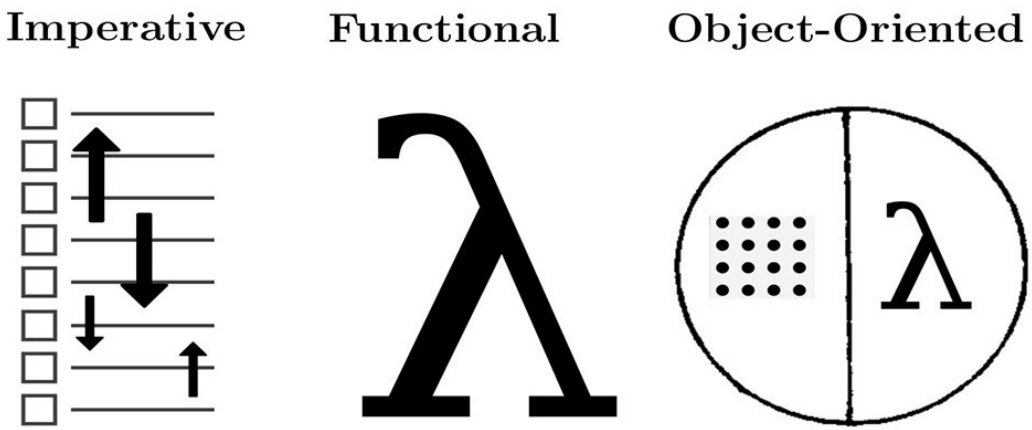
# Ingeniería en Informática
# Ingeniería en Computación
## Programación Orientada a Objetos

Dr. Mario Marcelo Berón

# Conceptos Preliminares

## PARADIGMAS DE PROGRAMACIÓN




## Paradigma

- Forma de entender y representar la realidad.
- Conjunto de teorías, estándares y métodos que, juntos, representan un modo de organizar el pensamiento.

# Conceptos Preliminares


## Abstracción

Supresión intencionada de algunos detalles de un proceso o artefacto, con el fin de destacar más claramente otros aspectos, detalles o estructuras.

# Conceptos Preliminares


## Paradigmas de Programación

- Imperativo
- Funcional
- Lógico
- Orientado a Objetos

## Atención

Los paradigmas antes mencionados no son los únicos.

# Conceptos Preliminares

## Paradigmas de Programación

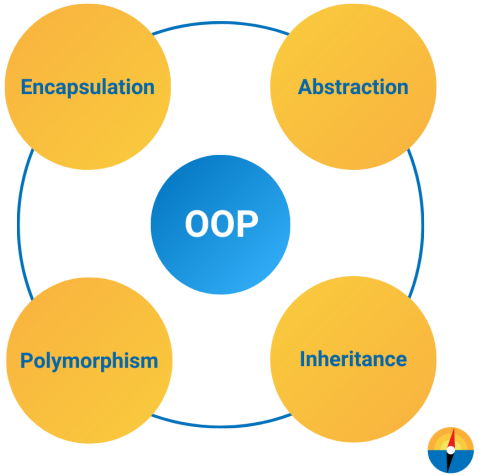
- Imperativo
  - Describe cómo debe realizarse el cálculo.
  - Un cómputo consiste en una serie de sentencias ejecutadas según un control de flujo explícito que modifican el estado del programa.
  - Es el estándar de facto.
  - Lenguajes: C, Pascal, C++, etc.
- Funcional
  - Basados en modelos de cómputo *cálculo lambda*, *Lógica Combinatoria*.
  - El control de flujo está asociado a la composición funcional, la recursividad y/o técnicas de reescritura y unificación.
  - Lenguajes: LISP, Miranda, Haskell.

# Conceptos Preliminares

## Paradigmas de Programación

- Lógico:
  - Basados en lógica de predicados de primer orden.
  - Los programas se componen de hechos, predicados y relaciones.
  - Evaluación basada en unificación y backtracking.
  - La ejecución consiste en la resolución de un problema de decisión, los resultados se obtienen mediante la instanciación de variables libres.
  - Lenguaje: Prolog.
- Orientado a Objeto: paradigma de programación que afecta a distintos niveles del desarrollo de software:
  - Análisis y diseño de software.
  - El modo de organización de los programas.
  - El sistema de tipos.
  - La filosofía de programación.
  - Lenguajes: eiffel, smalltalk, java, etc.

# El Paradigma Orientado a Objetos

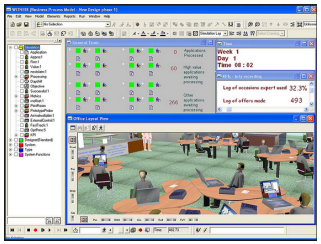



## Concepción

*Metodología de desarrollo de aplicaciones en la cual éstas se organizan como colecciones cooperativas de objetos, cada uno de los cuales representan una instancia de alguna clase, y cuyas clases son miembros de jerarquías de clases unidas mediante relaciones de herencia.*

*Grady Booch*

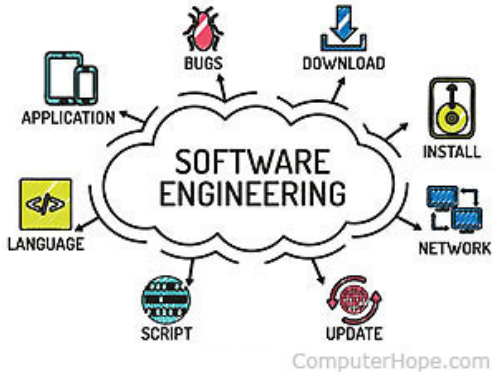
# Génesis



## Simulación

- Creación del Lenguaje de Programación SIMULA-67. Años 60. Objetivo: Desarrollar modelos del mundo real y sobre ellos ejecutar simulaciones.
  - Objetos: Identidad, Estructura, Comportamiento, Interacción.
  - SIMULA-67 introduce el concepto de Clase debido a la necesidad de crear muchos objetos con una misma estructura pero con Identidad diferente.
  - Objetos: Entidades reactivas capaces de responder requerimientos del exterior a través de la realización de operaciones sobre su estrctura interna.

# Génesis




## Ingeniería de Software

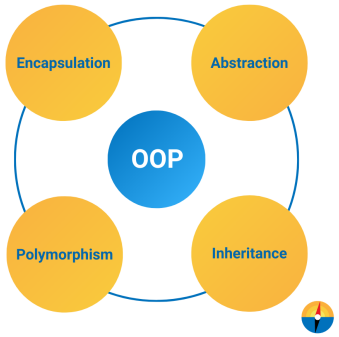
- Programación Estructurada. Refinamientos sucesivos.
- Reutilización de Software (Procedimientos y Funciones).
- Paso siguiente en los lenguajes para facilitar la reutilización: Los Módulos.
- Información Oculta: Datos locales a un módulo y procedimientos y funciones son servicios disponibles en el módulo para acceder a los datos desde el exterior. Módulos como abstracción de datos y no de control.

# Génesis


## Ingeniería de Software

- Independencia del Contexto (Permite la reutilización).
- Abstracción de Datos (Garantiza abstracción).
- Encapsulamiento (Garantiza abstracción y protección).
- Modularidad (Garantiza la composición de las partes).

# El Paradigma Orientado a Objetos



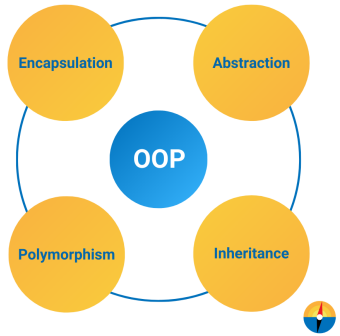

## Cambios

1. Los programas se organizan en clases.
2. Los programas se ejecutan a través del pasaje de mensajes entre objetos.

## Importante

- No basta con utilizar un lenguaje orientado a objetos para programar orientado a objetos.
- Se debe seguir los lineamientos del paradigma orientado a objeto.

# El Paradigma Orientado a Objetos




## Popularidad

1. POO es el paradigma de programación predominante por:
   1. Su alto nivel de escalabilidad.
   2. Su modelo de abstracción que posibilita razonar con técnicas que la gente usa para resolver problemas.
2. Gran desarrollo de Herramientas Orientadas a Objetos en todos los dominios.

# El Paradigma Orientado a Objetos

## Meta de la Programación Orientada a Objetos

El objetivo principal de la programación orientada a objetos es:

**Mejorar la Calidad de las Aplicaciones**

# El Paradigma Orientado a Objetos

## Medición de la Calidad: Factores Internos y Externos

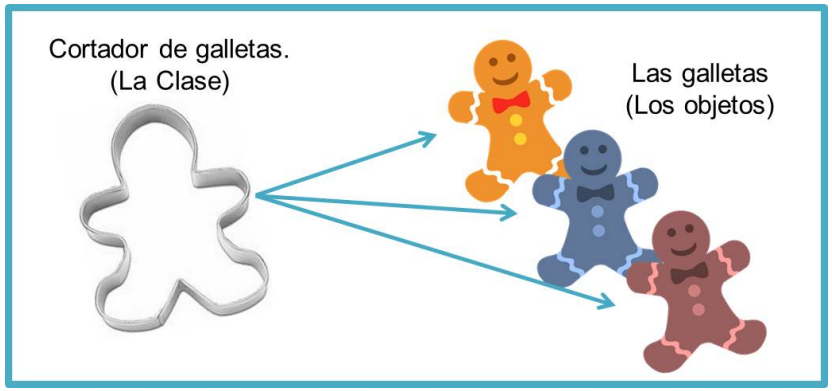
1. Factores Internos: Cualidades aplicables a los productos de software tales como modularidad, legibilidad, etc perceptibles solo por ingenieros de software que tienen acceso al código fuente.
2. Factores Externos
   - Correctitud: Es la capacidad de los productos de software de realizar exactamente la tarea definida en su especificación.
   - Robustez: Es la capacidad de los sistemas de software de reaccionar apropiadamente ante condiciones anormales.
   - Extensibilidad: Es la capacidad de adaptación de los productos de software a los cambios de especificación.
   - Reusabilidad: Capacidad de los elementos de software para ser utilizados en la construcción de diferentes aplicaciones.
   - Compatibiliad: Es la facilidad de combinar elementos de software con otros elementos.

# El Paradigma Orientado a Objetos

## Medición de la Calidad: Factores Internos y Externos

1. Factores Externos
   - Eficiencia: Es la capacidad de un sistema de software de requerir pocos recursos de hardware, tales como: tiempo del procesador, espacio ocupado en memoria interna y externa, ancho de banda utilizado en dispositivos de comunicación.
   - Portabilidad: Es la facilidad de transferir productos de software a varios ambientes de hardware y software.
   - Facilidad de Uso: Es la facilidad en la cual muchas personas de varios trasfondos conceptuales pueden aprender a usar productos de software y aplicarlos para resolver problemas.
   - Funcionalidad: Son las tareas provistas por un sistema.
   - Timeliness: Es la capacidad de un producto de software de ser entregado antes o cuando el usuario lo necesita.

# El Paradigma Orientado a Objetos




## Manera de Ver al Mundo

1. Objetos y Clases
2. Mensajes y Métodos
3. Jerarquía de Clases

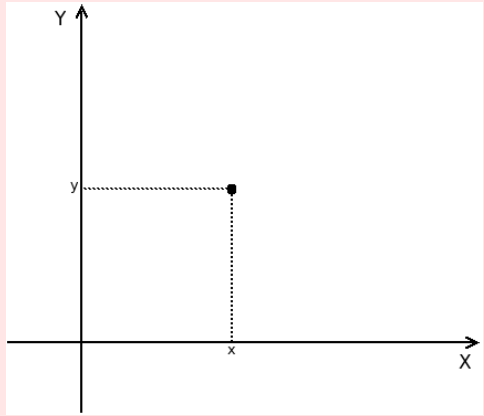
# Objeto

## Objeto Concreto

- Identificador: 331418320-M
- Estado: Nombre, Edad, Domicilio, Estado Civil, Número de Teléfono, etc.
- Comportamiento: Caminar, Ir al trabajo, Dormir, Almorzar, Cenar, Despertar, etc.

# Objeto

## Objeto Abstracto




- Identificador: Punto-1
- Estado: Coordenada X, Coordenada Y.
- Comportamiento: Asignar un Valor a la Coordenada X, Asignar un Valor a la Coordenada Y, Recuperar el Valor de la Coordenada X, Recuperar el Valor de la Coordenada Y, etc.

# Objeto

## Ejemplos

- Objetos Concretos:
  - Una bicicleta
  - Un lápiz
  - Una Computadora
  - Etc.
- Objetos Abstractos:
  - Una lista
  - Una Pila
  - Una Fila
  - Etc.

¿Podría dar más ejemplos de objetos?

# Objeto

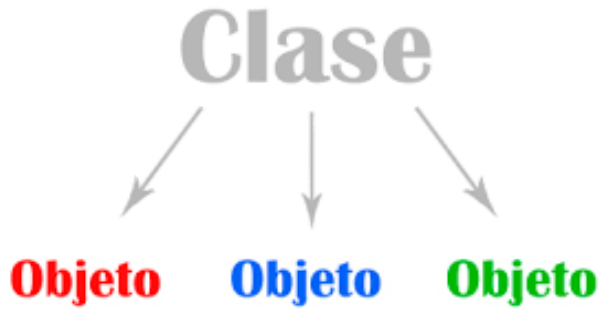
## Definición

Es la representación abstracta de una entidad autónoma que posee las siguientes características:

- Una identidad única.
- Un conjunto de atributos privados (el estado interno del objeto).
- Un conjunto de operaciones que son las únicas que pueden acceder de forma directa al estado interno. Estas operaciones pueden ser:
  - Invocadas desde el exterior (públicas). Estas operaciones son conocidas como la interfaz del objeto.
  - Sólo accesibles desde operaciones internas (privadas).

  Ambas clases de operaciones representan el comportamiento total del objeto.

# Objeto




- Atributos de un objeto = Variables de Instancia.
- Operaciones capaz de realizar = Métodos de Instancia.

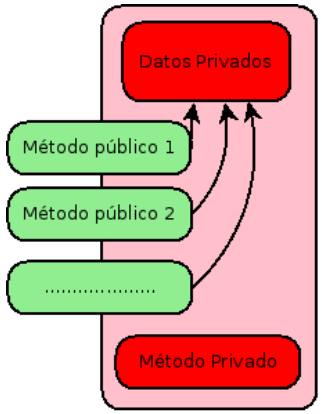
# Encapsulamiento: Propiedad Fundamental

## Encapsulamiento: Idea

| Auto | Motor |
| --- | --- |
|  |  |

Para manejar un auto no es necesario saber cómo funciona el motor.

# Encapsulamiento: Propiedad Fundamental




## Comentarios

- Un objeto es una unidad computacional cerrada y autónoma o sea un módulo capaz de realizar operaciones sobre su propio estado interno y devolver respuestas al exterior.
- Un objeto es capaz de prestar un servicio a través de la activación de los métodos públicos. Los servicios se traducen en respuestas a las activaciones realizadas desde el exterior.

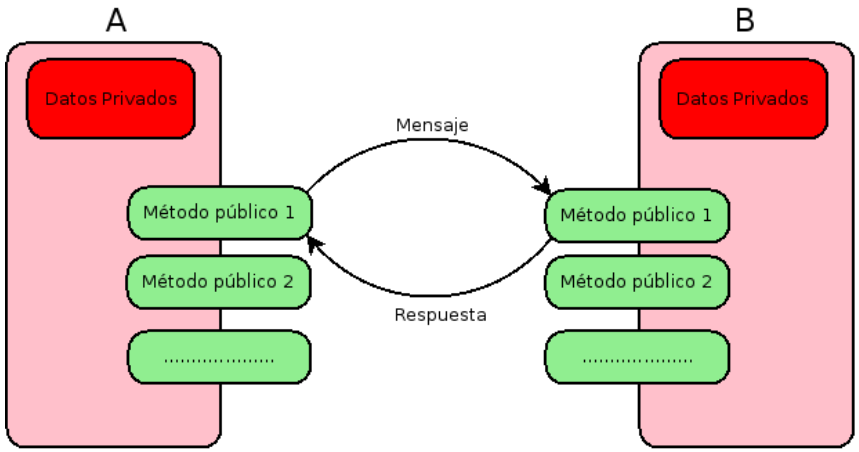
# El Paradigma Orientado a Objetos

## Mensajes y Métodos

Un mensaje se diferencia de un procedimiento/función en dos aspectos:

1. En el mensaje siempre hay un receptor, lo cual no ocurre en una llamada a procedimiento.
2. La interpretación de un mismo mensaje puede variar en función del objeto receptor.

# Mensajes




## Comentarios

- En cada computación existe un objeto emisor de un mensaje y un objeto receptor del mismo.
- El envío de un mensaje se realiza durante la ejecución de un método del emisor que necesita un servicio particular del receptor.

# Mensajes

## Formas Comunes de Envío de Mensajes

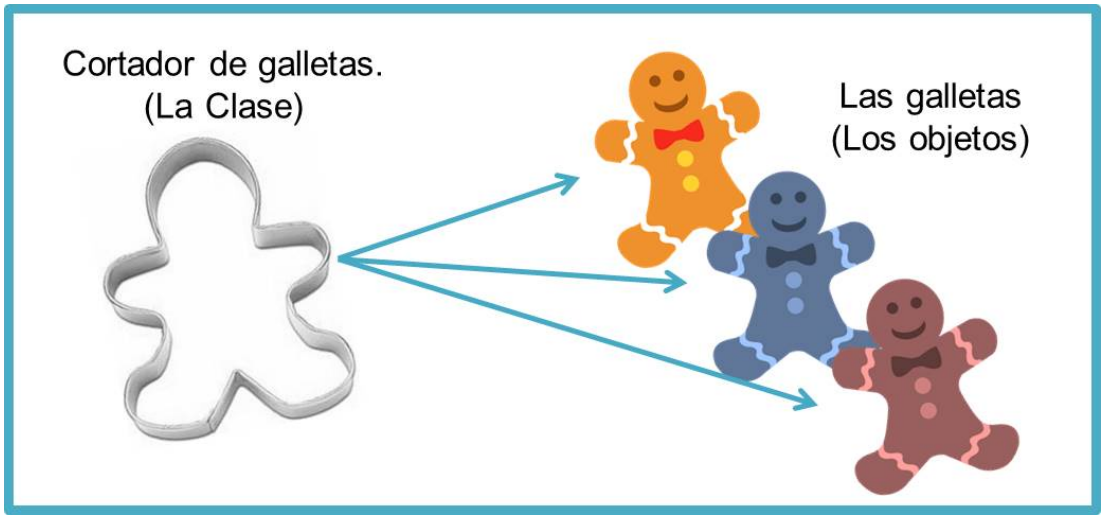
- receptor.mensaje()
- receptor.mensaje($arg_1$,$arg_2$,...,$arg_n$)
- resultado=receptor.mensaje()
- resultado=receptor.mensaje($arg_1$,$arg_2$,...,$arg_n$)

Aquí **resultado** y **receptor** representa identificadores de variables y **mensaje()** y **mensaje($arg_1$,$arg_2$,...,$arg_n$)** una forma de identificar a un método a ejecutar en el receptor.

## Atención

Mensaje y Método son entidades distintas. Los métodos son invocados por medio del envío de mensajes a un objeto.

# Clase




## Definición

Objetos particulares que sirven para:

- Contener la estructura y comportamiento de objetos similares.
- Crear objetos particulares que poseen tal estructura y comportamiento.

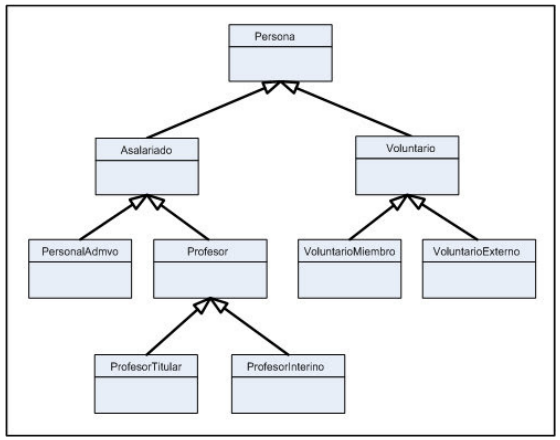
Los objetos son **fábricas** o **moldes** utilizadas para crear instancias.

# Clase


## Atención

Los objetos son **Instancias de una Clase**.

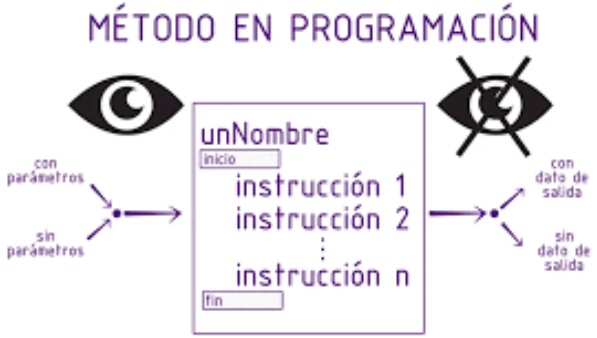
# Jerarquía de Clases




## Jerarquía de Clases

En la vida real, mucho conocimiento se organiza en términos de jerarquías. Este principio por el cual el conocimiento de una categoría más general es aplicable a una categoría más específica se denomina generalización, y su implementación en POO se llama herencia.

# Ligadura de Métodos




## Ligadura de Métodos

Instante en el cual una invocación a un método se asocia al código que se de debe ejecutar, puede ser:

1. Estática
2. Dinámica

# Características de los Lenguajes Orientados a Objetos

## Características

1. Todo es un objeto.
2. Cada objeto se construye a partir de otros objetos.
3. Todos los objetos de la misma clase pueden recibir los mismos mensajes (realizar las mismas acciones).
4. La clase es el lugar donde se define el comportamiento de los objetos y su estructura interna.
5. Las clases se organizan en una estructura de árbol de raíz única, llamada Jerarquía de Clases.
6. Un programa es un conjunto de objetos que se comunican mediante el envío de mensajes.

# Características de los Lenguajes Orientados a Objetos

## Características

- Polimorfismo: Capacidad de una entidad de referenciar elementos de distinto tipo en distintos instantes.
- Genericidad: Definición de clases parametrizadas que definen tipos genéricos.
- Gestión de Errores: Tratamiento de condiciones de error mediante excepciones.

# Características de los Lenguajes Orientados a Objetos

## Características

- Aserciones: Expresiones que especifícan qué hace el software en lugar de cómo lo hace.
  1. Precondiciones:Propiedades que deben ser satisfechas cada vez que se invoque un servicio.
  2. Postcondiciones: Propiedades que deben ser satisfechas cada vez que se finaliza la ejecución de un servicio.
  3. Invariantes: Aserciones que expresan restricciones para la consistencia global de sus instancias.

# Características de los Lenguajes Orientados a Objetos

## Características

- Tipado Estático
  1. Se asegura en tiempo de compilación que un objeto entiende los mensajes que se le envían.
  2. Evita errores en tiempo de ejecución
- Recolector de basura(garbage collector): Permite liberar automáticamente la memoria de aquellos objetos que ya no se utilizan.
- Concurrencia: Permite que diferentes objetos actúen al mismo tiempo, usando diferentes threads o hilos de ejecución.

# Características de los Lenguajes Orientados a Objetos

## Características

- Persistencia: Es la propiedad por la cual la existencia de un objeto trasciende la ejecución del programa. Normalmente esta tarea se lleva a cabo con una base de datos para almacenar objetos.
- Reflexión: Capacidad de un programa de manipular su propio estado estructura y comportamiento.