papeles y responsabilidades

82. *¿Cuáles son las dos responsabilidades más importantes que posee un*

*objeto?*

Las dos responsabilidades más importantes son: el conocimiento que un

objeto mantiene y las acciones que puede llevar a cabo.

83. *¿Es relevante el orden en que se invocan las operaciones de un objeto?*

La existencia de un estado en el seno de un objeto significa que el orden en el que

invocan las operaciones es importante. Para algunos objetos, esta ordenación de

las operaciones respecto a los eventos y al tiempo es tan penetrante que se puede

caracterizar mejor formalmente el comportamiento de tales objetos en términos de

una máquina de estado finitos equivalente.

84. *¿Por qué decimos que los objetos se pueden considerar como máquinas?*

Las instancias de clases, actúan como pequeñas máquinas y, ya que todas esas

instancias incorporan el mismo comportamiento, se puede utilizar la clase para

capturar esta semántica común de orden respecto al tiempo y los eventos.

85. *¿Qué es la identidad de un objeto?*

La identidad es aquella propiedad de un objeto que lo distingue de todos los

demás.

86. *Dadas dos variable X e Y del mismo tipo ¿qué significa que ambas son*

*iguales?*

Que ambas apuntan a la misma dirección de memoria

87. *Dadas dos variable X e Y del mismo tipo ¿qué significa asignarle Y a X?*

Asignarle la dirección de memoria que apunta Y a X

88. *Dadas dos variable X e Y del mismo tipo ¿qué significa clonar X en Y?*

Copiar los valores de X en Y

89. *¿Qué significa realizar una clonación superficial?*

Significa que se copia el objeto pero no se comparte su estado

90. *¿Qué significa realizar una clonación profunda?*

Significa que se copia el objeto y su estado

91. *¿Qué es el ciclo de vida de un objeto?*

El ciclo de vida de un objeto es el tiempo de vida, el tiempo de vida de un

objeto se extiende desde el momento que se crea por primera vez (y consume asi

espacio) hasta que ese espacio se recupera

92. *¿Cómo se libera el espacio ocupado por un objeto?*

Los objetos creados en el heap con el operador new deben ser destruidos

explícitamente con el operador GC.Collect.

93. *¿Qué tipos de relaciones existen entre los objetos?*

Existen dos tipos de relaciones:

• Enlaces

• Agregación

94. *¿Cómo podemos definir al enlace entre objetos?*

Enlace se define como una conexión física o conceptual entre objetos. Un

objeto colabora con otros objetos a través de sus enlaces con estos. Un enlace

denota la asociación especifica por la cual un objeto (cliente) utiliza los servicios de

otro objeto (servidor), o a través del cual un objeto puede comunicarse con otro.

95. *Cómo pueden ser los mensajes entre dos objetos en una relación de enlace*

El paso de mensajes entre dos objetos es típicamente unidireccional aunque

ocasionalmente puede ser bidireccional

96. *¿Qué es un mensaje unidireccional?*

Un objeto cliente invoca a un servidor

97. *¿Qué es un mensaje bidireccional?*

Los datos pueden fluir en ambas direcciones

98. *¿Quién inicia el paso de un mensaje entre dos objetos en una relación de*

*enlace?*

El paso de mensajes es iniciado por el cliente, dirigido hacia el servidor

*99.*

*¿Cuáles son los roles o papeles que puede desempeñar un objeto en una relació*

*n de enlace?*

Los roles que puede desempeñar un objeto en una relación de enlace son:

Actor, Servidor, Agente

100. *¿Qué significa que un objeto actúe como “Actor”?*

Un objeto que puede operar sobre otros objetos pero nunca se opera sobre

él por parte de otros objetos, en algunos contextos, el término objeto activo y actor

son equivalentes.

101. *¿Qué significa que un objeto actúe como “Servidor”?*

Un objeto que nunca opera sobre otro objetos, los otros objetos operan sobre el

102. *¿Qué significa que un objeto actúe como “Agente”?*

Un objeto que puede operar sobre otros objetos y además otros objetos

pueden operar sobre él. Un agente se crea normalmente para realizar algún trabajo

en nombre de un actor y otro agente

103. *Dados dos objetos A y B, si A le puede enviar un mensaje a B, estando*

*ambos relacionados por enlace, decimos que B respecto de A está:*

Visible  
*104.*

*¿Cuáles son las cuatro formas de visibilidad que puede poseer un objeto servido*

*r respecto de un objeto cliente?*

• El objeto servidor es global para el cliente

• El objeto servidor es un parámetro de alguna operación del cliente

• El objeto servidor es parte del objeto cliente

• El objeto servidor es un objeto declarado localmente en alguna

operación del cliente.

*105.*

*En una relación de enlace de dos objetos, cuando uno le pasa un mensaje al*

*otro , además de adoptar roles ambos deben estar:*

Sincronizados.

*106. ¿Cuáles son las posibles formas de sincronización?*

• Secuencial: La semántica del objeto pasivo está garantizada solo en

presencia de un único objeto activo simultáneamente

• Vigilado: La semántica del objeto pasivo está garantizada en presencia

de múltiples hilos de control, pero los clientes activos deben colaborar

para lograr la exclusión mutua.

• Síncrono: La semántica del objeto pasivo está garantizada en presencia

de múltiples hilos de control y el servidor garantiza la exclusión mutua.

*107. ¿Qué significa que dados dos objetos A y B estos están*

*secuencialmente sincronizados?*

El funcionamiento del objeto pasivo está garantizado por el accionar de un único

objeto activo simultáneamente. Hay un solo hilo de ejecución o de atención (uno

solo por vez)

*108.*

*¿Qué significa que la forma de sincronizarse de un conjunto de objetos es vigila*

*da?*

La semántica del objeto pasivo está garantizada en presencia de múltiples

hilos de control, pero los clientes activos deben colaborar para lograr la exclusión

mutua.

*109.*

*¿Qué significa que la forma de sincronizarse de un conjunto de objetos es*

*síncrona?*

La semántica del objeto pasivo está garantizada en presencia de múltiples

hilos de control y el servidor garantiza la exclusión mutua.

110. *¿El enlace es una relación de igual a igual o jerárquica?*

Los enlaces denotan relaciones de igual a igual o cliente servidor.

111. *¿La agregación es una relación de igual a igual o jerárquica?*

La agregación denota una jerarquía todo/parte, con la capacidad de ir desde

el todo (también llamado el agregado) hasta sus partes (conocidas también como

atributos)

112. *¿Qué tipo de jerarquía denota la agregación?*

Todo/parte

113. *¿Qué otros nombres recibe el “todo” en una relación de agregación?*

Agregado

114. *¿En una relación de agregación las “partes” forman parte del estado del*

*“todo”?*

Si, son los atributos

115. *¿Qué tipos de agregación existen?*

• Con contención física: Cuando un objeto no existe sin el otro. Sus ciclos de

vida están íntimamente relacionados. Por ejemplo, un avión se compone de

alas, motores, etc (es un caso de contención física).

• Sin contención física: El ciclo de vida de los elementos son independientes

uno del otro. Ejemplo: la relación entre un accionista y sus acciones es una

relación de agregación que no requiere contención física. El accionista

únicamente posee acciones, pero las mismas no son de ninguna manera

parte física del accionista.

116. *¿Qué caracteriza a la agregación con contención física?*

Cuando un objeto no existe sin el otro. Sus ciclos de vida están íntimamente

relacionados. Por ejemplo un avión se compone de alas, motores, etc (es un caso

de contención física)

117. *¿Qué es una clase?*

Una clase representa solo una abstracción, la esencia de un objeto. Se

puede definir una clase como un grupo conjunto o tipo marcado por atributos

comunes o un atributo común; una división, distinción o clasificación de grupos

basada en la calidad, grado de competencia o condición.

118. *¿La interfaz de la clase proporciona su visión interna?*

No esto lo proporciona la implementación, la implementación de una clase es

su visión interna, que engloba los secretos de su comportamiento. La

implementación de una clase se compone principalmente de la implementación de

todas las operaciones definidas en la interfaz de esta.

119. *¿La implementación de la clase proporciona su visión externa?*

No, esto lo proporciona la interfaz, la interfaz de una clase proporciona su visión

externa y por lo tanto enfatiza la abstracción a la vez que oculta su estructura y los

secretos de su comportamiento. Este interfaz se compone principalmente de las

declaraciones de todas las operaciones aplicables a instancias de esta clase, pero

también puede incluir la declaración de otras clases, constantes, variables y

excepciones, según se necesiten para completar la abstracción.

*120.*

*¿En cuántas partes la podemos dividir una interfaz en términos de la*

*accesibilidad o visibilidad que posee?*

Se puede dividir en 4 partes:

* Public: una declaración accesible a todos los clientes.
* Protected: una declaración accesible sólo a la propia clase, sus subclases, y

sus clases amigas.

* Internal: Solo puede accederse dentro del proyecto
* Private: una declaración accesible sólo a la propia clase y sus clases amigas.

121. *¿Qué tipos básicos de relaciones existen entre las clases?*

Existen tres tipos básicos de relaciones entre clases:

• Generalización/especialización: denota una relación es un. Por ejemplo,

una rosa es un tipo de flor, lo que quiere decir que una rosa es una

subclase especializada de una clase más general, la de las flores.

• Todo/Parte: Denota una relación parte de, así un pétalo no es un tipo de

flor, es parte de una flor.

• Asociación: Denota alguna dependencia semántica entre clases de otro

modo independientes. Por ejemplo, las rosas y las velas son clases

claramente independientes pero ambas representan cosas que podrían

utilizarse para decorar la mesa de una cena.

122. *¿Qué relaciones entre clases se desprenden de las tres relaciones básicas?*

• Herencia,

• Agregación

• Uso

• Instanciación

123. *¿La asociación denota una dependencia semántica y la dirección de esta*

*asociación?*

Una asociación solo denota una dependencia semántica y no establece la

dirección de esta dependencia ni establece la forma exacta en que una clase se

relaciona con otra.

La asociación puede ser bidireccional

124. *¿Qué significa la cardinalidad en una relación?*

Es la multiplicidad con que se relaciona una clase con otra.

125. *¿Qué cardinalidad puede existir entre clases relacionadas por asociación?*

• Uno a uno

• Uno a muchos

• Muchos a muchos

126. *¿Qué es la herencia?*

La herencia es una relación entre clases en la que una clase comparte la

estructura y/o el comportamiento definidos en una o más clases. La clase de la que las otras heredan se denomina superclase. La clase que hereda de otra se

denomina subclase.

127. *¿Cuántos tipos de herencia existen?*

Existen dos tipos:

• Herencia Simple

• Herencia múltiple

128. *¿A qué se denomina herencia simple?*

Una clase hereda parte de la estructura y/o comportamiento de otra clase

129. *¿A qué se denomina herencia múltiple?*

Una clase hereda parte de la estructura y/o comportamiento de más de una

clase

130. *¿Cómo se denomina a la clase que no se espera tener instancias de ella y*

*solo se utilizará para heredar a otras clases?*

Se denomina clase abstracta.

*131.¿Cómo se denomina a la clase que se espera tener instancias de ella y pued*

*e utilizarse para heredar a otras clases o no?*

Se denomina clase concreta.

132. *¿Cómo se denomina al método de una clase abstracta que no posee*

*implementación y fue pensado para que sea implementado en las sub*

*clases que lo heredan?*

Método virtual.

133. *¿Cómo se denomina a la clase más generalizada en una estructura de*

*clases?*

superclase o clase base.

134. *¿Qué es el polimorfismo?*

El polimorfismo es un concepto de teoría de tipos en el que un nombre

puede denotar instancias de muchas clases diferentes en tanto y en cuanto estén

relacionadas por alguna superclase común. Cualquier objeto denotado por este

nombre es, por lo tanto capaz de responder a algún conjunto común de

operaciones de diversas formas.

135. *¿Cómo se denomina cuando una clase posee métodos que comparten el*

*nombre y se diferencias por su firma?*

Se denomina sobrecarga de métodos.

136. *¿Qué sentencias de código se evitan utilizar cuando se aplica*

*correctamente el polimorfismo?*

Se evita el uso de if anidados, que generan un código altamente acoplado.

137. *¿Qué es la agregación cómo relación entre clases?*

Las relaciones de agregación entre clases tienen un paralelismo directo con

las relaciones de agregación entre los objetos correspondientes a esas clases.

Relación Jerárquica del tipo Todo / Parte. La agregación es un caso particular de la

asociación que denota posesión.

138. *¿Qué formas de contención física existen en la agregación?*

• Con contención Física : se subdivide en valor y referencia

• Sin contención Física: el ciclo de vida de los elementos son independientes

uno del otro

139. *¿Qué características posee la contención física por valor?*

Contención física por valor: es un tipo de contención que significa que el

objeto contenido no existe independientemente de la instancia contenedora que lo

encierra. Por el contrario, el tiempo de vida de ambos objetos está en intima

conexión. Cuando se crea una instancia del contenedor se crea una del contenido.

Cuando se destruye el contenedor, se destruye el contenido.

140. *¿Qué características posee la contención física por referencia?*

Los tiempos de vida de ambos objetos ya no están tan estrechamente

emparejados como antes. Se pude crear y destruir instancias de cada clase

independientemente.

141. *¿Qué es una relación de uso?*

Una asociación denota una conexión semántica bidireccional, una relación de

uso es un posible refinamiento de una asociación, por el que se establece qué

abstracción es el cliente y qué abstracción es el servidor que proporciona ciertos

servicios. Es una asociación refinada, se especifican roles (actor, cliente, servidor)

142. *¿Qué es la instanciación?*

La instanciación es la creación de instancias, es la acción por la cual se

somete a una clase para obtener un objeto

143. *¿Todo objeto es una instancia de una clase?*

Sí, todo objeto es una instancia de una clase.

144. *¿Qué es una metaclase?*

Una metaclase es una clase cuyas instancias son clases.

145. *¿Qué métricas hay que observar para determinar la calidad de una*

*abstracción?*

• Bajo acoplamiento

• Alta cohesión

• Suficiencia

• Compleción

• Ser primitivo

146. *¿Qué es el acoplamiento?*

La medida de la fuerza de la asociación establecida por una conexión entre

un módulo y otro. El acoplamiento fuerte complica un sistema porque los módulos

son más difíciles de comprender, cambiar o corregir por si mismos si están muy

interrelacionados con otros módulos. La complejidad puede reducirse diseñando

sistemas con los acoplamientos más débiles posibles entre los módulos.

147. *¿Qué es la cohesión?*

La cohesión mide el grado de conectividad entre los elementos de un solo

modulo (y para el diseño orientado a objetos, una clase u objeto). La forma de

cohesión menos deseable es la cohesión por coincidencia, en la que se incluyen en

la misma clase o modulo abstracciones sin ninguna relación. La forma más deseable

de cohesión es la cohesión funcional, en la cual los elementos de una clase o

modulo trabajan todos juntos para proporcionar algún comportamiento bien

delimitado.

148. *¿Qué es la suficiencia?*

La clase o módulo captura suficientes características de la abstracción como

para permitir una interacción significativa y eficiente. Lo contrario produce

componentes inútiles. En otras palabras es el grado en que una clase representa

elementos suficientes y necesarios para que sea una implementación eficiente.

149. *¿Qué es la compleción?*

Por completo, quiere decirse que la interfaz de la clase o módulos captura

todas las características significativas de la abstracción. Mientas la suficiencia

implica una interfaz mínima, una interfaz completa es aquella que cubre todos los

aspectos de la abstracción. Una clase o módulo completo es aquel cuya interfaz es

suficientemente general para ser utilizable de forma común por cualquier cliente.

Según apuntes de clase: es el grado en que la clase contempla características

significativas de la abstracción.

150. *¿Qué significa ser primitivo?*

Las operaciones primitivas son aquellas que pueden implantarse

eficientemente sólo si tienen acceso a la representación subyacente de la

abstracción. Apunte de clase: lo más básica posible para que sea reutilizable. Todo

esto depende del contexto.

151. *¿Qué se debe observar al momento de decidir si una abstracción debe*

*implementar un determinado comportamiento o no?*

• Reutilización (Reusabilidad): ¿sería este comportamiento más útil en más

de un contexto?

• Complejidad: ¿qué grado de dificultad plantea implementar este

comportamiento?

• Aplicabilidad: ¿qué relevancia tiene este comportamiento para el tipo en

el que podría ubicarse?

• Conocimiento de la implementación: ¿depende del comportamiento de

los detalles internos?

152. *¿Qué formas puede adoptar el paso de un mensaje?*

• Síncrono

• Abandono inmediato

• De intervalo

• Asíncrono

153. *¿Qué características posee un mensaje síncrono?*

Una operación comienza sólo cuando el emisor ha iniciado la acción y el

receptor está preparado para aceptar el mensaje, el emisor y el receptor esperarán

indefinidamente hasta que ambas partes estén preparadas para continuar.

154. *¿Qué características posee un mensaje de abandono inmediato?*

Igual que el síncrono, excepto en que el emisor abandonará la operación si

el receptor no está preparado inmediatamente.

155. *¿Qué características posee un mensaje de intervalo?*

Igual que el síncrono excepto en que el emisor esperará a que el receptor

esté listo solo durante un intervalo de tiempo especificado.

156. *¿Qué características posee un mensaje Asíncrono?*

Un emisor puede iniciar una acción independientemente de si el receptor

está esperando o no el mensaje.

157. *¿Qué significa que una abstracción está accesible a otra?*

Por accesible se entienda la capacidad de una abstracción para ver a otra y

hacer referencia a recursos de su vista externa. Una abstracción es accesible a otro

sólo donde sus ámbitos se superpongan y solo donde estén garantizados los

derechos de acceso.

158. *¿Qué expresa la ley de Demeter?*

La ley Demeter afirma que los métodos de una clase no deberían depender

de ninguna manera de la estructura de ninguna clase, salvo de la estructura

inmediata (nivel superior) de su propia clase. Además, cada método debería enviar

mensajes solo a objetos pertenecientes a un conjunto muy limitado de clases.

159. *¿Cuál es la consecuencia inmediata de aplicar la ley de Demeter?*

El efecto básico de la aplicación de esta ley es la creación de clases

débilmente acopladas, cuyos secretos de implantación están encapsulados. Tales

clases están claramente libres de sobrecargas, lo que significa que para

comprender el significado de una clase no es necesario comprender los detalles de

muchas otras clases.

160. *¿Cuáles son las cuatro formas fundamentales por las cuales un objeto X*

*puede hacerse visible a un objeto Y?*

• El objeto proveedor es global al cliente.

• El objeto proveedor es parámetro de alguna operación del cliente.

• El objeto proveedor es parte del objeto cliente.

• El objeto proveedor es un objeto declarado localmente en el ámbito del

diagrama de objetos.

161. *¿Para qué sirve clasificar a los objetos?*

La clasificación ayuda a identificar jerarquías de generalización,

especialización y agregación entre clases. Reconociendo los patrones comunes de

interacción entre objetos, se llega a idear mecanismos que sirven como alma de la

implantación. La clasificación también proporciona una guía para tomar decisiones

sobre modularización.

162. *¿Por qué es tan difícil la clasificación de objetos?*

Porque cualquier clasificación es relativa a la perspectiva del observador que

la realiza (hay tantas formas de dividir el mundo en sistemas de objetos como

científicos para emprender esa tarea) y la clasificación inteligente requiere una gran

cantidad de imaginación creativa.

163. *¿Cómo es el rol del observador en la clasificación de objetos?*

El rol es muy importante. El observador debe asimilar la información

derivada de sus observaciones y después sacar conclusiones acerca de sus

construcciones hipotéticas. Puede hacer inferencias por completo erróneas. Por el

contrario, si el observador es por completo objetivo y no conoce el tema de la

observación puede que lo observado no sea lo adecuado. La observación exige un

conocimiento competente de lo observado y de su significado.

164. *¿Cuáles son las aproximaciones generales a la clasificación?*

• Categorización Clásica

• Agrupamiento Conceptual

• Teoría de prototipos

165. *¿Qué es la categorización clásica?*

Todas las entidades que tienen una determinada propiedad o colección de

propiedades en común forman una categoría. Tales propiedades son necesarias y

suficientes para definir la categoría.

166. *¿Qué es el agrupamiento conceptual?*

En este enfoque las clases (agrupaciones de entidades), se generan

formulando primero descripciones conceptuales de estas clases y clasificando

entonces las entidades de acuerdo con las descripciones.

167. *¿Qué es la teoría de prototipos?*

Se crean clases prototípicas. A todas aquellas que se le aproximan en forma

significativa se las considera pertenecientes a ese tipo. Parte de escoger un objeto

prototipo que representa a una clase de objetos, y considerar a otros objetos como

miembros de la clase si y sólo si se parecen de modo significativo al prototipo.

168. *¿Qué es una abstracción clave?*

Es una clase u objeto que forma parte del vocabulario del dominio del problema.

El valor principal que tiene la identificación de tales abstracciones es que dan unos

límites al problema; enfatizan las cosas que están en el sistema y, por tanto, son

relevantes para el diseño, y suprimen las cosas que están fuera del sistema que son

superfluas. La identificación de abstracciones clave es altamente específica de cada

dominio.

Su identificación requiere descubrimiento e invención. Esta identificación inicial

da lugar a abstracciones que, en muchos casos, resultan erróneas. Requieren un

proceso de refinado y estructuración jerárquica. Se aconseja seguir las reglas de

notación:

• Los objetos deben denominarse con nombres propios, mientras las

clases tienen nombres comunes.

• Operaciones modificadoras deben denominarse con verbos activos,

mientras las operaciones de selección usan preguntas o nombres con el

verbo ser.

169. *¿Qué son los mecanismos?*

Mecanismo es cualquier estructura mediante la cual los objetos colaboran

para proporcionar algún comportamiento que satisficiese un requerimiento del

problema. Mientras el diseño de una clase incorpora el conocimiento de cómo se

comportan los objetos individuales, un mecanismo es una decisión de diseño sobre

como cooperan colecciones de objetos. Los mecanismos representan así patrones

de comportamiento.

170. *¿Qué es un Campo de una clase?*

El campo de una clase es una propiedad o atributo

171. *¿Qué es un método de una clase?*

Es un procedimiento o función que modifica el estado de una clase

172. *¿A qué denominamos sobrecarga?*

Sobrecargar un método significa que podrá proporcionar varios métodos con

el mismo nombre pero con diferentes firmas de parámetros (Es decir, con un

numero distinto de parámetros o con parámetros de diferentes tipos)

173. *¿Qué es una propiedad de una clase?*

Las propiedades de una clase son elementos que permiten acceder a

atributos de la clase a través de métodos get y set.

174. *¿Qué tipos de propiedades existen?*

• Solo Lectura

• Solo Escritura

• Lectura/Escritura

• Propiedades con Argumentos

175. *¿Qué ámbitos pueden tener los campos, métodos y propiedades de las*

*clases?*

• Private

• Protected

• Internal

• Public

*176.¿Qué características posee cada ámbito existente si se lo aplico a un*

*campo, un método y una propiedad de una clase?*

• **Private:** Un miembro declarado como Privado, sólo es accesible en la clase

en la que se ha declarado.

• **Protected:** Un miembro declarado como Protected es accesible en la clase

en la que se ha declarado y en las que deriven de ella.

• **Internal:** Serán accesibles desde el ensamblado en el cual están declarados

y tampoco se pueden utilizar en el interior de una función.

• **Public:** Visible por todo el mundo

177. *¿Para qué se utilizan los constructores?*

Es un método que se ejecutan cuando se crea una nueva instancia de la

clase. En dicho método se inicializa el estado de un objeto y se crea la contención

física para el mismo.

178. *¿A qué se denomina tiempo de vida de un objeto?*

El tiempo de vida es el ciclo durante el cual el objeto vive en el programa,

abarca la creación modificación y destrucción del mismo.

179. *¿Para administrar las instancias de .NET se utiliza un contador de*

*referencias?*

No se utiliza un contador de referencias.

180. *¿Qué objeto es el encargado de liberar el espacio ocupado por objetos que*

*ya no se utilizan?*

El encargado de liberar el espacio ocupado por objetos que ya no se utilizan

es el garbage collector.

181. *¿Qué son los sucesos?*

Los eventos o sucesos son el modo que tiene una clase de proporcionar

notificaciones a sus clientes cuando ocurre una determinada accion.

182. *¿Qué se utiliza para declarar un suceso?*

Se utiliza la palabra reservada event.

183. *¿Cómo se logra que ocurra un suceso?*

184. *¿Cómo se pueden atrapar los sucesos?*

Con EventHandler

185. *¿Para qué se utiliza Addhandler en un suceso?*

Se utiliza para iniciar eventos en tiempo de ejecución.

186. *¿Cómo y que cosas se pueden compartir en una clase?*

Campos, métodos y sucesos.

187. *¿Qué característica poseen los campos compartidos?*

188. *Que características poseen los métodos compartidos?*

189. *¿Qué características poseen los sucesos compartidos?*

190. *¿Qué son y para que se pueden utilizar las clases anidadas?*

Las clases anidadas son clases dentro de otras. La característica principal de

las clases anidadas es que pueden acceder a miembros privados de la clase externa

mientras tienen todo el poder de una clase.

191. *¿Qué ámbitos/modificadores de acceso existen? Explique las*

*características de cada uno.*

Los ámbitos que existen en C# son los siguientes:

• Public: Los elementos declarados serán accesibles desde cualquier

porción de código del proyecto y desde cualquier otro proyecto que

haga referencia a aquel donde están declarados.

• Protected: Se puede utilizar en el interior de una clase. Permite

restringir el acceso a la variable únicamente al código de la clase y

todas las clases que hereden de ella.

• Private: Restringe el acceso a la variable al módulo, a la clase o a la

estructura en la cual está declarada. No se puede utilizar en el

interior de un procedimiento o función.

192. *¿Qué cosas se pueden heredar?*

Se pueden heredar todos los componentes de una clase, menos los

constructores estáticos, constructores de instancia y destructores.

193. *¿Cómo y para que se puede aprovechar en la practica el polimorfismo?*

Se utiliza para reimplementar métodos en clases hijas y así poder extender

sus funciones. Se utiliza la palabra overide para reimplementar métodos virtuales y

abstractos.

194. *¿Cómo y para que se utiliza la clase derived?*

Se denomina clase derivada a la clase que hereda los miembros de la clase

base.

195. *¿Qué representa this?*

Sirve para referenciar la instancia actual de la clase donde se invoca.

196. *¿Qué clase representa a la clase base?*

Representa la superclase de la cual hereda la clase en la que se está

Posicionado

197. *¿Para qué se usa una clase abstracta?*

Es una clase que puede ser heredada pero no instanciada. Se usa para

agrupar atributos y métodos que queramos tener en determinadas clases hijas pero

que no sean instanciados.

198. *¿Para que se usa una clase sellada?*

Se usan cuando queremos generar clases que no tengan ningún tipo de

herencia. Es un nivel superior de encapsulamiento.

199. *¿Qué es la sobreescritura?*

La sobreescritura es un mecanismo que usamos para sobreescribir métodos

de la clase que estamos heredando, para modificar su uso internamente en otra

clase.

200. *¿Qué elementos se pueden sobreescribir?*

Los métodos.

201. *¿A que se denomina sombreado de metodos?*

El sombreado de métodos se usa para ocultar métodos de una clase base.

202. *¿Qué característica peculiar posee el sombreado vs la sobre-escritura?*