a. Objetos: Un objeto puede ser cualquier cosa, teniendo en cuenta que esta mismo tendrá datos y funciones que pueden interactuar con otras, y estas se pueden pertenecer a una clase.

b. Propiedades: Las propiedades son utilizados para acceder o modificar a los atributos.

c. Atributos: Son las características del mismo objeto. Son variables que pueden guardar los datos del estado que relacionan al objeto.

d. Estado: El estado de un objeto son los atributos a los que se les asignan valores (datos). La única manera de modificar el estado de un objeto es a través de sus métodos.

e. Métodos: También se le llaman funciones de miembro. Son las acciones (operaciones) que el objeto realiza por si mismo. Esto caracteriza su comportamiento. Las operaciones permiten que el objeto responda a los llamados externos o que actúe sobre otros objetos.

f. Clase: Las clases son como una clasificación o agrupa los objetos. proporcionándoles valores iniciales, e implementándoles métodos.

g. Instanciación: Es la creación de una instancia especifica para una clase.

Instancia: Cada objeto es una instancia de una clase, las cuales en su interior tendrá sus propiedades y funciones que se definieron en la clase pero la información de cada campo será propio de cada instancia.

h. Evento: Son instrucciones que se le da a los programas para que haga una cierta acción al detectar cuando un usuario esta interactuando con la interfaz en una aplicación.

i. Herencia: Es el comportamiento que hace que una clase pueda heredar los comportamientos y propiedades de otra, o incluso puede ampliarla o adaptarla según sus necesidades, pero siempre se heredó una base…

j. Abstracción: La abstracción es una forma de aislar un objeto para tener la oportunidad de analizar el objeto en específico, facilitando su comprensión.

k. Encapsulamiento: Es un proceso que ayuda a proteger elementos importantes, permitiendo el acceso a ellos a través de una interfaz definida para conservar la integridad de los datos.

l. Ocultamiento: Es la capacidad de que los clientes no puedan acceder a unos ciertos aspectos de una clase o componentes del software.

m. Modificadores de acceso:

Todos los tipos y miembros deben de tener un nivel de accesibilidad, esto controla a que pueden acceder o modificar.

1. public

Puede accesar cualquier miembro de cualquier otro código o de otro ensamblado que haga referencia a este. Y el nivel de accesibilidad de los miembros públicos de un tipo de controla mediante el nivel de accesibilidad del propio tipo.

1. protected

Solo el código de la misma “class” o bien de una “class” derivada puede acceder al tipo o miembro.

1. private

Solo el código de la misma clase puede acceder al tipo o miembro.

n. Modularidad:

El modularidad es dividir el programa en módulos que puedan compilarse por separado, pero siempre mantendrá las conexiones con otros módulos.

El modularidad tiene principios y son los siguientes:

* Capacidad de descomponer un sistema complejo.
* Capacidad de componer a través de sus módulos.
* Comprensión de sistema en partes.

o. Polimorfismo: Es la capacidad de poder implementar las propiedades o métodos heredados de diferentes en varias abstracciones.

3. Mediante dibujos o gráficas, describa un ejemplo de cada una de las siguientes características (no pueden ser los mismos ejemplos expuestos en clase):

a. Clase y objetos: instanciación, atributos, métodos

sí tenemos una clase llamada "Gato" algunos de sus atributos pueden ser el "Nombre" y el "Color del pelaje", mientras que algunas de sus métodos podrían ser "Comer" y "Maullar".

La clase Gato permitirá crear objetos tipo Gato y cada uno de estos objetos será una instancia de la clase Gato, en su interior tendrán definidos las propiedades y los métodos que se definieron en la clase, pero la información que haya en sus campos será propia de cada instancia, es decir cada objeto será una instancia independiente con un estado propio, esto hace que podamos tener por ejemplo dos instancias de la clase Gato cada una con un nombre y pelaje distinto.

b. Clases: herencia

Ejemplo de la vida real, imaginemos que queremos construir una casa. En lugar de diseñar y construir una casa completamente nueva desde cero, podríamos basarnos en un diseño existente, modificarlo para satisfacer nuestras necesidades específicas y construir nuestra casa basada en ese diseño modificado.