Programación Python

Instructor: Moisés Avalos

Programador

En esta presentación

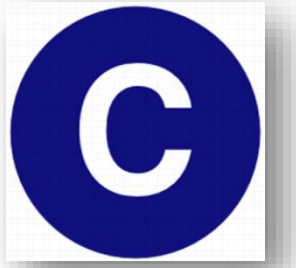
- Paradigmas de la programación
- Programación Orientada a Objetos
- Análisis y diseño orientado a objetos
- Diagrama de clases
- Identificación de clases
- Relación entre clases
  - Generalización / especialización
  - Agregación/composición
  - Asociación


python


Ruby


LISP


Haskell
pure → simple → fun


C


Java


C++


php

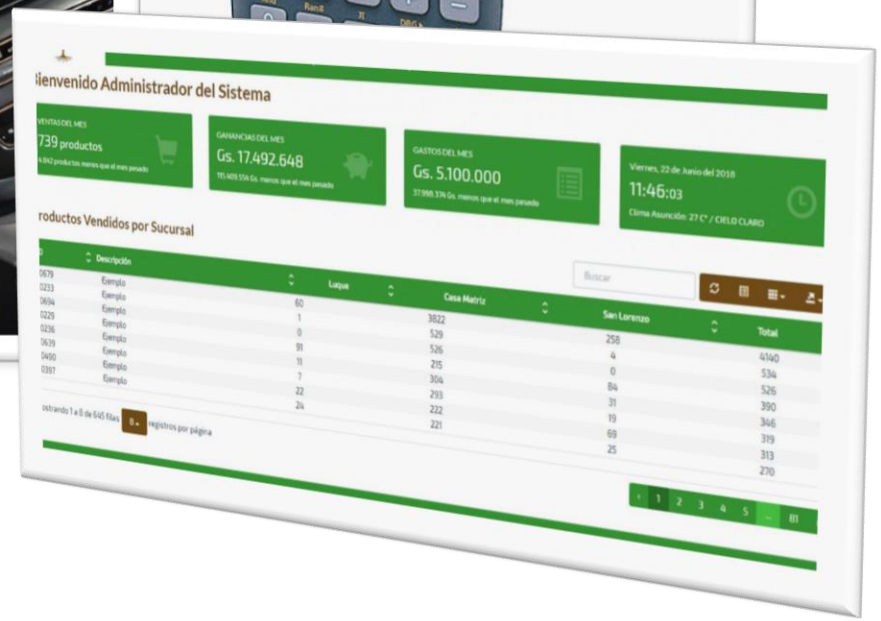

PASCAL


ADA


ProLog


CASIO
fx-82MS
S-V.P.A.M.
sin 63° 52° 41"
8.978590120E-01


UNO
ARDUINO
DIGITAL (PWM~)
POWER
ANALOG IN


ienvenido Administrador del Sistema
739 productos
Gs. 17.492.648
Gs. 5.100.000
11:46:03
roductos Vendidos por Sucursal

Paradigmas de la programación.

¿Qué es un paradigma?
Prototipo, ejemplar, modelo...

Un paradigma de programación es un estilo de programación que provee y determina la **visión que el programador tiene de la ejecución del programa.**

Hay lenguajes que soportan un paradigma y otros que soportan múltiples paradigmas.

Programación imperativa:

El código del programa está dividido en diferentes bloques llamados funciones o procedimientos.

Cada uno de estos bloques se encarga de realizar una tarea específica y puede ser llamado varias veces desde otros puntos del programa.

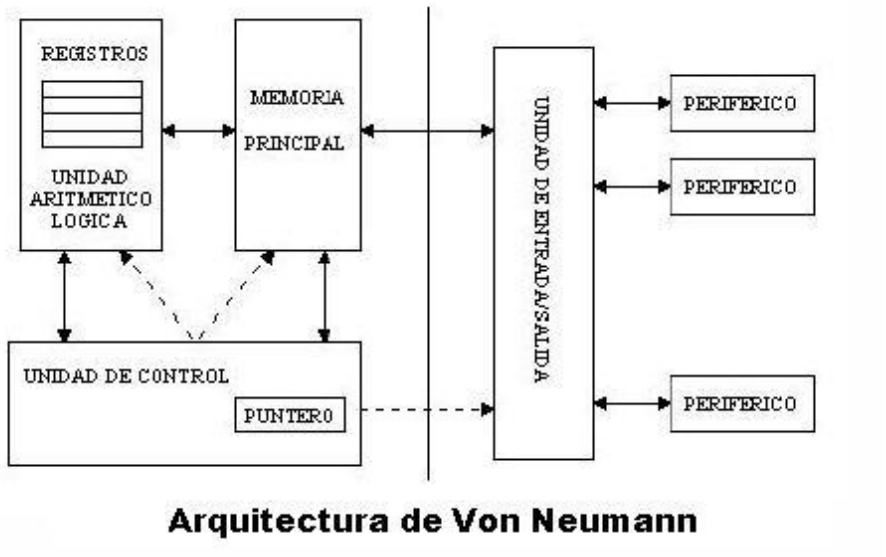
Algunos ejemplos de estos lenguajes son: Ada, Delphi, C, Pascal.

Programación declarativa o funcional:

Programación declarativa o funcional concibe la computación como una evaluación de funciones matemáticas y evita utilizar datos que vayan cambiando de valor a lo largo de la ejecución del programa.
Ejemplo: Lisp, Haskell

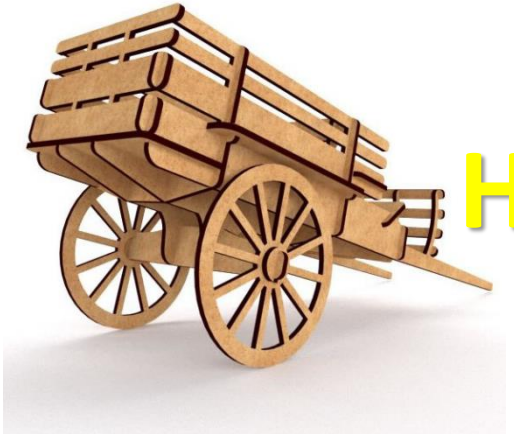
Programación lógica:

Los programas implementados suelen definir hechos, que son considerados verdaderos siempre, y reglas que no explican como se tiene que solucionar el problema, sino qué relaciones hay entre las entidades que se modelan.
Ejemplo: Prolog



Arquitectura de Von Neumann

SOFTWARE



HARDWARE

Programación paralela o concurrente:

Un programa concurrente es aquel que tiene más procesos en marcha a la vez que procesadores tiene la máquina y un programa paralelo es el que se ejecuta en varios procesadores a la vez
Ejemplo: Ada

Programación orientado a objetos:

Un programa esta formado por un conjunto de objetos independientes que interactúan entre ellos. Cada objeto contiene su propio código y sus datos. Tiene la capacidad de recibir peticiones (mensajes de otros objetos) para hacer alguna tarea, procesar datos y enviar peticiones (mensajes) a otros objetos.
Ejemplo: Java, PHP, C++

Introducción a los conceptos de la programación orientada a objetos

OBJETO

Los objetos *son/representan* cosas.
Los objetos pueden ser *simples* o *complejos*.

Objeto

Entidad que contiene los atributos que describen el estado de un objeto del mundo real y las acciones que se asocian con el objeto del mundo real.
Un objeto es designado con un nombre o un identificador.

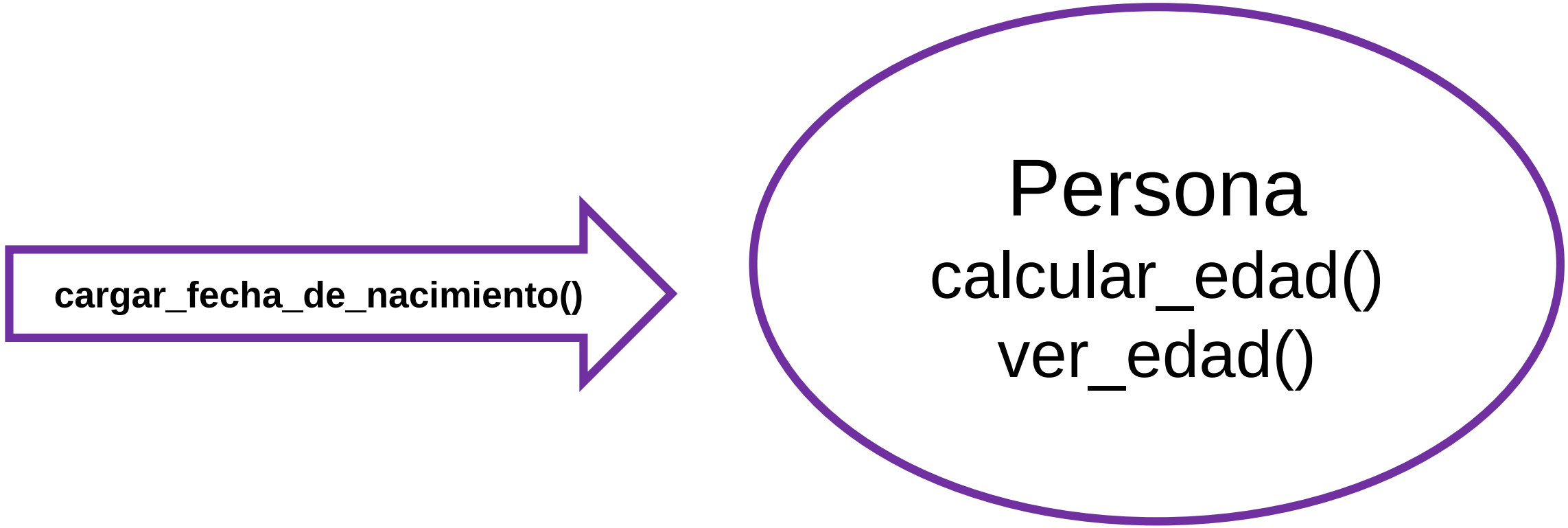
OBJETO = DATOS + OPERACIONES

Los datos deberían estar ocultos en el objeto, y las operaciones serían la interfaz del objeto con el exterior.

Un objeto es una materialización o instancia de una clase durante la ejecución del programa.

Objetos y encapsulamiento

La encapsulación se refiere a impedir el acceso a determinadas operaciones y atributos de los objetos estableciendo así que puede utilizarse desde fuera de la clase.



Clase:

Es la descripción de un conjunto de objetos.

Consta de métodos y datos que resumen las características comunes de un conjunto de objetos. Y muestra el comportamiento general de un grupo de objetos.

Una clase es un tipo de dato abstracto equipado con una implementación total o parcial.

Clase:

Los objetos son instancias (materializaciones) de clases.

Un objeto se crea mediante el envío de un mensaje de construcción a la clase y se destruye cuando recibe un mensaje de destrucción.

Clase:

Tendemos a agrupar seres o cosas con características similares en grupos o clases.

Así cuando existen una variedad de sillas podemos reconocer una silla incluso aun cuando no hayamos visto ese modelo de silla nunca.
A esto se denomina abstracción.

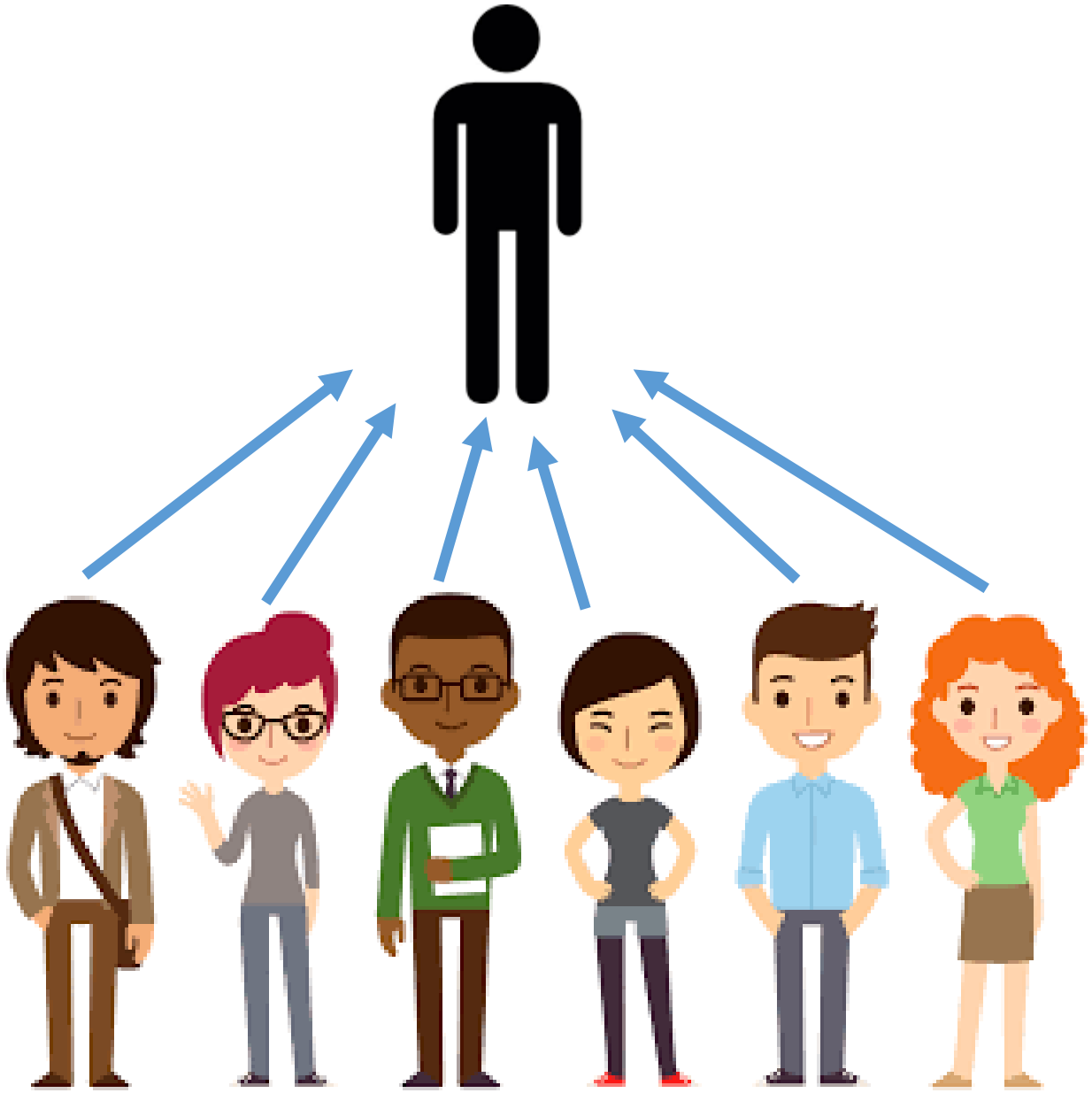
Herencia:

Es una propiedad que permite que los objetos sean creados a partir de otros ya existentes, obteniendo características (métodos y atributos) similares a los ya existentes.

Es la relación entre una clase general y otra clase más específica.
Es un mecanismo que nos permite crear clases derivadas a partir de clases denominadas base.

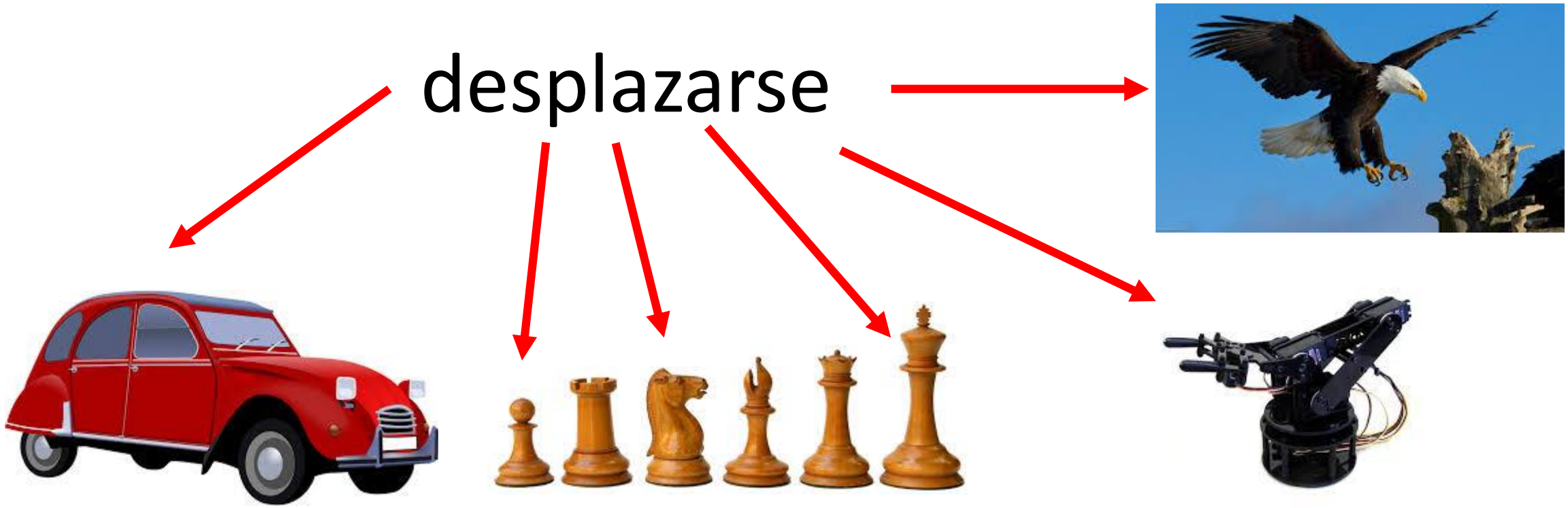
Nos permite compartir automáticameinte métodos y datos entre clases subclases y objetos.

Herencia:



Polimorfismo:

Cualidad que posee un objeto para responder de distinto modo ante el mismo mensaje.



Pilares de la programación orientada a objetos.

Encapsulamiento

Herencia

Polimorfismo

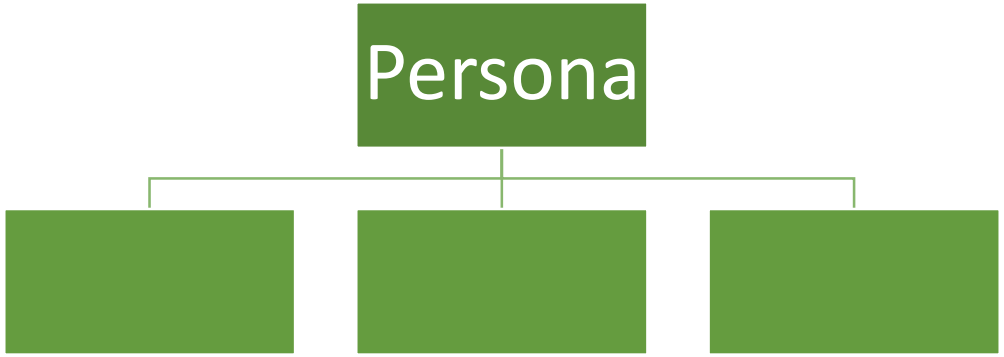
Abstracción

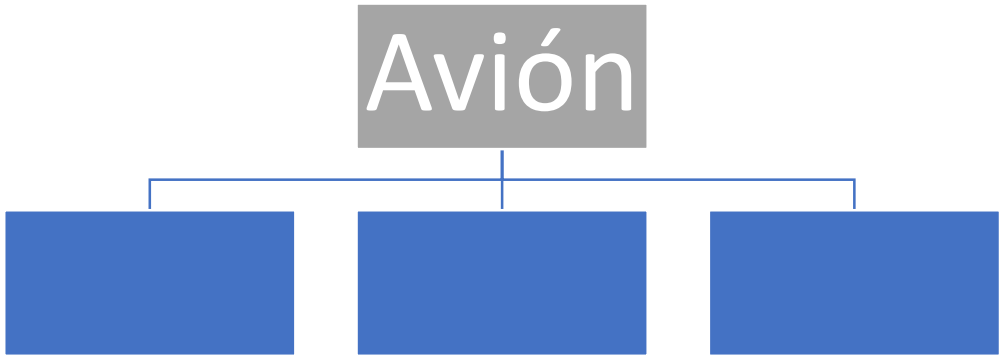
EJERCICIOS

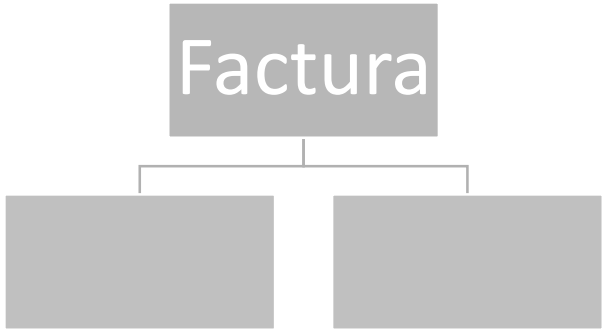
Defina tres atributos y tres operaciones sobre los objetos

- **Avión**
- **Factura**
- **Alumno**

EJERCICIOS
Herencia







EJERCICIOS
Defina operaciones polimórficas

Análisis y diseño orientado a objetos

Las preguntas clave en el diseño de un sistema son:

¿Cómo deberíamos asignar las responsabilidades a las clases de objetos?

¿Cómo deberían interaccionar los objetos?

¿Qué clases deberían hacer qué?

La asignación de responsabilidades influye fuertemente en la robustez, mantenimiento y reutilización de los componentes de software.

Análisis y diseño orientado a objetos

Durante el **análisis orientado** a objetos, se presta especial atención a encontrar y describir los objetos (o conceptos) en el dominio del problema.

Por ejemplo: en el caso del sistema de información de una biblioteca, algunos de los conceptos son Libro, Biblioteca y Socio.

Durante el **diseño orientado** a objetos, se presta especial atención a la definición de los objetos de software y en cómo colaboran para satisfacer los requisitos.

Por ejemplo: en el sistema de biblioteca, un objeto libro podría tener un atributo título y una operación para obtener un capítulo en particular.

Modelos de objetos

Una clase de objetos es una abstracción sobre un conjunto de objetos que identifica atributos comunes (como en un modelo semántico de datos) y los servicios u operaciones que son proporcionados por cada objeto.

Los objetos son entidades ejecutables que tienen atributos y servicios de la clase de objetos.

Los objetos son materializaciones de la clase de objetos, y pueden crearse muchos objetos a partir de una clase.

Generalmente, los modelos desarrollados utilizando análisis se centran en las clases de objetos y en sus relaciones.

El proceso de análisis para identificar objetos y las clases de objetos se reconoce como una de las áreas **más difíciles** del desarrollo de software orientado a objetos.

Diagrama de clases

Los diagramas de clases muestran las diferentes clases que componen un sistema y cómo se relacionan unas con otras.

Se dice que los diagramas de clases son diagramas «estáticos» porque muestran las clases, junto con sus métodos y atributos.

Además muestran las relaciones estáticas entre las clases: qué clases “conocen” a qué otras clases o qué clases “son parte” de otras clases, pero no muestran los métodos mediante los que se invocan entre ellas.

Diagrama de clases

Un diagrama de clases en Lenguaje Unificado de Modelado **(UML)** es un tipo de diagrama de estructura estática que describe la estructura de un sistema mostrando las clases del sistema, sus atributos, operaciones (o métodos), y las relaciones entre los objetos.

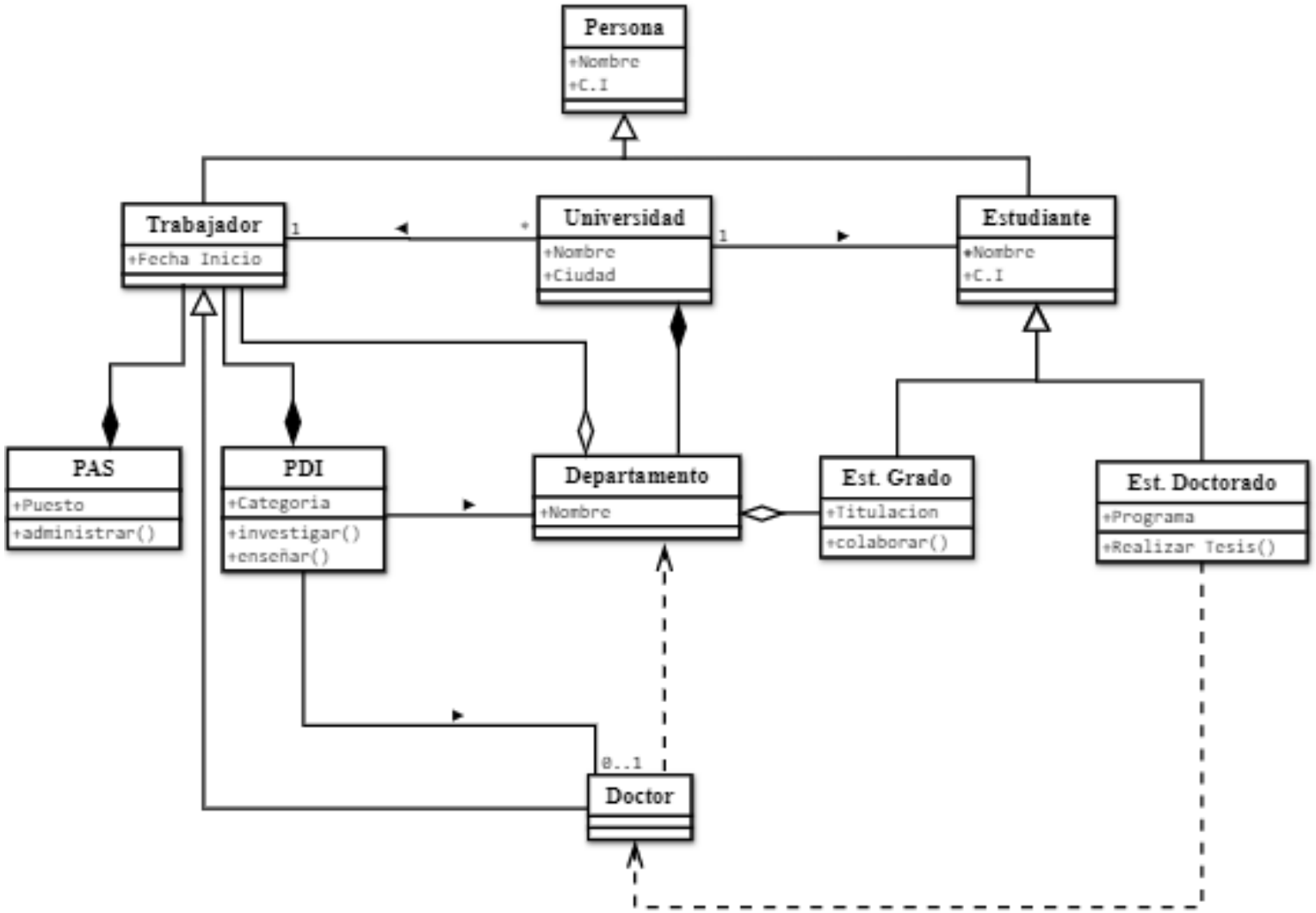
UML proporciona mecanismos para representar los miembros de la clase, como atributos y métodos, así como información adicional sobre ellos.

Diagrama de clases

Para especificar la visibilidad de un miembro de la clase (es decir, cualquier atributo o método), se coloca uno de los siguientes signos delante de ese miembro:

+	Público
-	Privado
#	Protegido
/	Derivado (se puede combinar con otro)
~	Paquete

Ejemplo de diagrama de clases de una universidad



Identificación de clase

Se podría utilizar un análisis gramatical del enunciado donde las clases y sus atributos son **sustantivos** y las operaciones o servicios son **verbos**, **aunque ésta técnica es rechazada por su extrema fragilidad**.

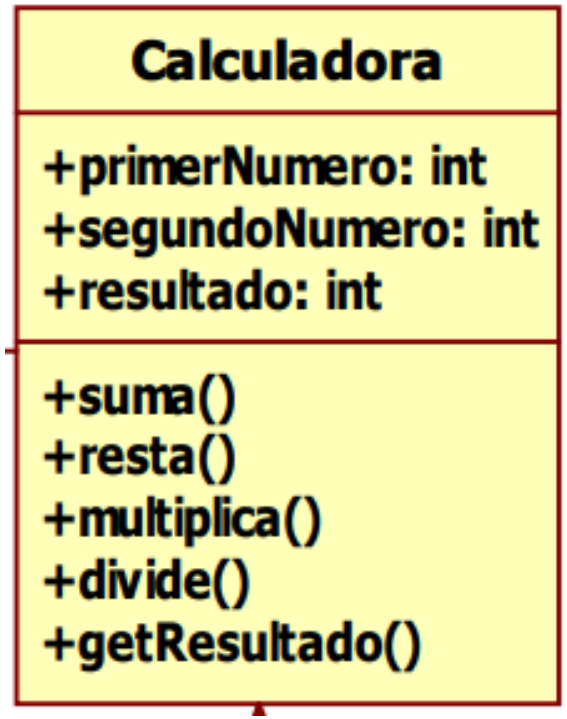
Utilizar un enfoque de comportamiento general del sistema en donde los participantes principales de dicho comportamiento se identifican como clases.

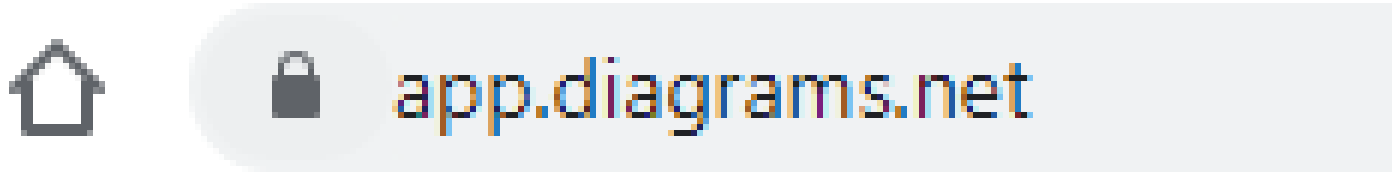
Utilizar un análisis basado en escenarios en el cual se identifican y analizan en su momento varios escenarios de la forma de utilizar el sistema.

De esta forma, por cada escenario se van identificando las clases, atributos y operaciones requeridos.

Ejercicios

Realiza un diagrama de clases y añade los atributos y métodos necesarios para implementar una calculadora





Ejercicios

Programa una Calculadora en Python