POO: Paradigma de programación, es decir una forma de analizar, diseñar y realizar soluciones. Se utiliza para la resolución de un problema a través de la interacción de objetos entre si

OBJETIVOS: *Escribir software fácilmente modificable y escalable. *Escribir software reusable. *Disponer de un modelo natural para representar un dominio.

VENTAJAS: *Fomenta la reutilización del software. *El software desarrollado es más flexible al cambio. *Es más cercano a pensar a la forma de las persona.

ABSTRACCIÓN: *Consiste en aislar un elemento de su contexto o del resto de los elementos que lo acompañan. *Denota según el observador las características esenciales de un objeto que lo distinguen de los demás. *La vista exterior del objeto puede definirse como un contrato, el cual el mismo se compromete a cumplir y los demás objetos dependen. Debemos identificar propiedades del objeto (atributos) y comportamientos (métodos).

OBJETO: Entidad que combina procedimientos e información, ejecuta operaciones y almacenan información. Los procedimientos o acciones que puede realizar un objeto se denominan métodos. La información que almacena un objeto sobre su estado se denomina atributos. un objeto es la instancia de una clase y estos se comunican entre si con mensajes (métodos).

Un objeto posee:

- Estado: dado por el conjunto de propiedades que posee y su valor en un momento dado.
- Comportamiento: Está dado por un conjunto de acciones que el mismo puede realizar.
- Identidad: En todo momento es diferenciable del resto y es constante.

CLASE: Mecanismo utilizado en la POO para abstraer conceptos(clasificación). Describe las características comunes a todos los objetos que pertenecen a ella. También especifica el comportamiento de los objetos, su estructura interna y le otorga la definición e implementación de todas sus acciones.

Método: es una acción que realiza el objeto

Atributo: es la característica de un objeto

DIFERENCIA ENTRE CLASE Y OBJETO: La diferencia entre estos es que las clases son definiciones estáticas y los objetos son entidades dinámicas.

ENCAPSULAMIENTO: Propiedad que asegura que la información de un módulo este oculta al exterior, así ocultando el estado de un objeto, solamente podremos modificar sus atributos mediante métodos.

CONSTRUCTORES: Es un método con el mismo nombre de la clase, estos se utilizan para controlar el estado inicial en el que se crea un objeto, no poseen ningún valor de retorno ni siquiera el tipo void, y estos no se pueden invocar directamente como otro método.

HERENCIA: Capacidad de crear nuevas clases basándose en clases previamente definidas de las que se aprovechan ciertos datos y métodos, se desechan otros y se añaden nuevos. En resumen, es un método de reutilización de código, se abstrae lo general en una clase padre y las clases hijas se centran en lo especifico, la desventaja de la herencia es que genera acomplamiento.

Composición: cuando una clase forma parte de otra como atributo

JERARQUÍA: Orden de subordinación de un sistema de clases.

POLIMORFISMO: Propiedad según la cual un mismo objeto puede considerarse como perteneciente a distintas clases. En resumen, cada clase tiene su propia forma de responder al mismo mensaje

ACCESO:

- Acceso privado: Solo son accesibles por los propios miembros de la clase, pero no desde funciones externas o desde funciones de clases derivadas. Es el acceso por defecto.
- Acceso público: Cualquier miembro público de una clase es accesible desde cualquier ora parte donde sea accesible el propio objeto.
- Acceso protegido: Respecto a las funciones externas, es equivalente al acceso privado, pero con respecto a las clases derivadas se comporta como el público.

DESTRUCTORES: Metodo especial que sirve para eliminar un objeto de determinada clase, si el objeto fue creado de forma dinámica hay que devolver la memoria de forma explicita usando el operador delete.

JERARQUIA DE CLASES: Cada nueva clase obtenida mediante herencia se conoce como clase derivada (clase hija/o), que derivada de una clase base (clase padre).

- CONSTRUCOTRS DE CLASES DERIVADAS: Cuando se crea u objeto de una clase derivada, primero se invoca al constructor de la clase o clases base y a continuación al constructor de la clase derivada. Si no se han definido los constructores de las clases, se usan los constructores por defecto que crea el compilador

DOWNCASTING Y UPCASTING: Un objeto de un tipo se puede tratar como objeto de otro tipo siempre que la clase fuente y la clse destino estén relacionadas por herencia. En C++ un puntero puede ser tipado por su clase general pero apuntar a una subclase.

- UPCASTING: Conversión de clase derivada a la clase base, se hace implícitamente.
- DOWNCASTING: Conversión de clase base a la clase derivada. Es explícito.
- Dynamic_cast: Verifica que una referencia a un objeto sea de un tipo o de un subtipo dado. Podemos utilizarlo antes de convertir un objeto para evitar errores.

MÉTODOS Y CLASES abstract: El calificador abstract condiciona el diseño de una jerarquía de herencia.

- Clases abstract: No pueden ser instanciadas.
- Métodos abstract: No pueden tener implementación. Deben ser implementados para las clases no abstractas que extiendan de su clase.