* **Objetos y clases**:

Un objeto es la representación de una unidad física o abstracta, la cual tiene atributos y funcionalidades(responsabilidades).

Una clase es la abstracción que clasifica un conjunto de objetos que tienen atributos y responsabilidades en común.

* **Objetos y mensajes**:

Consiste en darle a un objeto la información y que tiene que hacer, en vez de crear una sola unidad procedimental.

Existen 3 tipos de mensajes:

1. Interrogativos: Ordena al objeto a brindar información.
2. Informativos: Ordena al objeto a cambiar su estado.
3. Imperativos: Ordenan al objeto a realizar una acción.

* **Abstracción**:

Proceso de simplificar un problema hasta su enunciado o situación fundamentar, con la información estrictamente necesaria para resolverlo.

* **Encapsulamiento**:

Consiste en agrupar elementos relacionados en una única unidad, para ser referenciada a través de un identificador o nombre.

* **Ocultamiento de la información**:

A través del encapsulamiento se logra ocultar la información, con el fin de restringir la visibilidad a otras unidades de datos.

* **Cohesión**:

Cohesión es la medida de relación entre los métodos de un objeto.

Típicos problemas de cohesión:

1. Pérdida de responsabilidad entre instancias.
   1. Declaracion de metodos que imposibilitan a la clase.
2. Herencia o implementación incorrecta de interfaz.
   1. Mal uso de la herencia | no se utiliza la regla “es un”.
   2. Implementación de interfaces que incrementan la cantidad de atributos de la clase.
3. Redundancia con otras clases.
   1. Funcionalidades ya implementadas en otras subclases.

* **Acoplamiento**:

Corresponde a la medida de dependencia entre las partes.

1. **Bajo acoplamiento**: esto es lo que se busca en la orientación a objetos, permite la reutilizacion de codigo. Es la fácil adaptación de software en distintas áreas.
2. **Alto acoplamiento**: Define que un componente sólo puede ser usado o implementado bajo ciertas condiciones.

Existen 5 niveles de acoplamiento:

1. Nulo.
2. Exportación.
3. Apertura.
4. Covertura.
5. Total.

* **Herencia**:

Propiedad que permite construir software incrementalmente a través de la especialización de unidades preexistentes.

* **Polimorfismo**:

Capacidad de un objeto de tomar más de una forma.

* **Relación entre objetos**:

1. Composición: Las instancias forman parte integral del contenedor.
2. Agregación: Las instancias pueden formar parte del contenedor, más sin embargo el contenedor puede seguir existiendo sin estas instancias.