UNIVERSIDAD ADOLFO IBÁÑEZ

2022/01

Lenguajes y Paradigmas de Programación

INGENIERÍA

Discusión

Tarea: strcmpic

La solución implementada:

C:

https://repl.it/@DaniloBorquez/PeacefulGenuineMp3

Python:

https://repl.it/@DaniloBorquez/test

Estructuras en C

Tipo de datos compuesto

- Múltiples variables
- Accesibles a través de un único puntero o nombre
- Bloque contiguo de memoria (Remember sizeof!)

Sintaxis básica

```
struct tag_name {
   type member1;
   type member2;
   /* .... */
};
```

Podemos ...

... definir que esa estructura sea un nuevo tipo de datos

```
typedef struct tag_name struct_alias;
```

Uso

```
struct tag_name variable_struct;
struct_alias variable_struct;
```

Ejemplo


CÉDULA DE IDENTIDAD
REPÚBLICA DE CHILE
SERVICIO DE REGISTRO CIVIL E IDENTIFICACIÓN
APELLIDOS
FREDEZ
VIDAL
NOMBRES
MARCELA CAROLINA
NACIONALIDAD
CHILENA
SEXO
F
FECHA DE NACIMIENTO
21 FEB 1982
NÚMERO DOCUMENTO
100000001
FECHA DE EMISIÓN
1 SEP 2013
FECHA DE VENCIMIENTO
10 AGO 2023
FIRMA DEL TITULAR
RUN 12.749.625-K

Ejemplo

```
struct carnet_identidad {
    int numero;
    char dv;
    char* apellidos;
    char* nombres;
    char sexo;
    char* pais;
...
};
```

```
typedef struct
carnet_identidad ci;

ci persona;
persona.numero = 12749625;
persona.dv = 'k';
```

Objetivo

- Agrupar valores relacionados a un concepto
  - El carnet de identidad tiene:
    - Nombres
    - Apellido Paterno
    - Apellido Materno
    - Fecha de Nacimiento
    - ...

Pero...

Discusión

¿Qué pasa con los métodos asociados a esta agrupación?

Métodos

- Los puedo agrupar junto con la estructura en un módulo (.h y .c)
- PERO, la estructura debe pasarse como un parámetro del método de todas maneras

```
void calcula_digito(ci persona) {
    int rut = persona.numero;
    int suma = 0;
    int multiplicador = 1;
    while (rut != 0) {
        multiplicador++;
        if (multiplicador == 8) multiplicador
= 2;
        suma += (rut % 10) * multiplicador;
        rut = rut / 10;
    }
    suma = 11 - (suma % 11);
    if (suma == 11) persona.dv = '0';
    else if (suma == 10) persona.dv = 'k';
    else persona.dv = suma + '0';
}
```

¿Y un método para determinar si el rut es válido?
¿Y otro método que formatee el número en formato RUT?

Estructuras en C

Las estructuras en C fueron un paso que muchas personas consideraron muy útil, pero faltaba algo...

UNIVERSIDAD ADOLFO IBAÑEZ

¡Al rescate!

Programación Orientada al Objeto

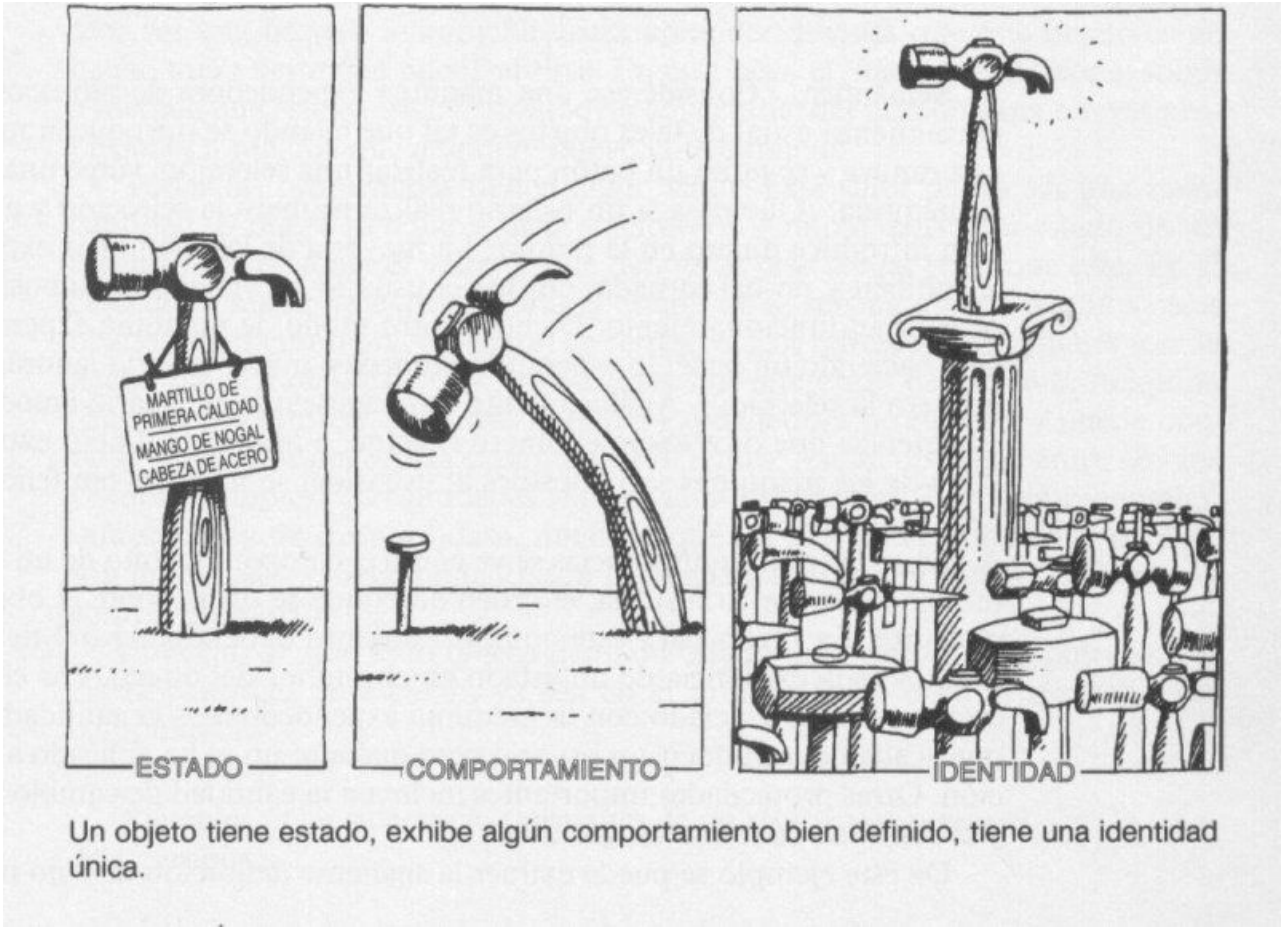
INGENIERÍA

¿Qué es la POO?

Como su nombre lo indica, es una forma de programar que se basa en objetos

¿Qué es un objeto?

Es una entidad que refleja objetos de la vida real y que está compuesto de un estado, de un comportamiento y de una identidad.



Un objeto tiene estado, exhibe algún comportamiento bien definido, tiene una identidad única.

http://www.gayatlacomulco.com/tutorials/fundamentosdeprog/t21.htm

UNIVERSIDAD ADOLFO IBAÑEZ

Historia de la Orientación a Objetos

Simula67

- Centro de cálculo Noruego
- Usado para la simulación de naves
- Agruparlos por características y métodos
- NO es orientado a objetos, pero introduce conceptos en la misma línea

SmallTalk

- Desarrollado en Xerox Park
- ¡Al principio estaba escrito en BASIC!
  - Luego fue escrito en Simula67
- El primer lenguaje puro orientado al objeto
- No planificar sino que investigar

SmallTalk

- Dos pilares fundamentales:
  - No planificar sino que investigar
    - Adaptarse al cambio
    - Desarrollar antes que los requerimientos estén refinados
  - Ocultar información
    - Limitar acceso a datos

C++

- El culpable del boom de la orientación al objeto
- Creado en AT&T labs para extender C a POO

Java

- El lenguaje más usado en la industria hoy en día
- Creado en 1996 por SUN microsystems
- Extensión de C++, pero con facilidad de uso

Conceptos Fundamentales

Conceptos primordiales

1. Encapsulamiento: agrupar TODO lo relacionado dentro de una única entidad

   Clase

   Exponer sólo lo necesario

Objeto vs Clase

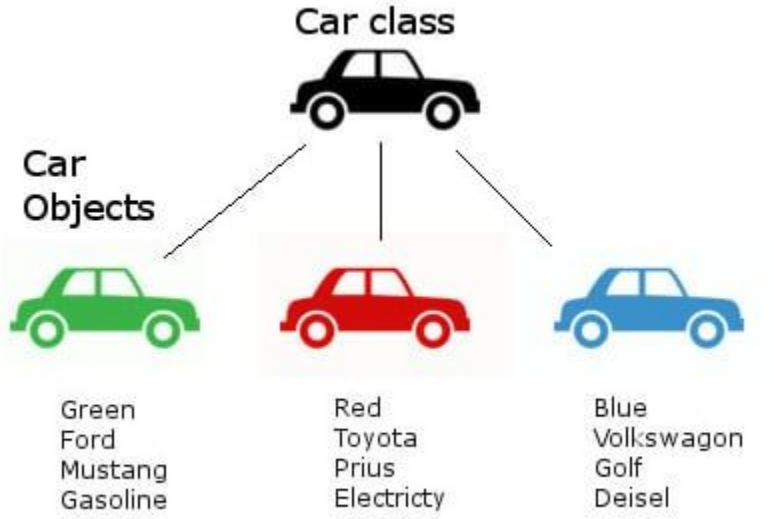
Una clase es una especificación de un objeto:

- variables
- funciones

Es sólo una representación lógica de datos.

Los objetos son INSTANCIAS de clases.

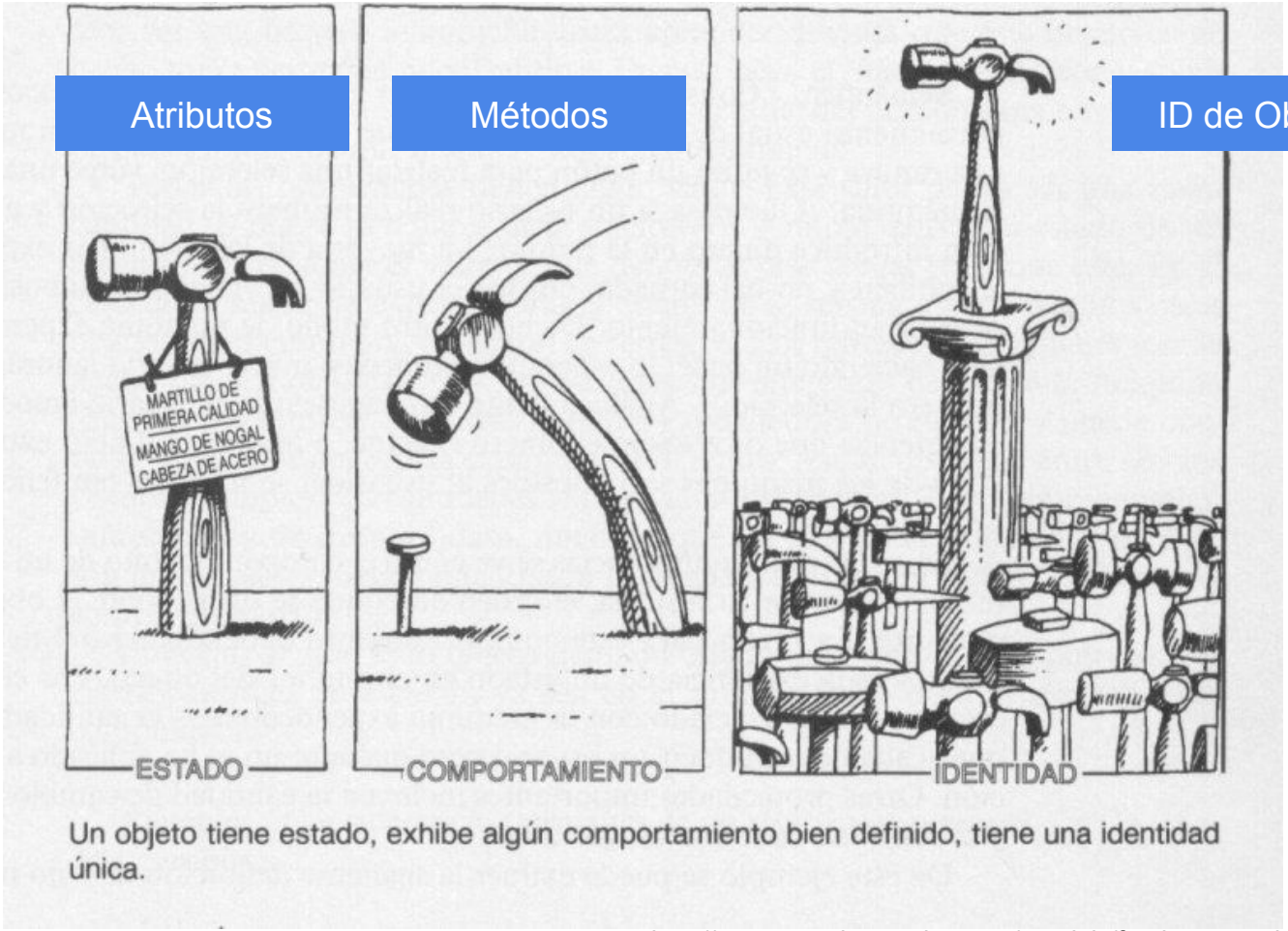
Ejemplo


Car class
Car
Objects
Green
Ford
Mustang
Gasoline
Red
Toyota
Prius
Electricty
Blue
Volkswagon
Golf
Deisel

Atributos

Métodos

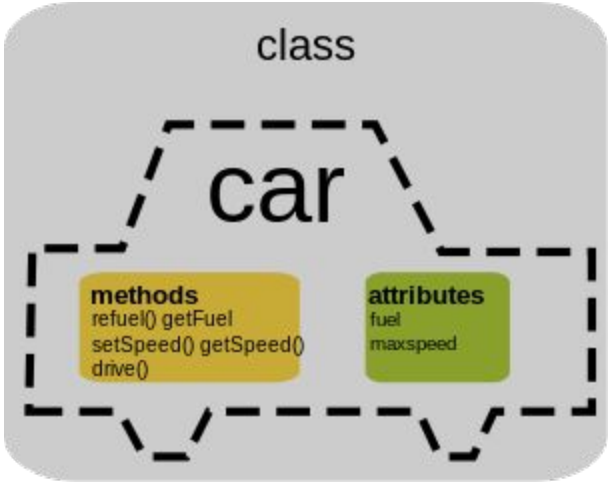
ID de Objeto

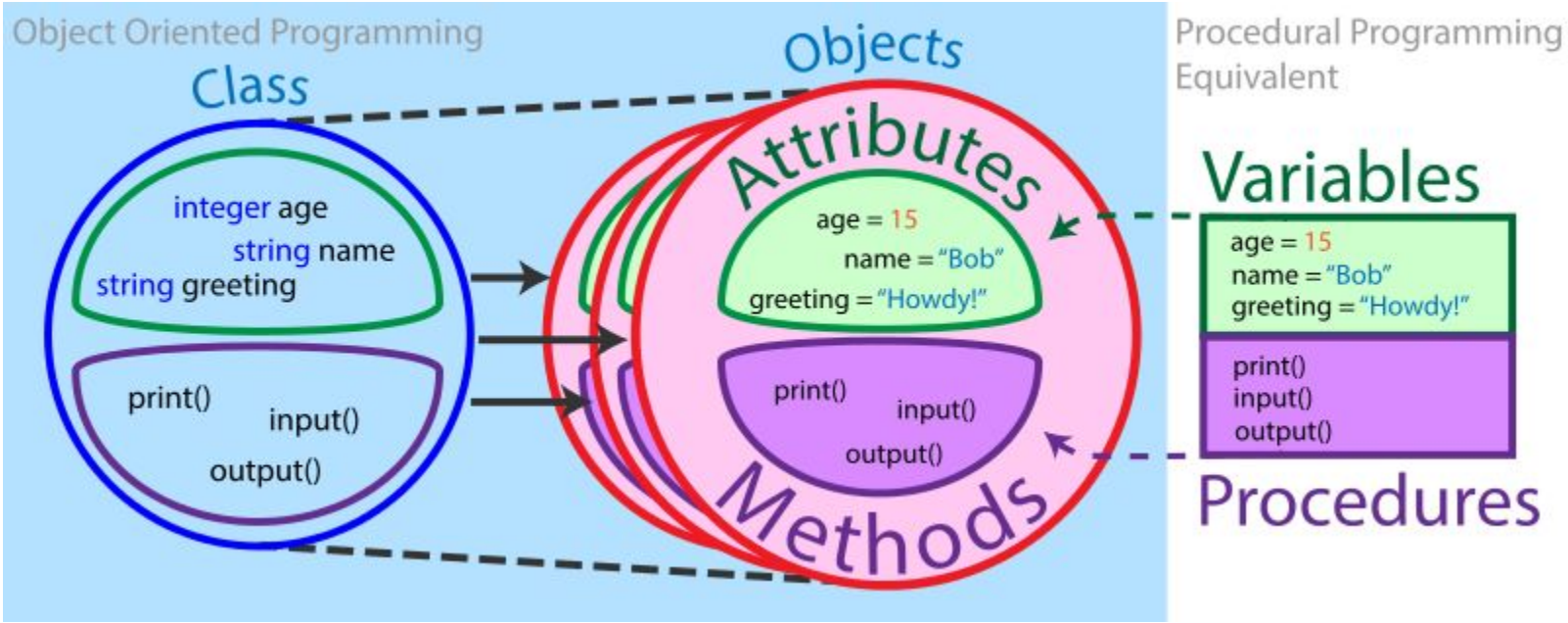


Un objeto tiene estado, exhibe algún comportamiento bien definido, tiene una identidad única.

http://www.gayatlacomulco.com/tutorials/fundamentosdeprog/t21.htm

Clase


class
car
methods
refuel() getFuel
setSpeed() getSpeed()
drive()
attributes
fuel
maxspeed


Object Oriented Programming
Class
integer age
string name
string greeting
print()
input()
output()
Objects
Attributes
age = 15
name = "Bob"
greeting = "Howdy!"
print()
input()
output()
Methods
Procedural Programming Equivalent
Variables
age = 15
name = "Bob"
greeting = "Howdy!"
print()
input()
output()
Procedures

Ejemplo: Clase CarnetIdentidad

Keyword class

JAVA

```
class CarnetIdentidad {

}
```

Nombre de la clase: Upper Camel Case

Ejemplo: Clase CarnetIdentidad

```
class CarnetIdentidad {
    Integer numero;
    Character dv;
    String apellidos;
    String nombres;
    Character sexo;
    String pais;
...
}
```

JAVA

Ejemplo: Clase CarnetIdentidad

Keyword class

Python

```
class CarnetIdentidad():
```

Nombre de la clase: Upper Camel Case

Ejemplo: Clase CarnetIdentidad

```
class CarnetIdentidad():
    def __init__(self):
        self.numero = 1
        self.dv = '9'
        self.apellidos = None
        self.nombres = None
        self.sexo = 0
        self.pais = None
```

Python

El método __init__ es lo que llamamos un constructor… básicamente es donde se inicializa un objeto (la instancia de la clase). La variable self hace alusión a la instancia, y por ende al hacer self.variable estoy definiendo ATRIBUTOS de la clase.
En Java los veremos la próxima clase

Ejemplo: Clase CarnetIdentidad

```
class CarnetIdentidad():
    def __init__(self, numero, dv, apellidos, nombres, sexo,
  pais):
        self.numero = numero
        self.dv = dv
        self.apellidos = apellidos
        self.nombres = nombres
        self.sexo = sexo
        self.pais = pais
```

Esta versión es más interesante

Ejemplo: Clase CarnetIdentidad

```
class CarnetIdentidad(object):
    def __init__(self, numero, dv, apellidos, nombres,
  sexo=None, pais='Chile'):
        self.numero = numero
        self.dv = dv
        self.apellidos = apellidos
        self.nombres = nombres
        self.sexo = sexo
        self.pais = pais
```

Y esta versión aún más :)
Acá digo que si no viene un parámetro le de un valor por omisión

Ejemplo: Clase CarnetIdentidad

```
class CarnetIdentidad(object):

    # código anterior
    # def __init__ ...

...

    def nombre_completo(self):
        return f'{self.nombres} {self.apellidos}'
```

Nombre de métodos: underscore

Uso

```
mi_carnet = CarnetIdentidad(11111111, '1’,
Bórquez Paredes’, Danilo Eduardo',
pais='BitNation')

print(mi_carnet.nombre_completo())
```

NO debo pasar self como parámetro. Al llamarlo como un método de objeto

Ejemplo: Clase CarnetIdentidad

Nombre de métodos: underscore

```
class CarnetIdentidad(object):

    # código anterior
    # def __init__ ...

...

    def nombre_completo(self, separador=''):
        return f'{self.nombres}{separador}{self.apellidos}
```

Uso

```
mi_carnet = CarnetIdentidad(11111111, '1’,
Bórquez Paredes’, Danilo Eduardo',
pais='BitNation')

print(mi_carnet.nombre_completo())

print(mi_carnet.nombre_completo('-'))
```

Paréntesis Programático

Estático vs Instancia

En Java, por omisión, todos los métodos definidos en una clase son de instancia, es decir, solo está disponible si se crea un Objeto de la clase.

(En Python, un método de este tipo es aquel que recibe como parámetro self)

Paréntesis Programático

Estático vs Instancia

En Java, si se usa el keyword static antes del nombre del método, entonces es un método estático y puede ser llamado usando el nombre de la clase en forma directa.

(En Python es si no recibe el parámetro self)

Paréntesis Programático

```
class CarnetIdentidad {

    int numero;
…
    static Character calculaDigito(int numero)
{
    ...
    }
}
```

Conceptos primordiales

2. Abstracción: **representar** lo esencial sin entrar en detalles

¿Qué? Vs ¿Cómo?

Ejemplo: Clase CarnetIdentidad

Keyword abstract

```
abstract class
DocumentoIdentidad
{
    String apellidos;
    String nombres;
    Character sexo;
    String pais;

    abstract Integer
edad();
    abstract Boolean
esValido();
}
```

Nombre de métodos: lower Camel Case

Ejemplo: Clase CarnetIdentidad

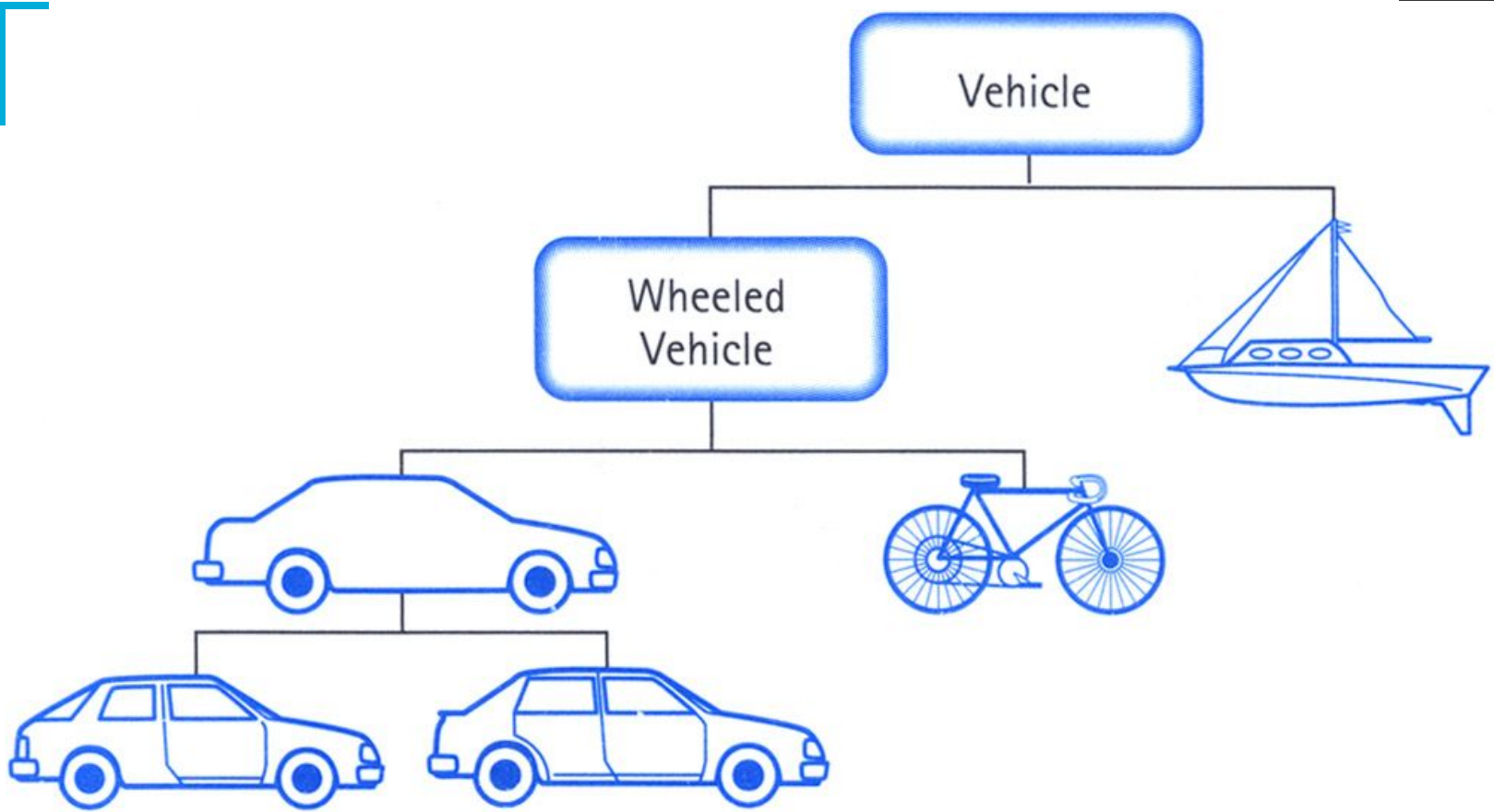
```
class DocumentoIdentidad(object):
    # ...
    # código anterior va acá :)
    # ...

    def digito(s):
        pass
```

Si tengo una clase madre, que debe tener un método por definición, pero la implementación difiere, entonces usamos abstracción

Al usar pass digo que el método no tiene cuerpo. Las clases hijas implementaran su propio cuerpo. Eso se llama Abstracción

Conceptos primordiales

3. Herencia: cuando una clase recibe los atributos encapsulados en otra



http://yadbw.com/inheritance-object-oriented-programming.html

Ejemplo: Clase Pasaporte

```
abstract class Pasaporte extends
DocumentoIdentidad {

    String numeroPasaporte;
}
```

Keyword extends

Pasaporte tiene TODOS los atributos y métodos de DocumentoIdentidad y le agrega numeroPasaporte

Ejemplo: Clase Pasaporte

```
class Pasaporte (DocumentoIdentidad):
    def __init__(self):
        super(DocumentoIdentid
self).__init__(self)
        self.nu                 porte = 1234
```

La clase madre va entre paréntesis… pueden haber varias separadas por ,

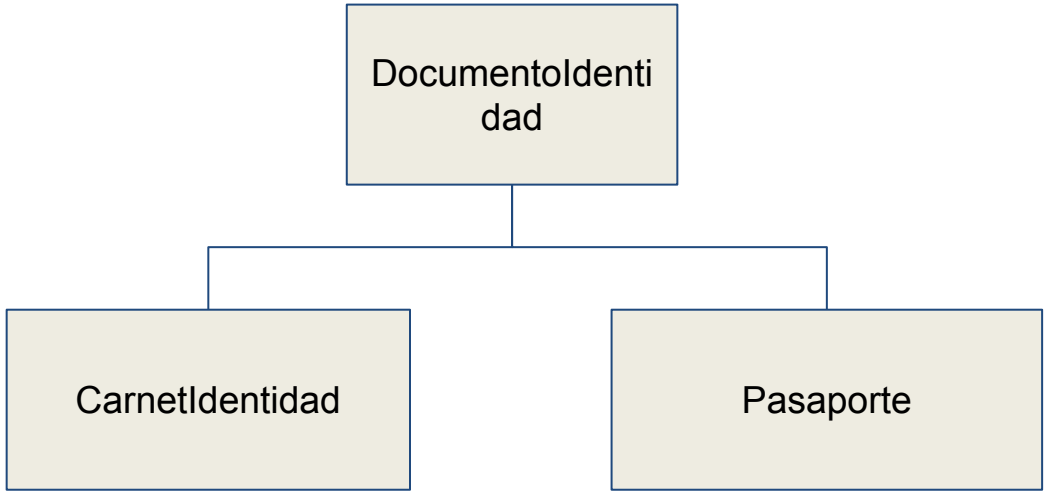
Con esto digo que inicialice los datos según la clase madre

Pasaporte tiene TODOS los atributos y métodos de DocumentoIdentidad y le agrega numeroPasaporte

Conceptos primordiales

4. Polimorfismo: muchas maneras, un nombre

Ejemplo: DocumentoIdentidad



Validación

Carnet Identidad:

```
class CarnetIdentidad(DocumentoIdentidad):
    def digito(s):
        rut = int(s.replace('.', ''))
        value = 11 - sum([ int(a)*int(b)  for
a,b in zip(str(rut).zfill(8), '32765432')])%11
        return {10: 'K', 11: '0'}.get(value,
str(value))
```

Pasaporte:

```
class Pasaporte(DocumentoIdentidad):
    def digito(s):
        return None
```

Ejercicio

Modifiquen su código que implementaba strcmpic de forma que ahora esté dentro de una clase llamada mystring que sea hija de la clase string de Python

¿Qué vimos?

- Discutimos la tarea de strcmpic
- Hicimos la transición a POO
- Un poco de historia de POO
- Conceptos básicos de POO

Tarea

Quebrantahueso programático

¿Quebrantahuesos?

Es un poema que se elabora en base a recortes, juntando frases o palabras que no tienen relación entre sí.

Tarea

Cada uno de Ustedes creará un método (NO una clase, solamente un método), y lo compartirá con sus compañeros a través del foro.

Luego cada uno de Ustedes tomará TODOS los métodos y tratará de agruparlos en una o más clases y luego deberá hacer un main para ejecutar el programa.

¿Preguntas?

El que no pregunta… no pregunta :)