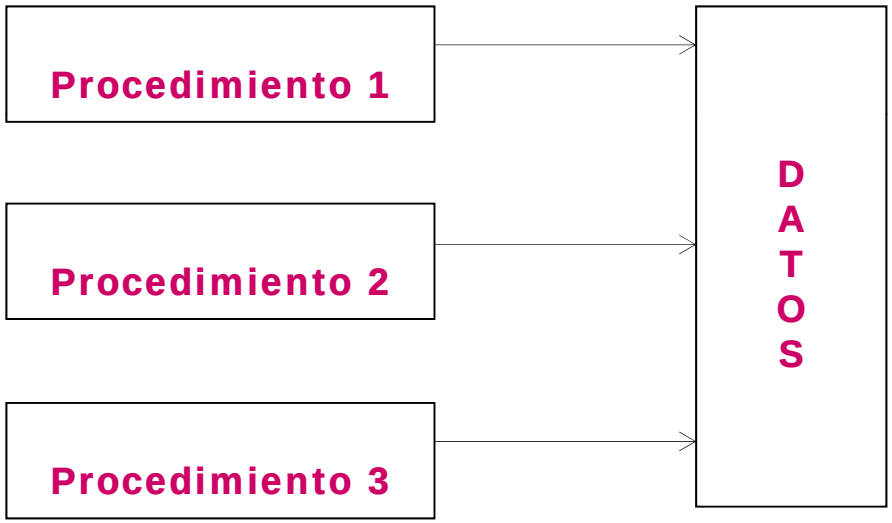
PROGRAMACIÓN ORIETADA A OBJETOS

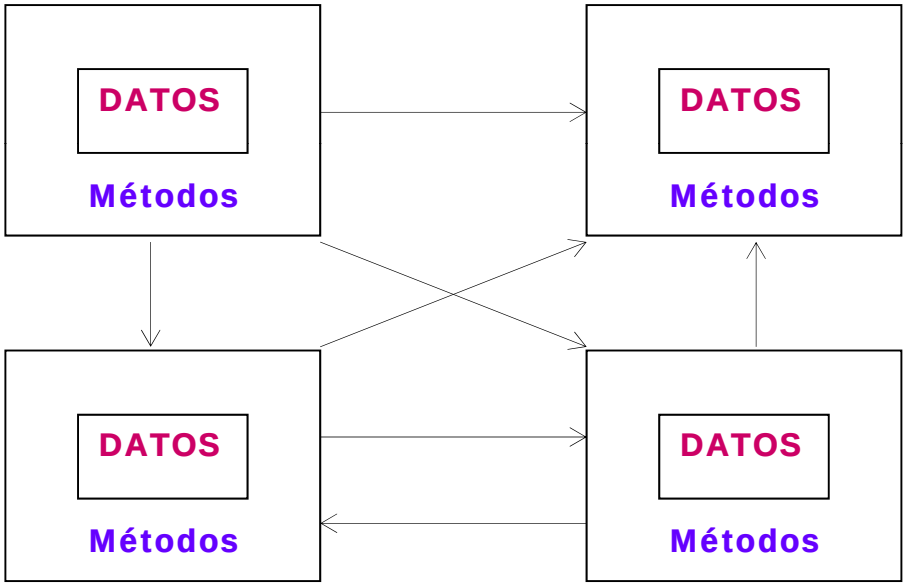
Introducción

- **Paradigma:**
  - Programación procedimental
- **Nuevo Paradigma:**
  - Programación Orientada a Objetos (POO)

Programación Procedimental



Programación Orientada a Objetos



POO

- Facilita la ***creación*** de software de calidad pues sus características potencian:
  - La mantenibilidad
  - La extensión y
  - La reutilización del software generado bajo este paradigma
- La POO se basa en la ***idea natural*** de un mundo lleno de ***objetos*** y que la resolución de problemas se realiza mediante el modelo de objetos

POO

- La visión de ***Objetos***:
  - Mesas
  - Sillas
  - Computadores
  - Autos
  - Cuentas bancarias
  - Partidos de fútbol
  - Perros, etc

POO

- Los objetos se pueden ***Organizar*** según su necesidad
  - **Mamíferos** : Perro, Ciervo
  - **Teléfonos** : Fijo, Celular
  - **Deportes** : Fútbol, Tenis
  - **Vehículos** : Automóvil, Camión

METODOLOGÍA

Modelado

- La base de esta tecnología es:

Modelar

Los problemas que se desea resolver

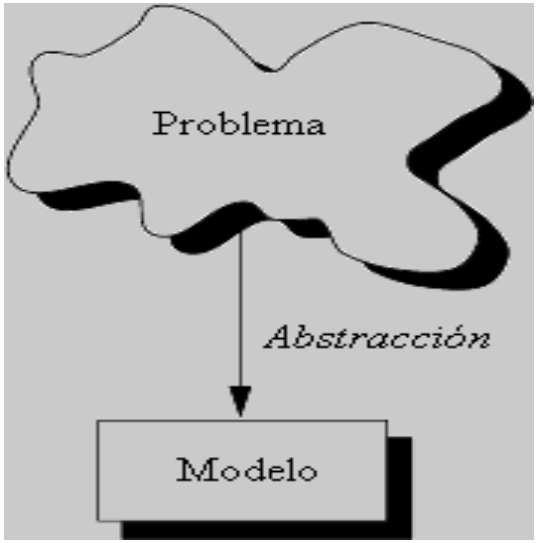
con base en

Entidades

Relevantes y tangibles en el problema

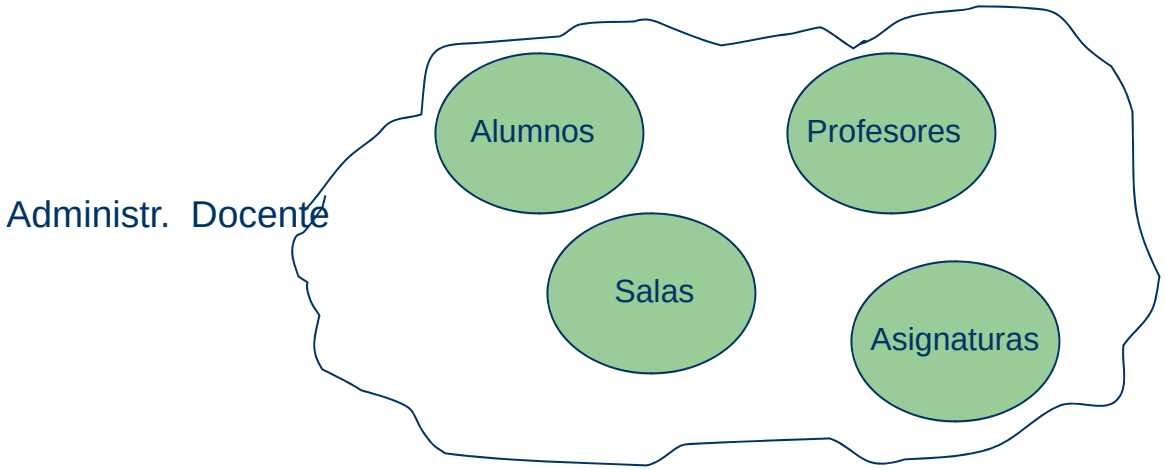
Abstracción

- El modelo define una ***perspectiva abstracta*** del problema
  - Los ***datos*** que son afectados
  - Las ***operaciones*** que se aplican sobre los datos



Ejemplo

- Para la administración docente, las entidades que participan son:



POO

- Identificar los objetos importantes
- Organizar los objetos en jerarquías
- Otorgar, a los objetos, atributos relevantes que describan sus características
- Proveer, a cada objeto, las funciones inherentes a su comportamiento

Conceptos de la POO

- La POO trata de aproximarse al modo de ***actuar del hombre*** y no al de la máquina

- Conceptos fundamentales que sustentan la POO:
  - Clase
  - Objeto
  - Instancia
  - Atributos
  - Métodos

Clase

- Una clase es como un tipo de dato creado por el usuario, que posee dos categorías de miembros:
  - ***Atributos*** (Datos) ⇒ Estado
  - ***Métodos*** (Algoritmos) ⇒ Comportamiento

Clase

- En general, es posible crear una clase a partir de cualquier objeto que esté a nuestro alrededor. Por ejemplo:
  - Persona
  - Automóvil
  - Mascota

Instancias-Objetos

- Una ***Instancia*** es una ocurrencia de la clase
- Al momento de crear un objeto se produce la instanciación
- Un ***Objeto*** es una instancia de una Clase específica

Instancias-Objetos

EJEMPLO:

Suponer que existe la clase **Vehículo**

- El auto con placas WF-4631 es una instancia de la clase **Vehículo**, o sea, un objeto de esa clase
-
- La camioneta con placas ZT-9283 es otra instancia de la clase **Vehículo**

Atributos

- Son los datos que caracterizan a los objetos de una clase y determinan el estado de un objeto

  - Marca
  - Año
  - Color
  - Placas, etc.

Métodos

- Representan todas aquellas ***acciones*** que se pueden realizar ***sobre un objeto*** de cierta clase
- En la implementación, estos métodos son segmentos de código en la forma de funciones
- La clase **Vehículo** puede incluir los métodos:
  - Encender
  - Acelerar
  - Virar
  - Frenar

Principios de la POO

- Propiedades, que lo llevan a ser ***un estilo de desarrollo*** que permite crear ***código re-utilizable***

  - Encapsulamiento
  - Herencia
  - Polimorfismo

ENCAPSULAMIENTO

Abstracción de Datos

Encapsulamiento

- Proceso por el que se ocultan:
  - Las estructuras de datos
  - Los detalles de la implementación
- Permite considerar a los objetos como "cajas negras", evitando que otros objetos accedan a detalles que NO LES INTERESA
- Una vez creada la clase, las funciones usuarias no requieren conocer los detalles de su implementación

Encapsulamiento

- Toda clase tiene un conjunto de ***atributos*** y ***métodos*** asociados a ella
- Todos ellos están ***encapsulados*** o contenidos dentro de la misma clase, de manera que son ***miembros*** de dicha clase
- Esos métodos y atributos pueden ser utilizados por ***otras*** clases ***sólo si*** la clase que los encapsula les brinda los ***permisos*** necesarios para ello

Encapsulamiento

Atributos de una Cuenta Corriente:

- Número
- Saldo

¿Cómo se almacenan estos datos?

Métodos:

- Depositar
- Girar
- Conocer el saldo

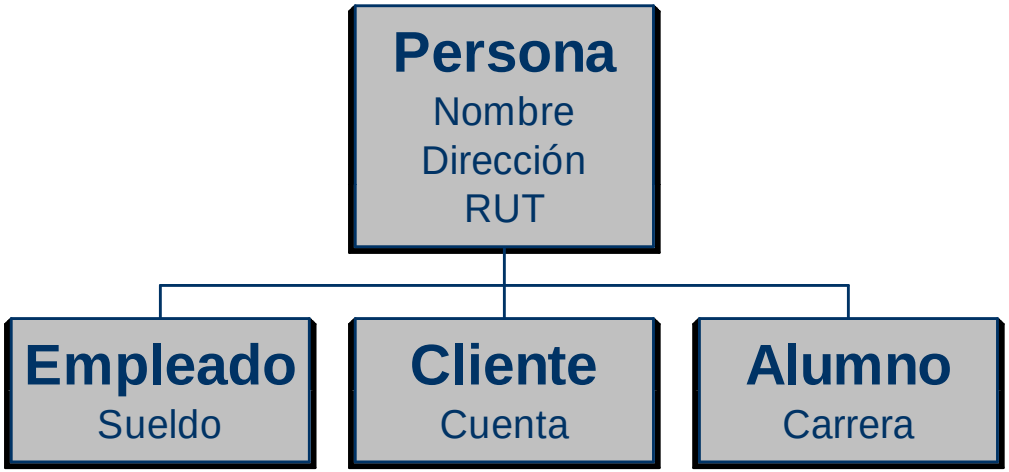
HERENCIA

Organización jerárquica

Herencia

- Permite reutilizar código creando ***nuevas*** clases a partir de las existentes (construidas y depuradas)
- Compromete una relación de jerarquía (**es-un**)
- Una nueva clase se generará agregando atributos y/o código a una clase existente
- Una clase (derivada) ***puede heredar*** de otra clase (base):
  - Atributos y
  - Métodos

Herencia


Persona
Nombre
Dirección
RUT
Empleado
Sueldo
Cliente
Cuenta
Alumno
Carrera

POLIMORFISMO

Polimorfismo

- Capacidad que permite a dos ***clases diferentes*** responder de forma distinta a un ***mismo mensaje***
- Esto significa que dos clases que tengan un método con el mismo nombre y que respondan al mismo tipo de mensaje (es decir, que reciban los mismo parámetros), ejecutarán acciones distintas

Polimorfismo

Ejemplo 1:

Al presionar el acelerador esperamos que aumente la velocidad del auto, independiente de si se tiene un:

- Motor con carburador
- Motor con inyección electrónica

Polimorfismo

Ejemplo 2:

Si se tienen las clases **Entero** y **Char**, ambas responderán de manera distinta al mensaje **"Sucesor"**

OBJETOS	MENSAJE Sucesor	RESULTADOS
'A'	→	'B'
3	→	4

OBJETOS

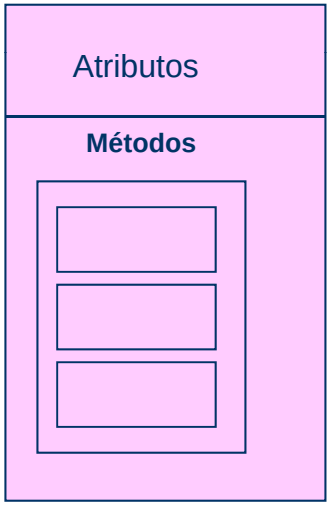
Objetos

- En la **POO** se dice que un objeto: "integra datos y algoritmos"
- En la **programación estructurada**, variables y funciones están separadas

POO		Procedural
Clase	→	Tipo
Objeto	→	Variable
Instanciación de objetos	→	Declaración de variables

Objetos

- Cada objeto ***es responsable*** de inicializarse y destruirse en forma correcta
- Un ***objeto*** consta de:
  - Tiempo de vida
  - Estado
  - Comportamiento



Tiempo de vida de un objeto

- La duración de un objeto en un programa siempre está limitada en el tiempo
- La mayoría de los objetos sólo existen durante una parte de la ejecución del programa
- Los objetos son creados mediante un mecanismo denominado ***instanciación***
- Los objetos dejan de existir cuando son ***destruidos***

Estado de un objeto

- Queda definido por sus ***atributos***
- Con él se definen las ***propiedades del objeto,*** y el estado en que se encuentra en un momento determinado de su existencia

Comportamiento de un objeto

- Queda definido por los ***métodos***
- Los prototipos de los ***métodos*** definidos en la interfaz de una clase permiten a otros objetos, que forman parte de la aplicación, interactuar con los objetos de esa clase

CLASES

Clases

- Las clases ***son abstracciones*** que representan a un conjunto de objetos con un:
  - Comportamiento e
  - Interfaz común
- Es la ***implementación*** de ***un tipo de dato*** (considerando los objetos como instancias de las clases)

Clases

- Permiten definir y representar colecciones de objetos
- Proveen un ***modelo*** para la creación de objetos
- Los elementos que componen la clase son

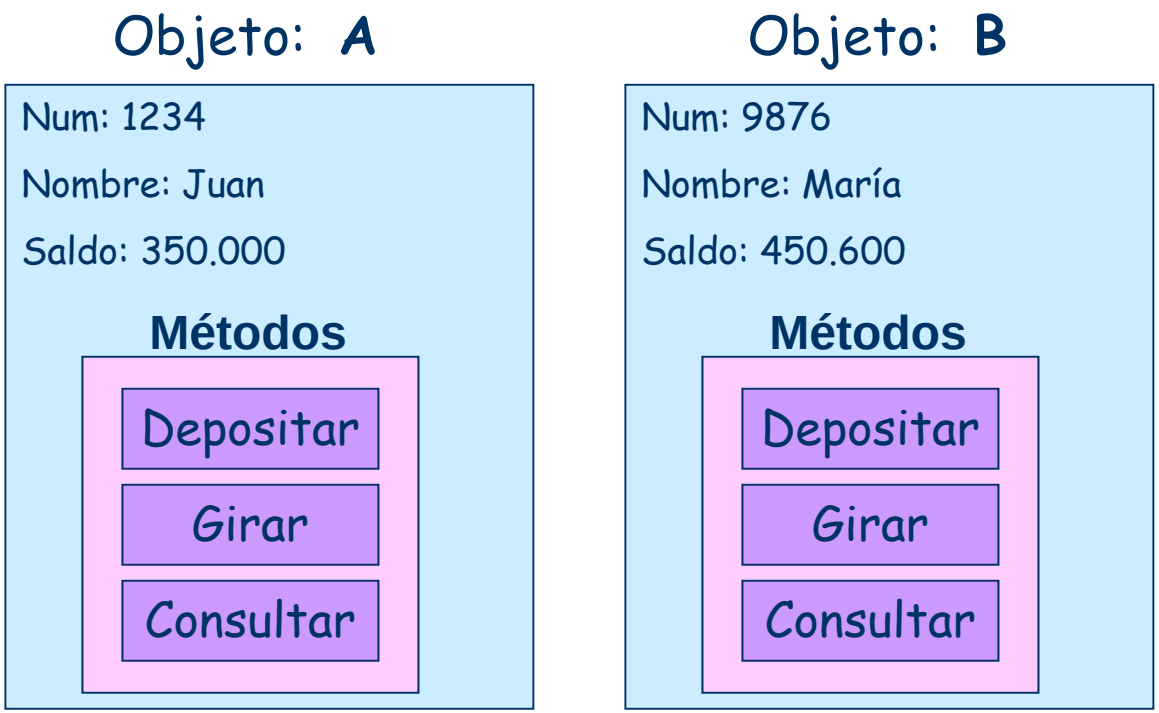
Atributos	Representan el **estado** de un objeto (variables de ejemplar)
Métodos	Representan el **comportamiento** de un objeto (funciones miembro)

Ejemplo de instancia de objetos

- **Clase:** Cuenta corriente
  - Atributos:
    - ✓ Número
    - ✓ Nombre
    - ✓ Saldo
  - Métodos:
    - ✓ Depositar
    - ✓ Girar
    - ✓ Consultar saldo

Ejemplo de instancia de objetos

- **Clase:** Cuenta corriente
- **Instanciación:** **Cuenta Corriente** A, B



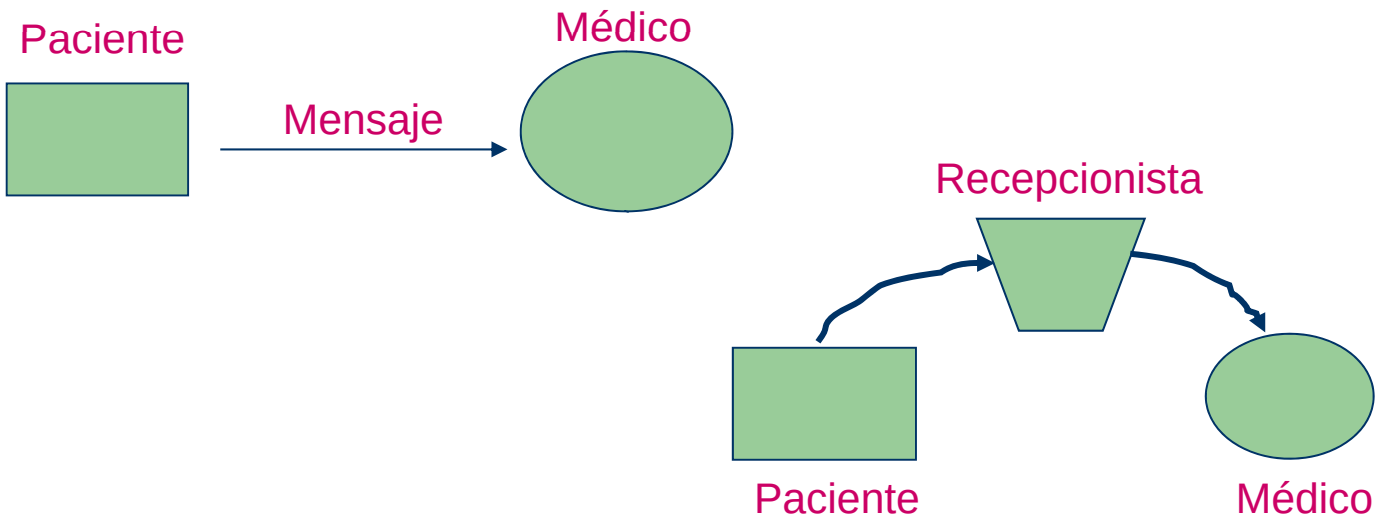
MENSAJES

Mensaje

- Mecanismo por el cual ***se solicita*** una acción sobre el objeto
- Un programa en ejecución es una colección de objetos que se crean, ***interactúan*** y se detruyen
- La ***interacción*** se basa en ***mensajes*** que son enviados de un objeto a otro, de modo que el emisor le pide al receptor la ejecución de un método

Mensajes

- Un objeto invoca un método como una reacción al recibir un mensaje
- La interpretación del mensaje dependerá del receptor



MÉTODOS

Métodos

- Un método es una función miembro de una clase
- Establece el comportamiento del objeto
- Opera directamente sobre el objeto que lo invocó
- Recibe, como ***parámetro implícito***, el objeto que lo invocó

Métodos

- Si el método requiere otros objetos de la clase, éstos deberán ser pasados como parámetros explícitos y el método sólo podrá acceder en forma indirecta a estos objetos

- Ejemplos:
  - Depositar
  - Girar
  - Consultar

RESUMEN

Mensajes y métodos

- Un objeto (agente emisor) envía un mensaje a otro objeto (agente receptor)
- El mensaje tiene codificada la petición de una acción
- El mensaje incluye la información (argumentos) necesaria para satisfacer la petición
- Si el receptor acepta el mensaje, acepta la responsabilidad de ejecutar la acción indicada
- En respuesta a un mensaje, el receptor ejecuta un método para satisfacer la petición

Clases y ejemplares

- Todos los objetos son ejemplares de una clase
- La clase del receptor determina el método que se activa como respuesta a un mensaje
- Todos los objetos de una clase usan el mismo método en respuesta a mensajes similares

Clases y métodos

- Los objetos son ejemplos de TAD's
- Un TAD tiene dos caras: una exterior, la que ve el usuario, y una interior, la que sólo ve el programador
- El usuario ve nada más que un conjunto de operaciones que definen el comportamiento de la abstracción
- El programador ve las variables de datos que se usan para mantener el estado interno del objeto
- Un ejemplar es un representante de una clase

Clases y métodos

- Una ***variable de ejemplar*** es una variable interna mantenida por un ejemplar
- Cada ejemplar tiene su propia colección de variables de ejemplar
- Las variables de ejemplar sólo son modificables por los métodos definidos en la clase
- Un objeto es la combinación de ***estado*** y ***comportamiento***

Clases y métodos

- El estado lo determinan las variables de ejemplar
- El comportamiento lo determinan los métodos
- Desde el exterior, los clientes sólo pueden ver el comportamiento de los objetos
- Desde el interior, los métodos proporcionan el comportamiento apropiado mediante las modificaciones del estado

Clases y métodos

- La interfaz describe la forma en que un objeto se conecta con el mundo
- La implementación describe cómo se logra la responsabilidad prometida en la interfaz
- Una clase se puede concebir como un registro con dos variedades de campos: datos y procedimientos
- Los datos constituyen las variables de ejemplar
- Los procedimientos constituyen los métodos