UCA

Programación Orientada a Objetos I

PARADIGMA ORIENTADO A OBJETOS

E. Christian López Pasarón

chlopezpasaron@uca.edu.ar

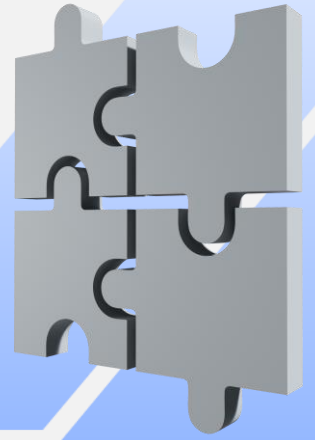
- Licenciado en Sistemas de Información, Magister en Administración de Negocios con Orientación en Dirección de Sistemas.
- Director del Laboratorio de Informática y Redes de la Facultad de Ingeniería y Ciencias Agrarias de Universidad Católica Argentina (UCA)
- Docente de varias materias en UCA
- Docente de varias materias y tutor de la Práctica Profesional Supervisada en la USAL
- Ex Docente de otras asignaturas (Seguridad Informática, Sistemas de Información, Programación, etc.)
- Perito, Consultor y Profesional Informático

Hablemos de software…

Complejidad del software:

- No es un accidente
- Es una propiedad inherente al software
- Se deriva de 4 elementos:
  - Complejidad del dominio: Propia del problema a resolver
  - Dificultad de gestión: Miles de líneas de código, cientos de ficheros, muchos desarrolladores y gente implicada. La buena gestión es vital
  - Flexibilidad de las herramientas: componente de terceros confiables.
  - Comportamiento impredecible del software: Hay tantas variables, procesos, etc. que algo no previsto puede provocar errores

Hablemos del software…



¿Cómo tratamos la complejidad?

1. Descomponer:
   Dividir el problema mayor en partes menores

   - Independientes / Más fáciles de tratar / Con menor probabilidad de error / Reutilizables / Confiables
     - Descomposición algorítmica: Tareas simples (y cada vez más simples)
     - Descomposición orientada a objetos: Objetos de cuya interacción surge la solución (cada objeto se puede descomponer en más objetos)

   Ventaja OO vs Algorítmica: reusabilidad / adaptación al cambio

   Ventaja Algorítmica vs OO: Para problemas pequeños/sencillos

Hablemos del software…

2. Abstraer
   - Ignorar detalles poco significativos
   - Concentrarse en aspectos esenciales
   - Construcción de un modelo simplificado
   - Adoptar varios niveles de abstracción

3. Jerarquizar
   - Ordenar
   - Agrupar por similitudes y/o diferencias

Paradigmas de programación

Modelo conceptual para desarrollar programas

- Orientado al procedimiento: Imperativo – C/Fortran
- Orientado a funciones: Funcional, declarativo – Lisp/Haskel
- Orientado a lógica: Reglas y predicados – Prolog
- Orientado a objetos: Relaciones entre objetos – SmallTalk/Java

Hay modelos mixtos o híbridos, que mezclan paradigmas

Y hay nuevos paradigmas que van surgiendo (orientado a aspectos, etc.)

Que es un “Paradigma”

Paradigma:
(en el contexto de programación e informática)

- Conjunto de principios, patrones, enfoques o modelos que establecen la manera en que se debe abordar la resolución de problemas y la construcción de programas o sistemas de software.

- Los paradigmas de programación son enfoques conceptuales fundamentales que guían la forma en que se desarrolla el software, definiendo cómo se estructura el código, cómo se organizan los datos, cómo se realizan las interacciones y cómo se resuelven los problemas.

Programación Orientada a Objetos

Programación Orientada a Objetos (POO):

Es un enfoque fundamental en la programación que se basa en la creación y manipulación de "objetos" como las unidades principales de diseño y organización del código.

En este paradigma, los objetos son instancias de "clases", que actúan como plantillas que definen tanto las características (atributos o propiedades) como los comportamientos (métodos o funciones) que los objetos tendrán.

Programación Orientada a Objetos

Mecanismos básicos POO:

- Objetos
- Clases
- Mensajes
- Métodos

Objetos

Objeto: Encapsulamiento genérico de datos y de los procedimientos para manipularlos.

- Un programa tradicional se compone de procedimientos y datos.
- Un programa OO consiste solamente de objetos.

Las personas tenemos una idea clara de lo que es un objeto

- Un objeto podría ser real o abstracto, por ejemplo una organización, una factura, una pantalla de usuario, un avión, un vuelo de avión, una reservación aérea.

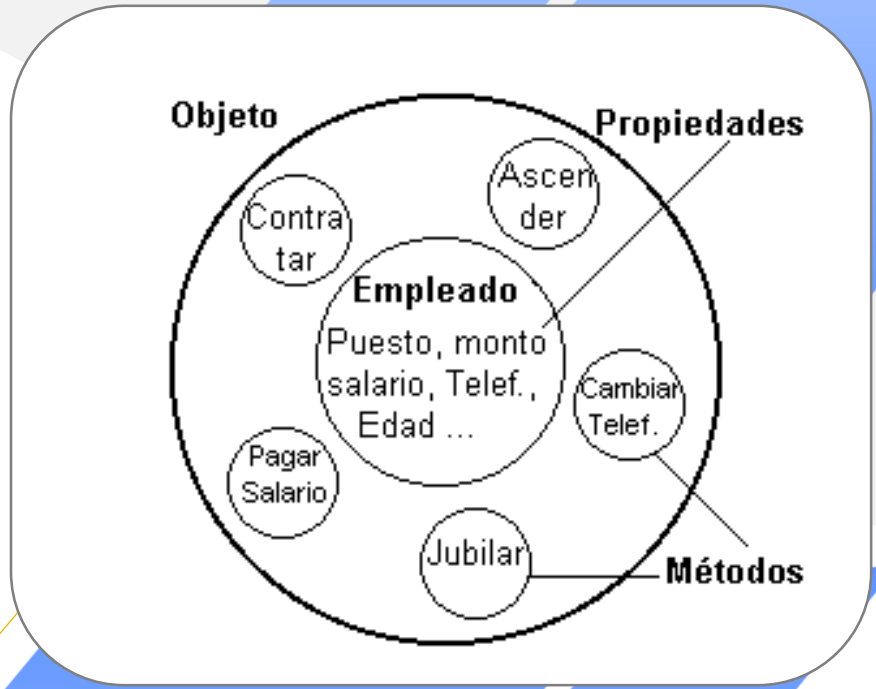
Objetos

- Un objeto es una cosa, generalmente extraída del espacio del problema (dominio) o del espacio de la solución (codominio)
- Todo objeto tiene asociados: Un nombre (se puede identificar) un estado, Atributos y Acciones que realiza (comportamiento)
- Un objeto actúa cuando recibe un ‘Mensaje’ del exterior para el cual tenga un comportamiento asociado.

Objetos

Dentro del software orientado a objeto, un objeto es cualquier cosa, real o abstracta, acerca de la cual almacenamos datos y los métodos que controlan dichos datos.

La programación orientada a objetos (POO) encapsula datos (atributos o propiedades) y métodos (comportamientos o acciones) en objetos; de esta manera los datos y métodos de un objeto están íntimamente relacionados entre sí.

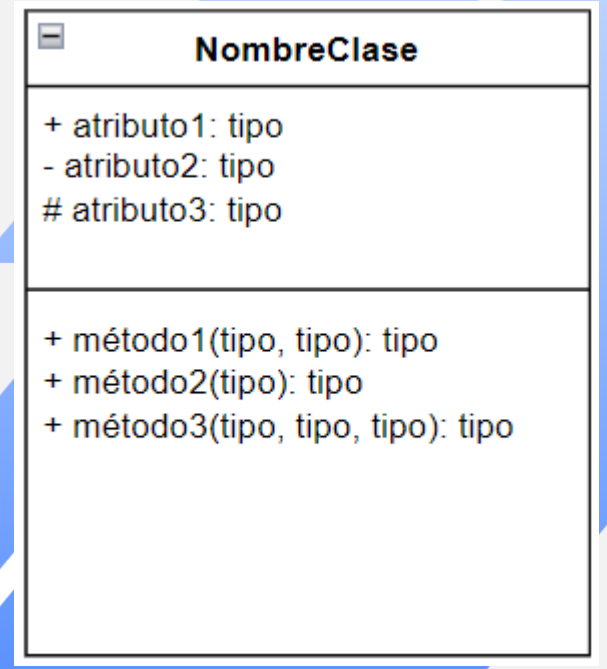


Objetos

- Todos los objetos son ejemplares de alguna clase
- Una clase es una descripción generalizada de un conjunto de objetos similares.
- Todos los objetos que existen dentro de una clase heredan sus atributos y los métodos disponibles para la manipulación de esos atributos
- Un objeto es una instancia de una clase
- Un objeto se distingue de otros miembros de la clase por sus atributos.

Clases

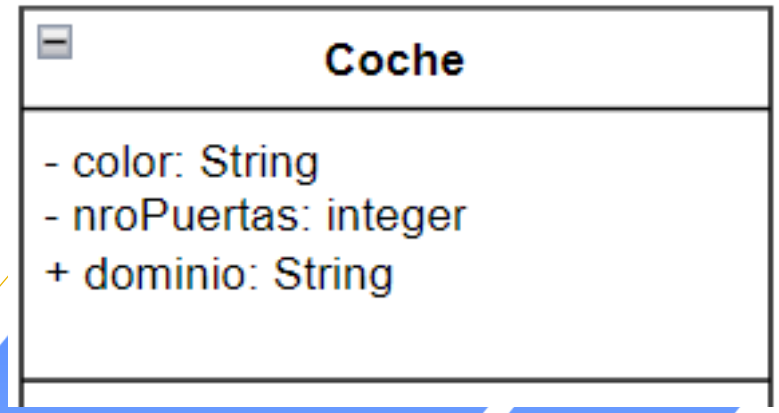
- Es una descripción de un conjunto de objetos similares que comparten los mismos atributos, operaciones, relaciones y semántica.
- Posee un ***nombre***, ***atributos*** y ***métodos***
- Un objeto es una “instancia” de una clase.



Instancia: Es una manifestación concreta de una clase (un objeto de valores concretos). También se le suele llamar ocurrencias.

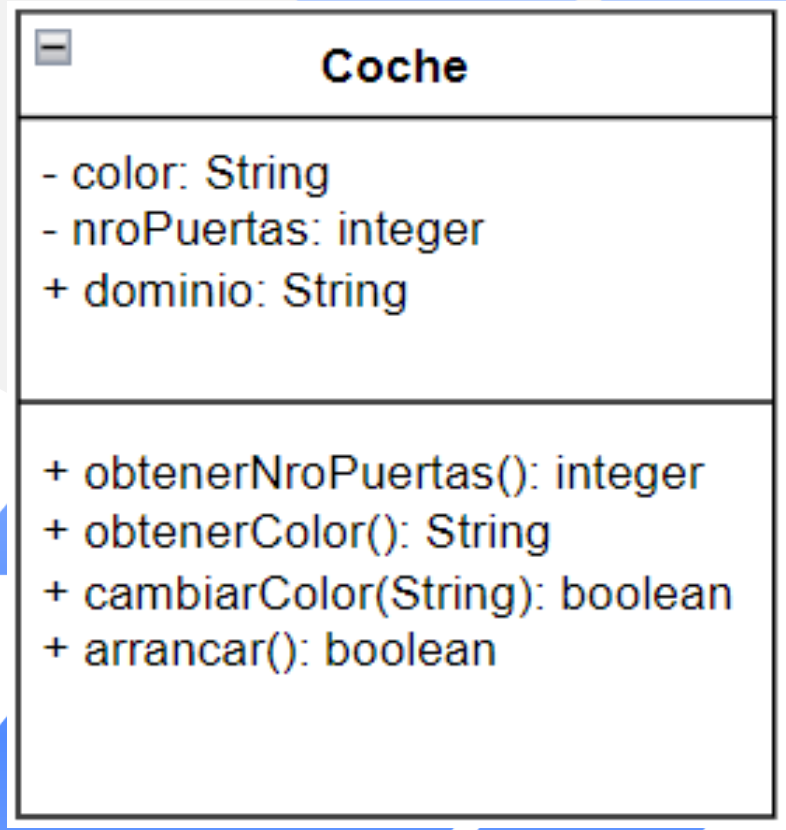
Clases: Atributos

- Es una característica concreta de una clase. Por ej. Atributo de la clase COCHE pueden ser el Color, Número de Puertas.
- Es una característica de una clase que la define o describe
- Puede tener mucho o ningún atributo
- Se puede identificar su tipo
- Se puede inicializar en la definición



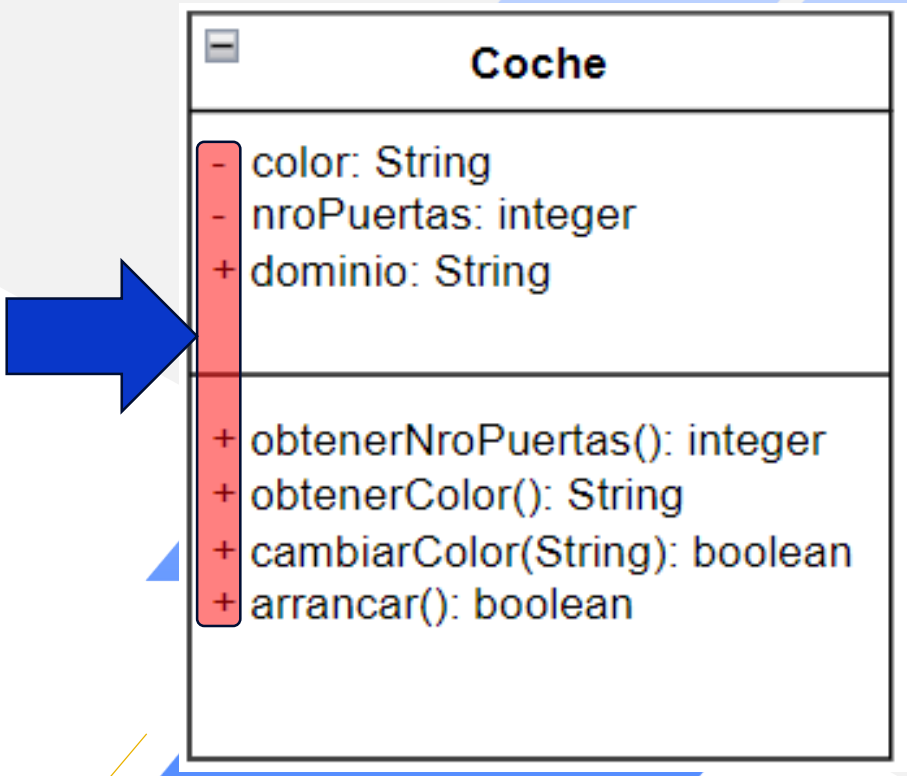
Clases: Métodos

- Es una operación concreta de una determinada clase. Por ej. De la clase “Coche” podríamos tener un método arrancar(), hace poner en marcha el coche.
- Es el accionar de un objeto ante la recepción de un “Mensaje” que viene del exterior.
- Son comportamientos o acciones, especifican la forma en que se controlan los datos de **un** objeto. Los métodos en un objeto sólo hacen referencia a la estructura de datos de **ese** objeto, **no deben tener acceso directo a las estructuras de datos de otros objetos**. Para utilizar la estructura de datos de otro objeto, deben enviar mensajes a éste.
- Desde el punto de vista de la programación, los métodos son funciones que pueden ser llamadas dentro de una clase o por otras clases.



Clases: Visibilidad

- Determina el nivel de encapsulamiento de los elementos de una clase.
  - (-) Privado : Los atributos/operaciones son visibles solo desde la propia clase.
  - (+) Los atributos/operaciones públicos son visibles a otras clases (cuando se trata de atributos se está transgrediendo el principio de encapsulación)
  - (#) Los atributos/operaciones protegidos están visibles para la propia clase y para las clases derivadas de la original



Clases: Mensajes

- Es la forma de comunicarse e interactuar que tienen los objetos.
- Un objeto puede recibir infinitos mensajes, pero solo reacciona frente a los mensajes que reconoce.
- El conjunto de mensajes que un objeto puede recibir se denomina interfaz (algunos autores también lo llaman protocolo).
- Un mensaje puede incluir información para clarificar la petición.

```
objeto1.arrancar();
objeto1.cambiarColor("Rojo");
```

Programación Orientada a Objetos

Características de la orientación a objetos:

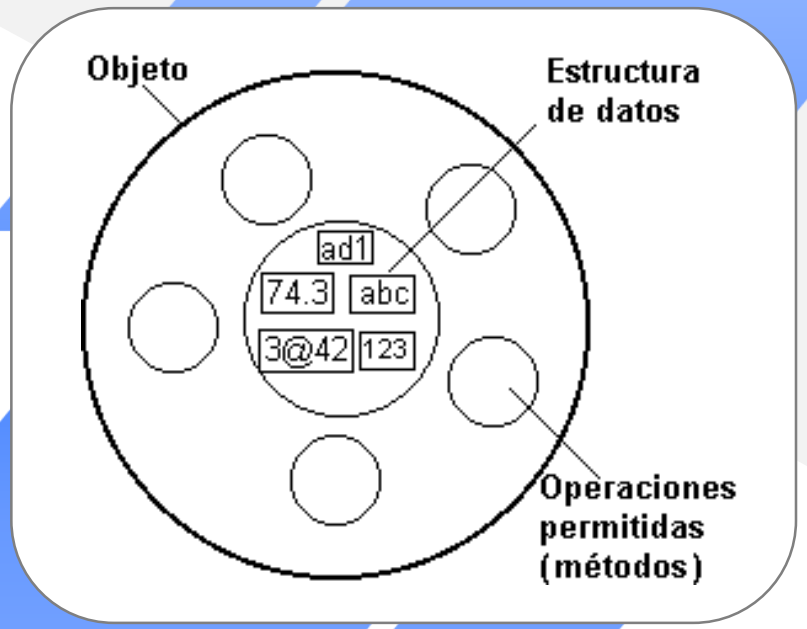
- Abstracción
- Encapsulamiento
- Herencia
- Polimorfismo

Abstracción

- Generalización conceptual de los atributos y métodos de un determinado conjunto de objetos.
- La abstracción hace referencia a quitar las propiedades y acciones de un objeto para dejar solo aquellas que sean necesarias para la resolución del problema en cuestión.
- La clave de la OO consiste en abstraer los métodos y los datos comunes a un conjunto de objetos y almacenarlos en una clase.

Encapsulamiento

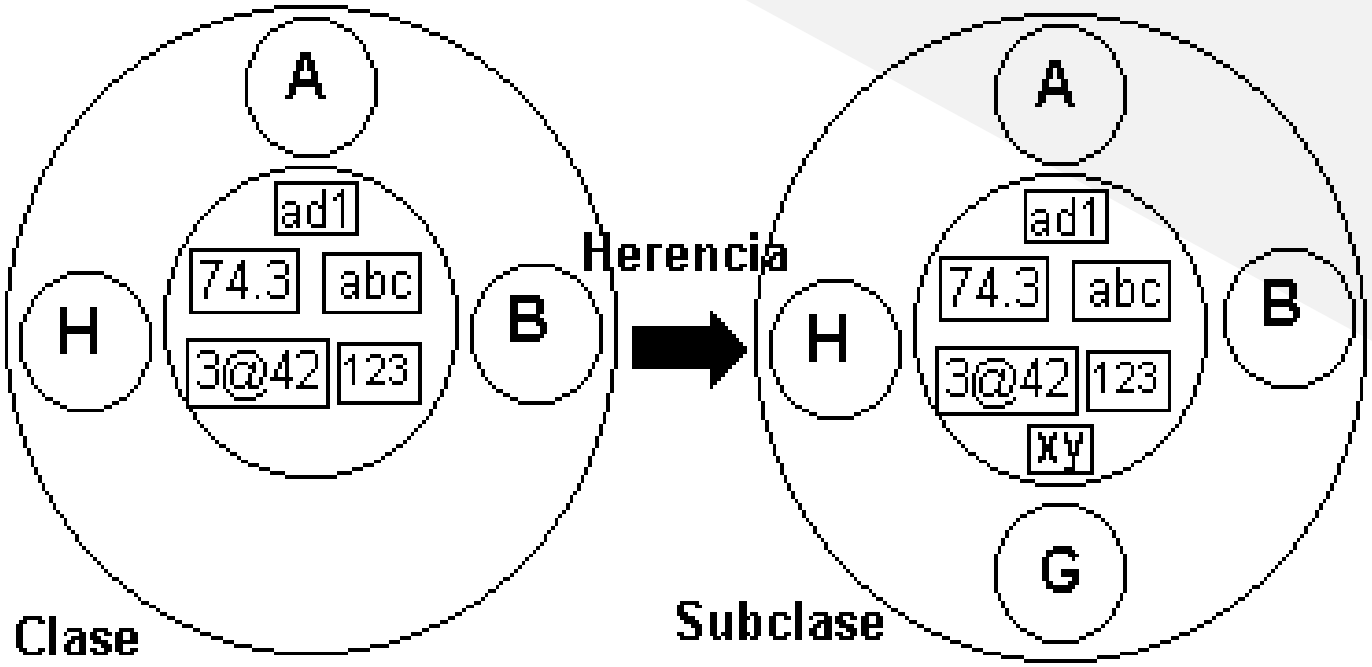
- La existencia de atributos y operaciones dentro de una estructura común (la clase) es lo que en el paradigma de O.O. se conoce como Encapsulamiento
- El encapsulado es el resultado de ocultar los detalles de implementación de un objeto respecto de su usuario
- Los atributos están cerrados al exterior, estableciendo la privacidad del objeto y confidencialidad de sus atributos y métodos
- Un objeto oculta su funcionamiento al exterior
- El objeto esconde sus datos de los demás objetos y permite el acceso a los datos mediante sus propios métodos. Esto recibe el nombre de ocultamiento de información y evita la corrupción del objeto
- Permite tratar a los objetos como cajas negras, sin considerar su estructura interna. Tiene dos beneficios adicionales:
  - Modularidad
  - Cohesión



Herencia

- La herencia es un mecanismo que permite la definición de una clase (llamada hija) a partir de la definición de otra ya existente (llamada padre)
- Es la característica clave de los sistemas orientados a objeto para propiciar la reusabilidad
- Una clase implanta el tipo de objeto. Una subclase hereda propiedades de su clase padre, una sub-subclase hereda propiedades de las subclases, etc. Una subclase puede heredar la estructura de datos y los métodos, o algunos de los métodos, de su superclase. También tiene sus métodos e incluso tipos de datos propios
- Se dice entonces que la clase hija 'HEREDA' todas las características de la clase padre; pudiendo modificarlas o agregar nuevas características propias.
- Organiza las clases en un árbol Jerárquico
- Cualquier cambio en los datos u operaciones en un padre se hereda automáticamente en las hijas

Herencia



Una clase puede tener sus propios métodos y estructura de datos, así como también heredarlos de su superclase.

Polimorfismo

- Es la capacidad de los distintos objetos (pertenecientes a distintas clases) de reaccionar a un mismo mensaje, pero cada uno a su manera
- Facilita la reusabilidad del Código