

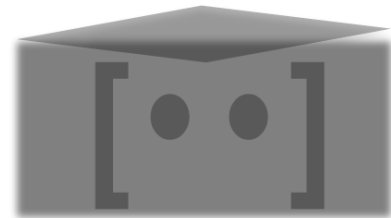
JAVA

Profesor: Nahuel Meza

Cod.Ar[••]

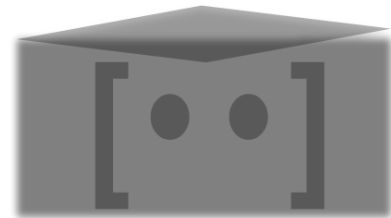
Repaso

¿Qué vimos la clase pasada?....

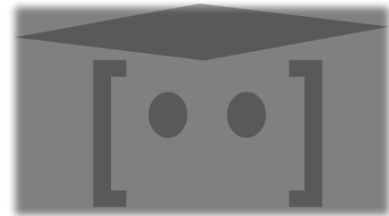


EL MOMENTO MÁS ESPERADO

P.O.O
(PROGRAMACIÓN ORIENTADA A
OBJETOS)



La Programación orientada a objetos es un **paradigma de programación** que organiza el código alrededor de **objetos** en vez de funciones y lógica.



Conceptos básicos modularidad

- Componente/especificación:
 - En palabras el comportamiento esperado del módulo.
- Interfaz:
 - tipos y operaciones visibles **fuera** del componente
- Implementación:
 - estructura de datos y funciones dentro del componente



Conceptos básicos modularidad

- Componente/especificación:
 - Función que calcula el cuadrado de un número
- Interfaz:
 - `public static int square (int x);`
- Implementación:
 - `public static int square (int x)`
`{ return x*x; }`



Propiedades importantes de P00

- Clases
- Objetos
- Encapsulación
- Herencia y Subtipado



Clases

Define el **comportamiento** de todos los objetos.

Es una plantilla que define a un objeto, los cuales son **instancias** de la clase.



Objetos

Posee datos **privados**:

- posiblemente funciones ocultas
- Variables de instancia.

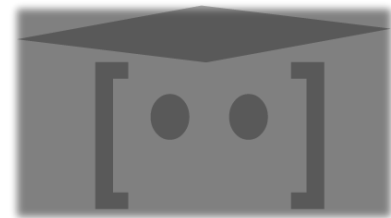
Posee operaciones **públicas**:

- métodos (funciones)
- puede tener variables publicas.



Encapsulación

Uno de los conceptos fundamentales del POO.
Define como un usuario puede usar un objeto
abstrayéndose de la lógica del programa.
Mientras que un **programador** tiene visible toda la
lógica.



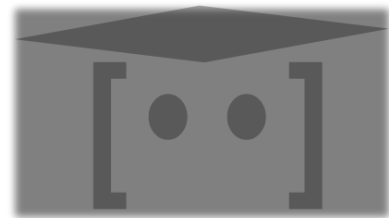
Herencia y subtipado

NO ES LO MISMO.

El **subtipado** es una relación de **interfaces**.

La **herencia** permite utilizar las **implementaciones** de una clase madre.

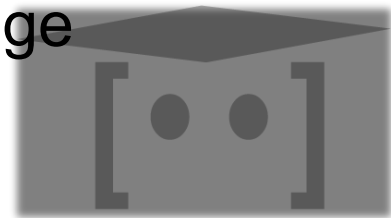
Java no tiene una diferencia entre ambas.



Visibilidad en Java

Java tiene distintos tipos de visibilidad para clases y/o métodos/campos de clase:

- `public`: Puede ser accesible desde la instancia de clase. (En caso de clases, es que puede ser instanciada)
- `private`: Sólo puede ser accedido dentro de la clase.
- `protected`: Sólo accesible dentro del package o en subclases.



Otros modificadores

- `static`: Pertenece a la **clase** no al objeto. Puede ser usado sin instanciar la clase.
- `abstract`: define un método sin cuerpo en la clase. La subclase está obligada a darle lógica.

