Programación Científica

Presentación

Temas a Cubrir en el Curso

- Paradigma Orientado a Objetos
- Introducción a Java
- Arquitectura de 3 Capas
  - Vista (Interface Gráfica)
  - Lógica de Negocio
  - Persistencia.

Programación Orientada a Objetos I

Conceptos del Paradigma Orientado a Objetos

Conceptos Generales Paradigma orientado a Objetos

Conceptos Generales

- Objetos y clases
- Atributos
- Mensajes y métodos
- Encapsulación y ocultamiento
- Interfaces
- Herencia de clases
- Polimorfismo
- Vinculación dinámica
- Composición de objetos

Por qué la orientación a objetos

- La técnica orientada a objetos sigue con frecuencia el mismo método que aplicamos en la resolución de problemas de la vida diaria.
- El *análisis y diseño orientado a objetos* modela el mundo en términos de objetos que tienen **estado** y **comportamiento**, y **eventos** que activan operaciones que modifican el estado de esos objetos. Los objetos interactúan de manera formal con otros objetos mediante **mensajes**.

Algunos beneficios de la orientación a objetos

- **Reutilización**. Permite la reusabilidad de código y la herencia ahorrando dinero y empleando menos tiempo de desarrollo.
- **Integridad**. Los mecanismos de encapsulación protegen sus propios componentes contra los procesos que no tengan derecho a acceder a ellos.
- La **forma de pensar** en **objetos** es más natural. El diseñador piensa en términos de objetos y no en detalles de bajo nivel.
- **Programación más sencilla**. Los programas se crean a partir de piezas pequeñas.

Otros beneficios de la tecnología OO

- Los métodos de los objetos pueden ser **polimórficos**, es decir, tienen la habilidad de enviar un mismo mensaje a objetos de clases diferentes, se "comportan" de distintas maneras.
- Es más **sencillo modificar código existente**, cada clase efectúa sus funciones independientemente de las demás.
- Se construyen clases cada vez más complejas a partir de otras más sencillas ya existentes.
- **Confiabilidad**. Generalmente las clases ya están probadas.
- Estabilidad de los modelos respecto a entidades del mundo real.

Motivos que han conducido al éxito la tecnología de objetos

- Avances en arquitectura de computadores
- Avances en lenguajes de programación (Smalltalk, C++, Java, Eiffel, ...).
- Ingeniería del software (modularidad, encapsulado de la información, proceso de desarrollo incremental)
- Los límites de la capacidad de gestionar la complejidad de los sistemas simplemente con técnicas de descomposición algorítmica

Qué es la Programación Orientada a Objetos

- Organización de los programas de manera que representan la interacción de las cosas en el mundo real.
- Un programa consta de un conjunto de objetos.
- Los objetos son abstracciones de cosas del mundo real.
- Nos interesa qué se puede hacer con los objetos más que cómo se hace.
- Cada objeto es responsable de unas tareas.
- Los objetos interactúan entre sí por medio de mensajes.
- Cada objeto es una instancia de una clase.
- Las clases se pueden organizar en una jerarquía de herencia.

La programación OO es una simulación de un modelo del universo.

Paradigma orientado a objetos

En el paradigma de la orientación a objeto, ***un sistema se concibe como un conjunto de objetos que se comunican entre si mediante*** mensajes.

Objetos + Mensajes = Programa.

Mediante este modelo se construyen más fácilmente sistemas complejos a partir de componentes individuales.

¿Qué es un Objeto?

- Las personas tenemos una idea clara de lo que es un objeto: conceptos adquiridos que nos permiten sentir y razonar acerca de las cosas del mundo. Un **objeto** podría ser ***real*** *o* ***abstracto***, por ejemplo una *organización*, una *factura*, una *figura* en un graficador, una *pantalla* de *usuario*, un *avión*, un *vuelo* de *avión*, una *reservación aérea*.

¿Qué es un Objeto?

- Informalmente, un objeto representa una entidad del mundo real
- **Entidades Físicas**
  - (Ej.: Vehículo, Casa, Producto)
- **Entidades Conceptuales**
  - (Ej.: Proceso Químico, Transacción Bancaria)
- **Entidades de Software**
  - (Ej.: Lista Enlazada, Interfaz Gráfica)

¿Qué es un Objeto?

- **Definición Formal (Rumbaugh):**
  - "**Un objeto es un concepto, abstracción o cosa con un significado y límites claros en el problema en cuestión**"
- **Un objeto posee (Booch):**
  - Estado
  - Comportamiento
  - Identidad

Un objeto posee Estado

Lo que el objeto sabe

- El estado de un objeto es una de las posibles condiciones en que el objeto puede existir
- El estado normalmente cambia en el transcurso del tiempo
- El estado de un objeto es implementado por un conjunto de propiedades (atributos), además de las conexiones que puede tener con otros objetos

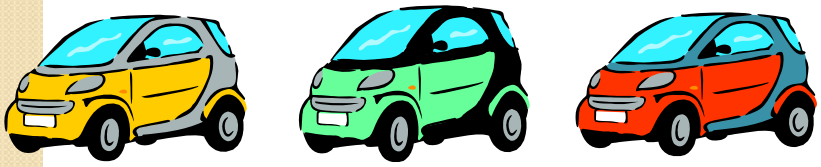
Un objeto posee Comportamiento

Lo que el objeto puede hacer

- El comportamiento de un objeto determina cómo éste actúa y reacciona frente a las peticiones de otros objetos
- Es modelado por un conjunto de mensajes a los que el objeto puede responder (operaciones que puede realizar)
- Se implementa mediante métodos

Un objeto posee Identidad

- Cada objeto tiene una identidad única, incluso si su estado es idéntico al de otro objeto

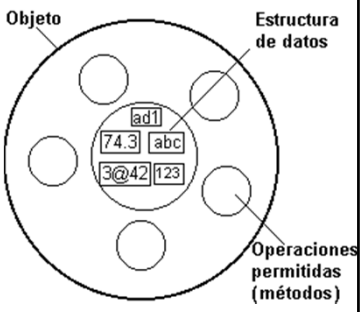


¿Qué es una Clase?

- Una clase es una descripción de un grupo de objetos con:
  - Propiedades en común (atributos)
  - Comportamiento similar (operaciones)
  - La misma forma de relacionarse con otros objetos (relaciones)
  - Una semántica en común (significan lo mismo)
- Una clase es una abstracción que:
  - Enfatiza las características relevantes
  - Suprime otras características (simplificación)

Software orientado a 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.



Programación orientada a objetos

- *"Es un método de implementación en el que los programas se organizan* ***como colecciones cooperativas de objetos****, cada uno de los cuales representa una instancia de alguna clase, y cuyas clases son, todas ellas, miembros de una jerarquía de clases unidas mediante relaciones de herencia".*
- La POO:
  - Utiliza ***objetos*** no algoritmos, como sus bloques lógicos de construcción fundamentales.
  - **Cada objeto es una *instancia* de alguna *clase*.**
  - Las clases se relacionan unas con otras por medio de relaciones de *herencia y asociaciones*.
- Sí en un programa falta alguno de estos elementos no es OO.

Pilares de la Orientación a Objetos

Tipos de datos abstractos

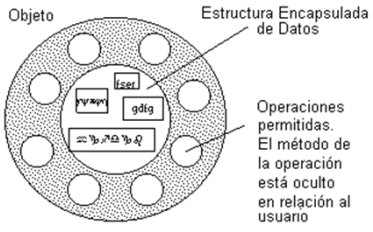
- Los tipos de datos abstractos (**TDA**) permiten describir una estructura de datos en función a las operaciones que pueden efectuar, dejando a un lado su implementación.
- Los **TDA** mezclan estructuras de datos junto a una serie de operaciones de manipulación. Incluyen una interfaz pública, que es lo que verá el usuario, y una implementación (algoritmos de operaciones sobre las estructuras de datos y su representación en un lenguaje de programación), que el usuario no tiene necesariamente que conocer para manipular correctamente los tipos de datos abstractos.
- Los **objetos** están representados por medio de los **TDA** y se caracterizan por el encapsulamiento.

Abstracción

- Es el conocimiento que se tiene de una cosa prescindiendo de las demás que están con ella.
- La abstracción localiza y oculta los detalles de un modelo o diseño para generar y manipular objetos.
- Conocemos un objeto viéndolo, sabemos qué es sin necesidad de ver su interior, su implementación o su forma de construcción.

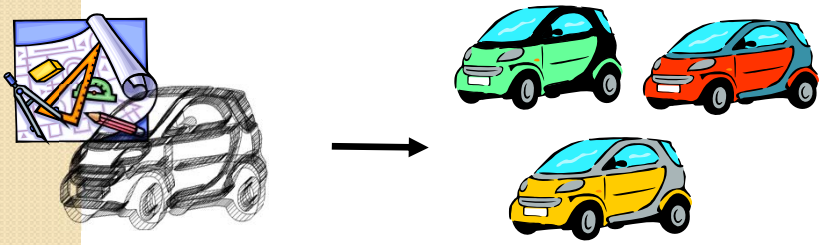
Encapsulación

- El **encapsulado** es el resultado de ocultar los detalles de implementación (estado interno) de un objeto respecto de su usuario.
- El empaque conjunto de datos y métodos se llama encapsulación.
- El objeto esconde (protege) 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 de los datos de un objeto.
- Esto facilita el cambio (de implementación), mejora la modularidad.



Objetos y Clases

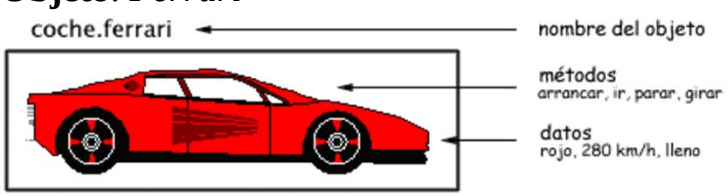
- Una clase es una definición abstracta de un objeto
  - Define la estructura y el comportamiento compartidos por los objetos
  - Sirve como modelo para la creación de objetos
- Los objetos pueden ser agrupados en clases



Ejemplo de clases y objetos

- **Clase**: *Coche*

- **Objeto**: *Ferrari*



Herencia

- La ***herencia*** es un mecanismo que permite la definición de una clase a partir de la definición de otra ya existente.
- Es la característica clave de los sistemas orientados a objeto para propiciar la reusabilidad.
- Una subclase puede ***heredar*** la estructura de datos y los métodos, o algunos métodos, de su superclase. También puede tener métodos y tipos de datos propios.

Polimorfismo

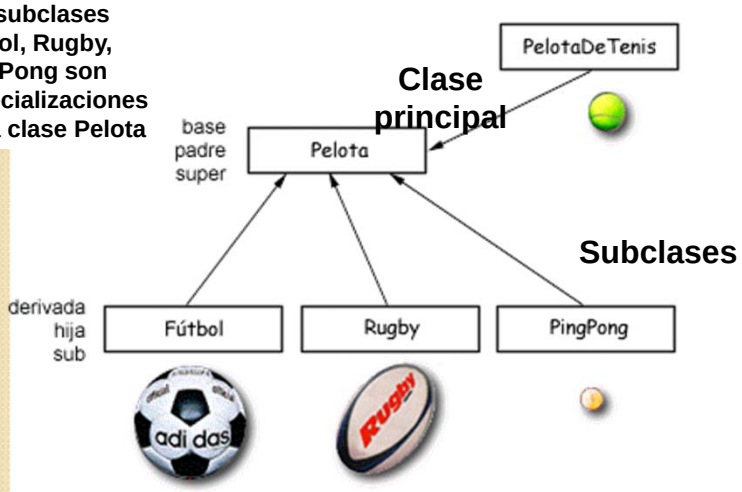
- Polimorfismo se aplica a métodos que adoptan varias formas de implementación según el tipo de objeto, pero cumple siempre el mismo objetivo.
- Los métodos son polimórficos si se aplican a objetos de distintas clases para conseguir el mismo significado semántico. Por ejemplo, *lanzar* puede ser implementado tanto para un objeto **Tenis** como para el objeto **Béisbol o Fútbol**.
- Una de las ventajas del polimorfismo es que se puede hacer una solicitud de una operación sin conocer el método que será llamado, es decir, existe un **enlace tardío** entre el mensaje y el método.
  - Método **área**, en una clase *Figura,* y subclases *Cuadrado* y *Rectángulo.*

Superclases, clases y subclases

- Una **clase** de alto nivel puede especializarse en clases de bajo nivel. Es decir, un **clase** puede tener **subclases**.
- Por ejemplo, una clase ***Persona*** puede tener subclases ***Estudiante*** y ***Empleado***. A su vez, la clase **Estudiante** puede tener como subclase a **Estudiante de pregrado** y **Estudiante de postgrado**, mientras que **Empleado** puede tener como subclase a **Académico** y **Administrativo**.
- Existe de este modo una jerarquía de clases y subclases.

Clases y subclases

Las subclases Fútbol, Rugby, PingPong son especializaciones de la clase Pelota



Métodos

- Los métodos 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 o procedimientos que pueden ser llamadas dentro de una clase o por otras clases.

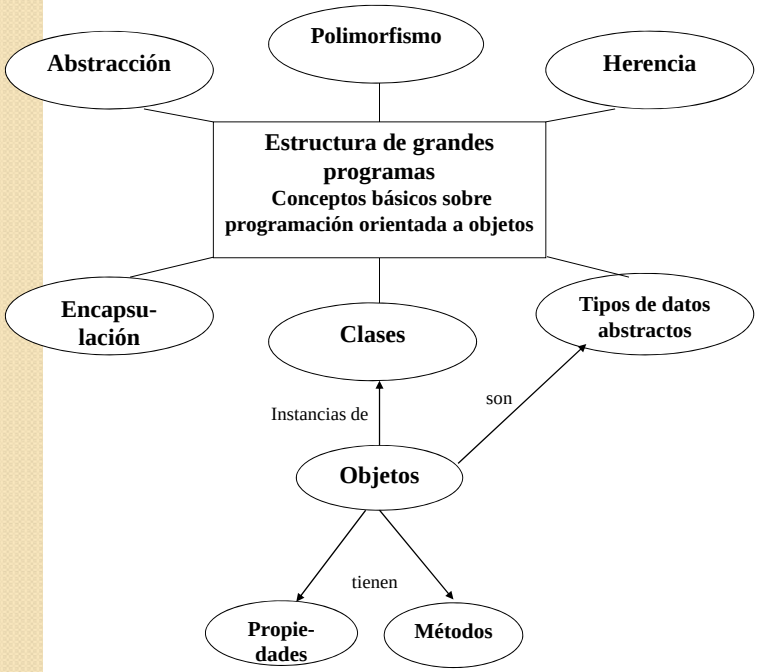
Mensajes y métodos

En la programación orientada de objetos, la acción se inicia mediante la transmisión de un *mensaje* a un agente (*un objeto*) responsable de la acción.

El mensaje tiene codificada la petición de una acción y se acompaña de cualquier información adicional (argumentos) necesaria para llevar a cabo la petición. El *receptor* es el agente al cual se envía el mensaje. Si el receptor acepta el mensaje, acepta la responsabilidad de llevar a cabo la acción indicada. En respuesta a un mensaje, el receptor ejecutará algún *método* para satisfacer la petición.

Resumen de conceptos de POO

- Un programa orientado a objetos es un conjunto de clases que describe el comportamiento de los objetos del sistema
- Los objetos se comunican mediante mensajes
- Cada objeto tiene su propio estado, que consta de otros objetos
- Cada objeto es un ejemplar de una clase (agrupación de objetos)
- Todos los objetos que son ejemplares de una misma clase pueden realizar las mismas acciones
- Las clases están organizadas en una jerarquía de herencia



Programación Científica

Técnicas para identificar Objetos

Identificación de Objetos

Conceptos - Estrategias para identificarlos

Obtención de los Objetos a partir de la lista de categorías:

Consiste en preparar un lista de conceptos idóneos a partir de una agrupación de categorías de conceptos.

- Objetos físicos o tangibles.
- Especificaciones, diseños o descripciones de cosas.
- Lugares.
- Transacciones.
- Línea o renglón de elemento de transacción.
- Papeles de las personas.
- Contenedores de otras cosas.
- Cosas dentro de un contenedor.
- Eventos.
- Etc.

Modelo Conceptual - Conceptos

Conceptos - Estrategias para identificarlos

Obtención de los conceptos a partir de la identificación de Frases Nominales.

Consiste en identificar las Frases Nominales en la descripción textual del dominio.

Descripción del Dominio

La red permite a los clientes del banco que posean una cuenta que operen con la misma a través de un cajero automático. Para poder realizar las operaciones permitidas (extracción, deposito y consulta de saldo), el cliente debe insertar la tarjeta en el cajero, quien registrara los datos de la misma en forma temporal, hasta que el cliente decida terminar las operaciones y solicita al cliente el ingreso de la clave, la cual es almacenada en forma temporal también.
Cuando el cajero recibe la orden de realizar una de las operaciones, este interactuara con la red, que es la que conoce las cuentas. La red buscara la cuenta correspondiente y realizará la operación, si la clave ingresada es la correcta.
La operación de extracción implica decrementar el saldo en un monto determinado de una cuenta, el deposito incrementa el saldo de una cuenta, y consultar el saldo es poder conocer cual es el saldo disponible de la cuenta.

Identificar Objetos

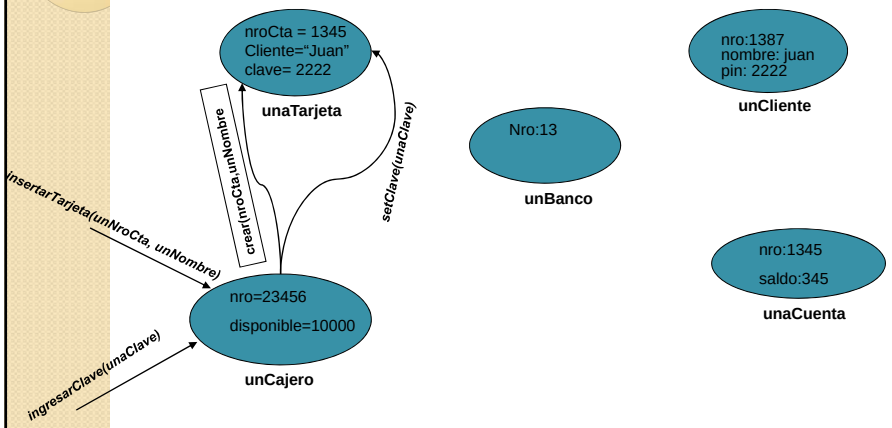
La **red** permite a los **clientes** del **banco** que posean una **cuenta** que operen con la misma a través de un **cajero automático**. Para poder realizar las operaciones permitidas (extracción, deposito y consulta de saldo), el **cliente** debe insertar la **tarjeta** en el **cajero**, quien registrará los datos de la misma en forma temporal, hasta que el **cliente** decida terminar las operaciones , también solicita al cliente el ingreso de la **clave**, la cual es almacenada en forma temporal . Cada **cuenta** tiene un **titular** que es un **cliente** del banco.
Cuando el **cajero** recibe la orden de realizar una de las operaciones, este interactuara con la **red**, que es la que conoce las **cuentas**. La **red** buscará la **cuenta** correspondiente y realizará la operación, si la **clave** ingresada es la correcta.
La operación de extracción implica decrementar el **saldo** en un **monto** determinado de una **cuenta**, el deposito incrementa el **saldo** de una **cuenta**, y consultar el saldo es poder conocer cual es el **saldo** disponible de la **cuenta**.

Objetos Candidatos	*Opereraciones*
Red	*extraer*
Cliente	*Depositar*
Banco	*consultarSaldo*
Cuenta	*insertarTerjeta*
Cajero Automático	*ingresarPin*
Tarjeta	
Clave	
Titular	
Saldo	
Monto	

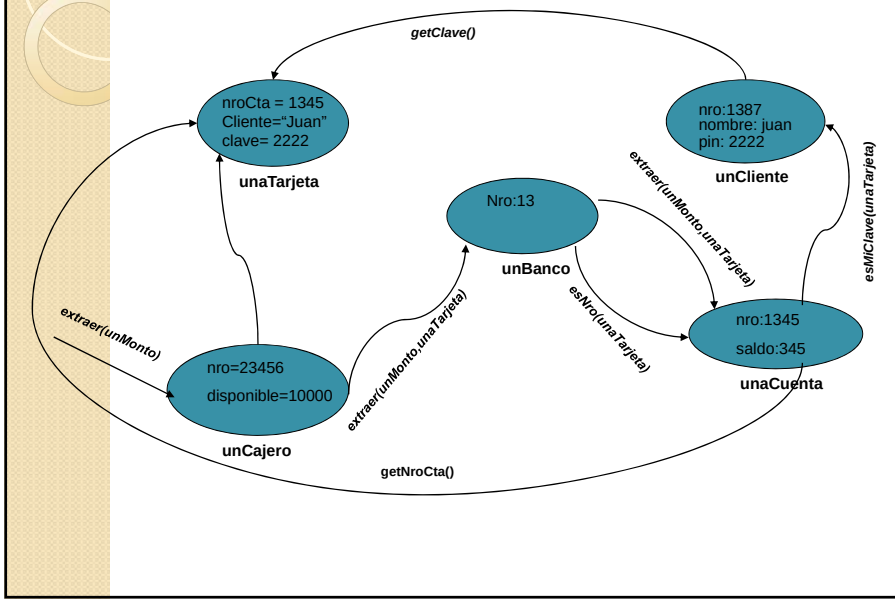
Identificar Responsabilidades

- Guía:
- - Mantener el comportamiento relacionado a cierta información en la misma clase. **Si un objeto mantiene determinada información, las operaciones que se realizan sobre dicha información deben agregarse a la misma clase**.
- - Mantener la información acerca de una misma cosa en una misma clase o clases muy relacionadas.
- - Distribuir responsabilidades compuestas entre clases relacionadas .

Interacción entre Objetos – Operaciones: insertarTarjeta e ingresarClave



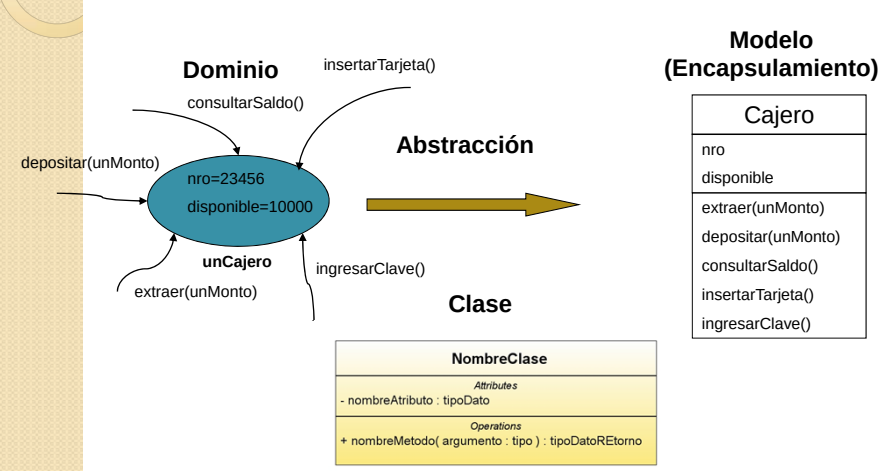
Interacción entre Objetos – Operación extraer



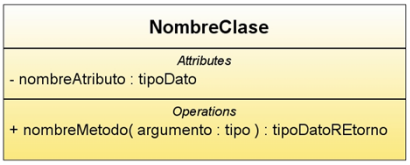
Clase

- **Es una Plantilla a partir de la cual se crean instancias. Contiene una definición del estado (atributos) y métodos (comportamiento) del objeto.**
- **Estado : Conjunto de características relevantes del objeto.**
- **Métodos: Las responsabilidades definidas para el objeto**

Definir Atributos y Métodos de las clases



Clase



Nombre de la clase comienza con mayúsculas.

Nombre de los atributos comienzan con minúsculas

Nombre de los métodos comienzan con minúsculas

Si el nombre esta compuesto por más de una palabra:

En la clase: Todas las palabras comienzan con mayúsculas

En los atributos y métodos: Todas las palabras comienzan con mayúsculas, salvo la Primera.

Descripción de la clase

Nombre: NombreDeLaClase

Atributos

tipoDato **identificador**

Métodos

tipoDatoRetorno **nombreMetodo**(tipoDato arg1,...)

"descripción de lo que debe hacer el método"

Nombre: Cuenta

Atributos

entero **nro**
real **saldo**

Métodos

void **extraer**(real unMonto,Tarjeta unaTarjeta)

"Verificar con el cliente que la clave de la trajeta se corresponda con el pin, si es asi, decrementar saldo en unMonto"