Apunte POO

INTRODUCCIÓN A LA O.O:
Programación Imperativa: los algoritmos se describen en base a procesos.
Programación modular: el problema se divide en módulos autónomos que se pueden programar, verificar y modificar individualmente.

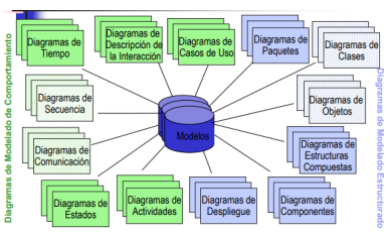
Tipo de Datos Abstracto:

1. Un tipo de datos definido por el programador
2. Un conjunto de operaciones abstractas sobre objetos de ese tipo
3. Encapsulamiento de los objetos de ese tipo, de tal manera que el usuario final del tipo no pueda manipular esos objetos excepto a través del uso de las operaciones definidas.

Características del Diseño O.O:

- La programación orientada a objetos encara la resolución de cada problema desde la óptica del objeto.
- El objeto combina los datos con los procedimientos u operaciones que actúan sobre dichos datos.
- Los objetos interactúan entre sí enviando mensajes
- Los métodos son muy similares a los procedimientos de programación imperativa tradicional y los mensajes se podrían pensar como invocaciones a esos procedimientos.
- El programador orientado a objetos puede agrupar características comunes de un conjunto de objetos en clases, a través de un proceso de abstracción.
- Los descendientes de estas clases se construyen por medio del mecanismo de subclasificación, permitiendo que sean heredados los métodos programados anteriormente y debiendo programar solamente las diferencias.
- La Programación Orientada a Objetos no se puede desligar de todo el paradigma de orientación a objetos.
- El principio fundamental del paradigma de programación orientada a objetos es construir un sistema de software en base a las entidades de un modelo elaborado a partir de un proceso de abstracción y clasificación.
- Para hacer buena POO hay que desarrollar todo el sistema utilizando el paradigma empezando por un análisis y un diseño orientados a objetos.
- En general, el desarrollo de software implica, desde una visión orientada a objetos tiene una serie de etapas para hacer: requerimientos, análisis, diseño, implementación, prueba.

UML



Es un lenguaje que permite la visualización, especificación y documentación de sistemas orientados a objetos, no es una metodología sino una notación, que aglutina distintos enfoques de orientación a objetos, UML 2 define trece diagramas para describir distintas perspectivas del sistema.

ELEMENTOS BÁSICOS DE POO:
Objetos: Un objeto es una unidad atómica que encapsula estado y comportamientom, la encapsulación en un objeto permite una alta cohesión y un bajo acoplamiento.
Los objetos son entidades que tienen atributos y comportamiento particular.
Atributos de un objeto: Los atributos describen la abstracción de características individuales que posee un objeto.
Comportamientos: Los comportamientos de un objeto representan las operaciones que pueden ser realizadas por un objeto.

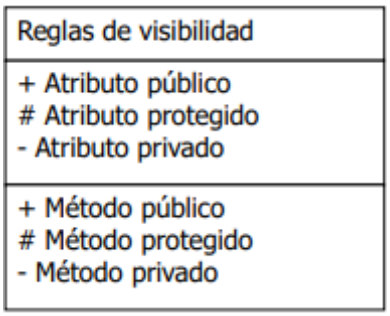
Objeto = Estado + Comportamiento + Identidad

- El estado agrupa los valores instantáneos de todos los atributos de un objeto, evoluciona con el tiempo.
- El comportamiento describe las acciones y reacciones de ese objeto.
- Las acciones u operaciones de un objeto se desencadenan como consecuencia de un estímulo externo, representado en forma de un mensaje enviado por otro objeto. El estado y el comportamiento están relacionados.
- La identidad permite distinguir los objetos de forma no ambigua,independientemente de su estado. Esto permite distinguir dos objetos en los que todos los valores de los atributos son idénticos.

CLASES:
Una clase abstrae las características de un conjunto de objetos con comportamientos similares.
La encapsulación de una clase permite la cohesión y presenta distintas ventajas básicas:
- Se protegen los datos de accesos indebidos
- El acoplamiento entre las clases se disminuye
- Favorece la modularidad y el mantenimiento

Una clase es una descripción de un conjunto de objetos, ya que consta de comportamientos y atributos que resumen las características comunes del conjunto.
La posibilidad de definir clases es una de las ventajas de la orientación a objetos; definir clases significa colocar código reutilizable en un depósito común en lugar de redefinirlo cada vez que se necesite, cada objeto es instancia de una clase.

- visibilidad
Los atributos de una clase no deberían ser manipulables directamente por el resto de objetos, no obstante existen distintos niveles de encapsulación también llamados niveles de visibilidad.



-métodos y mensajes
Los objetos tienen la posibilidad de actuar, la acción sucede cuando un objeto recibe un mensaje, que es una solicitud que pide al objeto que se comporte de manera determinada.

- Cada objeto recibe, interpreta y responde a mensajes enviados por otros objetos.
- Los comportamientos u operaciones que caracterizan un conjunto de objetos residen en la clase y se llaman métodos.
- Los métodos son el código que se ejecuta para responder a un mensaje, y el mensaje es la llamada o invocación a un método.

-variables de instancia
Las variables de instancia o miembros dato se usan para guardar los atributos de un objeto particular.

-variables de clase
son aquellos atributos que tienen el mismo valor para cada objeto de la clase. Representa un área de memoria compartida por todos los objetos de la clase.

HERENCIA:
Generalización: consiste en factorizar los elementos comunes (atributos, métodos y restricciones) de un conjunto de clases en una clase más general llamada súper clase.

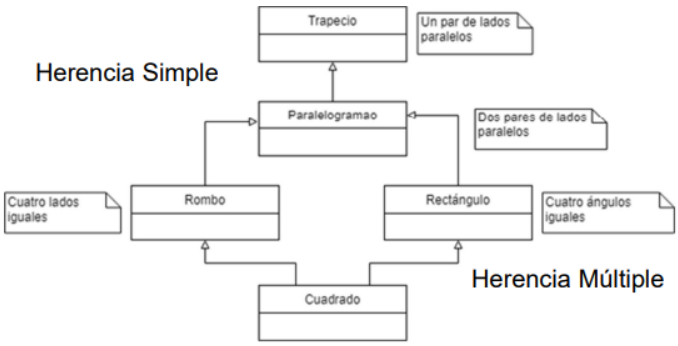
Especialización: permite capturar las particularidades de un conjunto de objetos no discriminadas por las clases ya identificadas.

- Las clases se ordenan según una jerarquía, una súper clase es una abstracción de sus subclases.
- Los árboles de clases se determinan partiendo de las hojas mientras que los niveles superiores son abstracciones construidas para ordenar y comprender.
- Una subclase identifica el comportamiento de un conjunto de objetos que hereda las características de la clase padre y adicionan algunas específicas que ésta no posee.

Clase abstracta: Una clase es abstracta cuando no existe un objeto que sea instancia directa de ella, pero sí existe una subclase de ella que es instanciable.

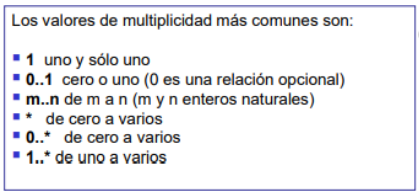
Clase concreta: Una clase concreta es aquella que es instanciable, es decir que existe al menos un objeto de esa clase.

Tipos de herencia:



RELACIONES ENTRE CLASES:
Las relaciones muestran el acoplamiento de las clases, es el número de clases con las que una clase concreta está relacionada. Una clase está acoplada si sus objetos lo están. Un objeto está acoplado con otro si uno de ellos actúa sobre el otro.

ASOCIACIÓN:
Es una conexión entre dos clases, la comunicación puede ser tanto uni como bi-direccional, una asociación puede tener nombre y se representa por una línea continua entre las clases asociadas, la asociación no es contenida por las clases, ni subordinada a las clases asociadas.



Cardinalidad

Clase asociación: Una dupla de objetos, instancias de cada una de las clases que participan en la asociación, se relaciona con una única instancia de la clase asociación. No importa la multiplicidad en ambos extremo.

Clase que modela la asociación: Para una dupla de objetos existe más de objeto asociado a la clase asociada.

AGREGACIÓN:
Reutilización y extensión: Se denomina reutilización al uso de clases u objetos desarrollados y probados en un determinado contexto, para incorporar esa funcionalidad en una aplicación diferente a la de origen.
La extensión se basa en aprovechar las clases desarrolladas para una aplicación, utilizándolas para la construcción de nuevas clases, en la misma u otra aplicación.

- La agregación consiste en definir como atributos de una clase a objetos de otras clases ya definidas.
- La agregación es una relación no simétrica en la que una de las clases cumple un papel predominante respecto de la otra.
- La agregación declara una dirección a la relación todo/parte.

Agregación (propiamente dicha): Este tipo de relación se presenta en aplicaciones en las cuales un objeto contiene como partes a objetos de otras clases, pero de tal modo que la destrucción del objeto continente no implica la destrucción de sus partes.
Los tiempos de vida de los objetos continente y contenido no están tan estrechamente acoplados, de modo que se pueden crear y destruir instancias de cada clase independientemente.

Composición: La forma más simple de reutilizar una clase es simplemente haciendo una nueva clase que la contenga. Esta técnica se llama composición. Las composiciones generan una relación de existencia entre el todo y cada una de sus partes.

- Un objeto de una clase contiene como partes a objetos de otras clases y estas partes están físicamente contenidas por el agregado.
- Los objetos agregados no tienen sentido fuera del objeto resultante. - Los tiempos de vida de los objetos continente y contenido están estrechamente acoplados
- La destrucción del objeto continente implica la destrucción de sus partes

CONCEPTOS CLAVE:
Encapsulamiento:

- Término formal que describe al conjunto de métodos y de datos de un objeto de manera tal que el acceso a los datos se permite solamente a través de los métodos propios de la clase a la que pertenece el objeto.

- La comunicación entre los distintos objetos se realiza solamente a través de mensajes explícitos.

Abstracción:

- La orientación a objetos fomenta que los programadores y usuarios piensen en las aplicaciones en términos abstractos.
- A partir de un conjunto de objetos, se piensa en los comportamientos comunes de los mismos para situarlos en superclases, las cuales constituyen un depósito para elementos comunes y reutilizables.

Polimorfismo:

- Capacidad que tienen objetos de clases diferentes, relacionados mediante la herencia, a responder de forma distinta a una misma llamada de un método.
- Fomenta la extensibilidad del software
- Software escrito para invocar comportamiento polimórfico se escribe en forma independiente del tipo de los objetos a los cuales los mensajes son enviados
- Nuevos tipos de objetos, que pudieran responder a mensajes existentes, pueden ser agregados en dicho sistema sin modificar el sistema base.

Persistencia:

- designa la capacidad de un objeto de trascender el tiempo o el espacio
- un objeto persistente conserva su estado en un sistema de almacenamiento permanente
- el objeto puede ser reconstruido por otro proceso y se comportará exactamente como en el proceso inicial