Algoritmos

Logro de sesión

- Al finalizar la sesión, el estudiante **utiliza las relaciones entre clases** en la construcción de programas.

Relaciones entre clases

Contenido:

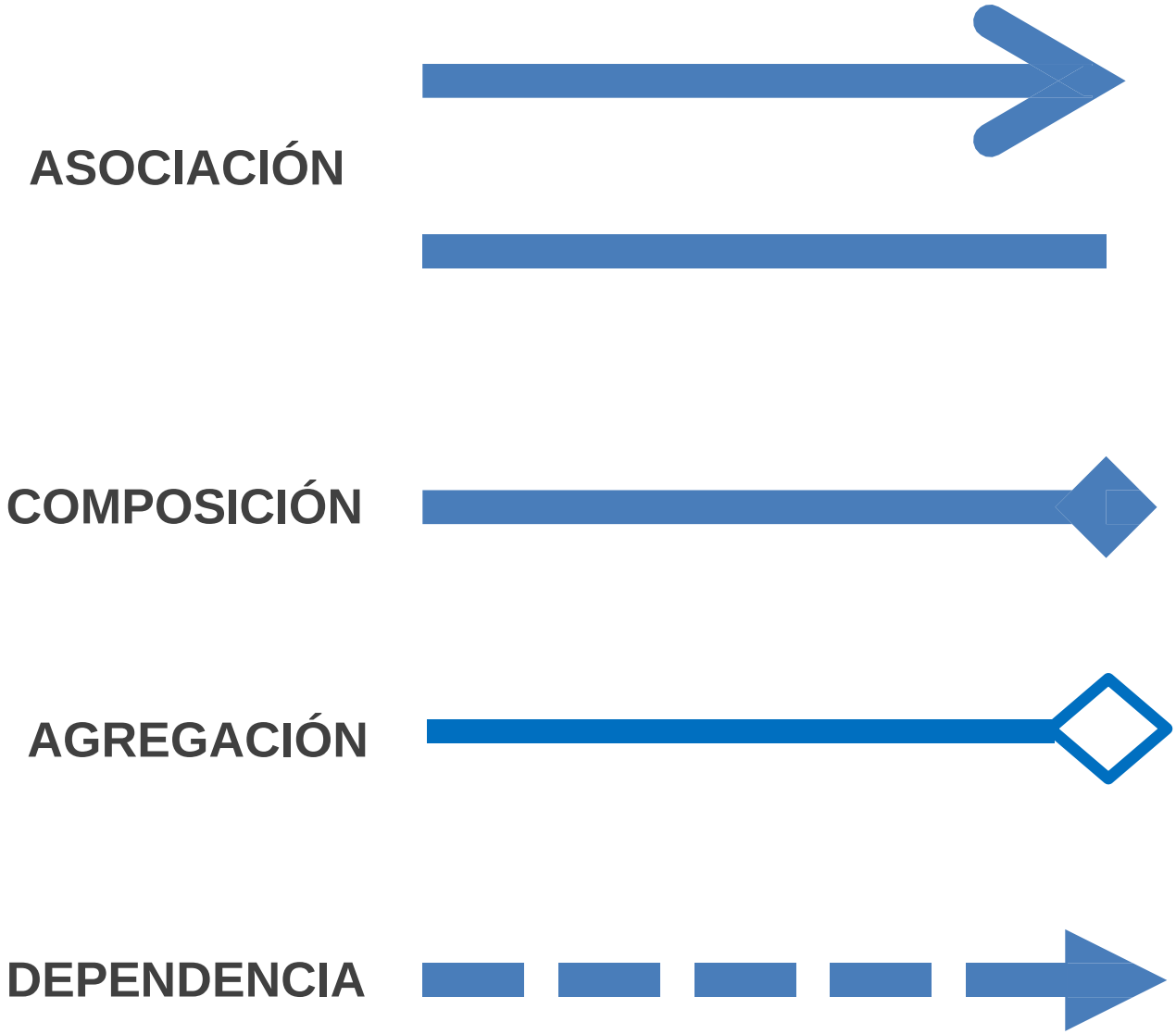
- Relaciones entre clases
  - Dependencia
  - Agregación

Repaso

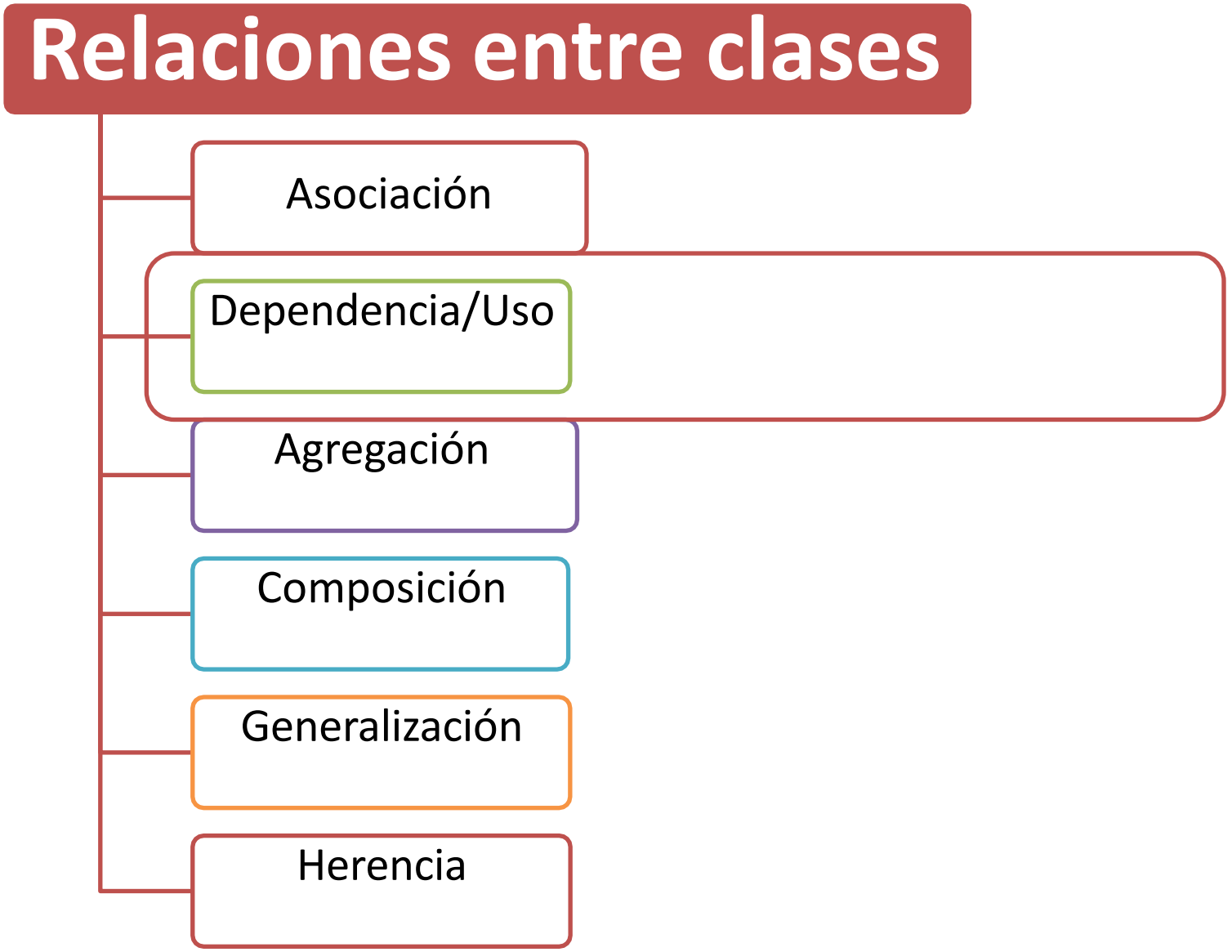
Relaciones entre clases

- Asociación
- Dependencia/Uso
- Agregación
- Composición
- Generalización
- Herencia

Asociacion: Caso Especial de Agregación

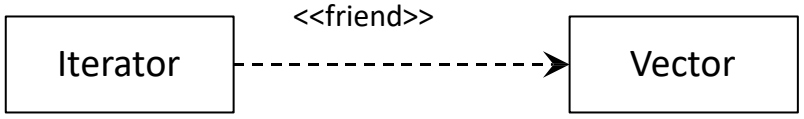


Repaso


Relaciones entre clases
Asociación
Dependencia/Uso
Agregación
Composición
Generalización
Herencia

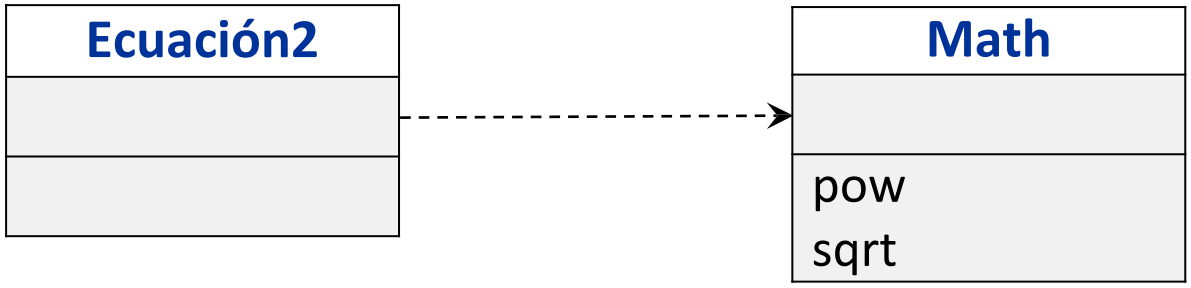
Asociación Caso Especial: Dependencia o Uso

- La dependencia es **una forma más débil de relación** que indica que **una clase depende de otra porque la usa en algún momento.**
- Una clase depende de otra, si la clase independiente es una variable de parámetro o una variable local de un método de la clase dependiente.
- **Esto es diferente de una asociación pura**, donde un atributo de la clase dependiente es una instancia de la clase independiente.



Asociacion: Caso especial - Dependencia

- La Ecuacion2 utiliza los servicios de la clase Math (funciones pow y sqrt) para el cálculo:



- Se utiliza para representar una relación en la que **un objeto cliente solicita un servicio a un objeto servidor**

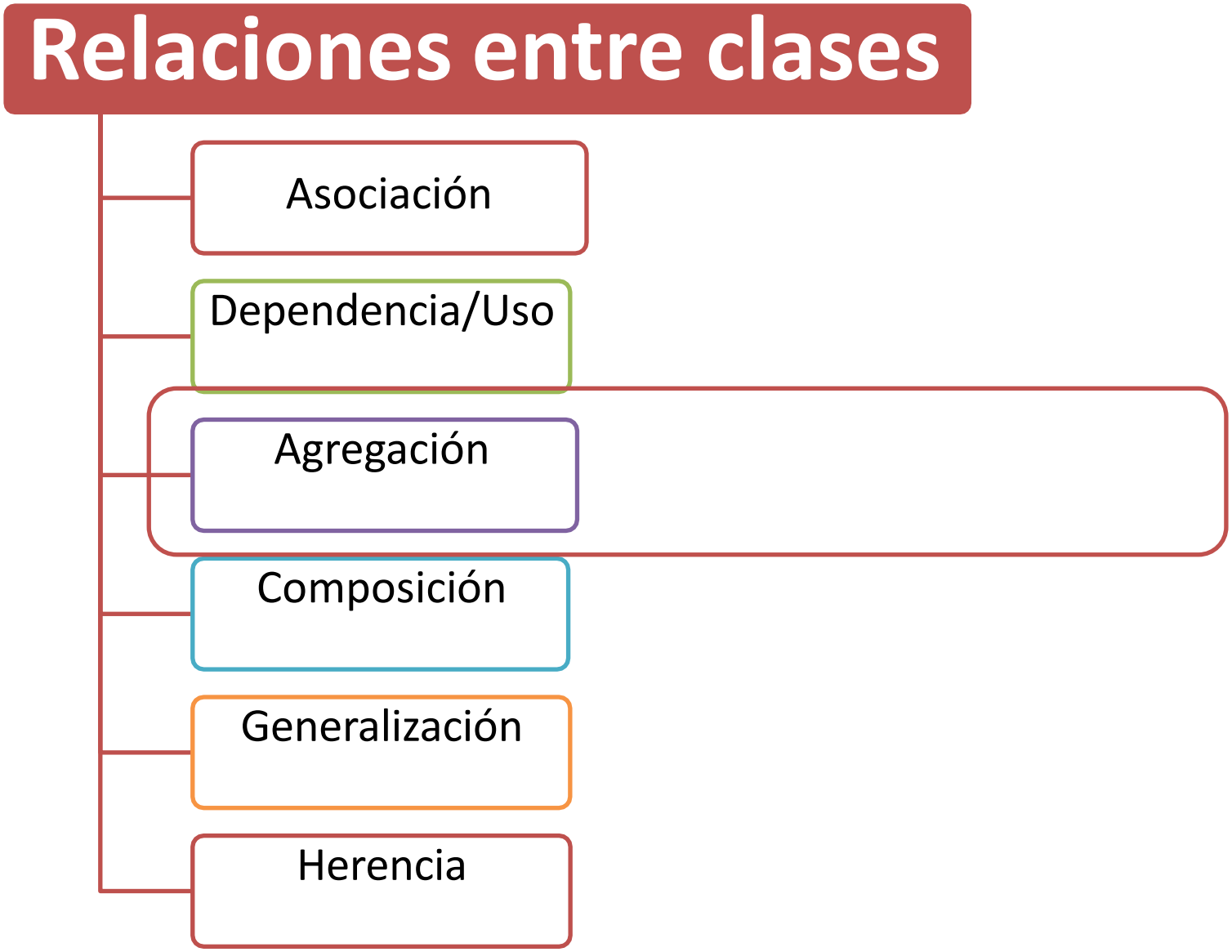
$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Ejemplo: Implementación en c++

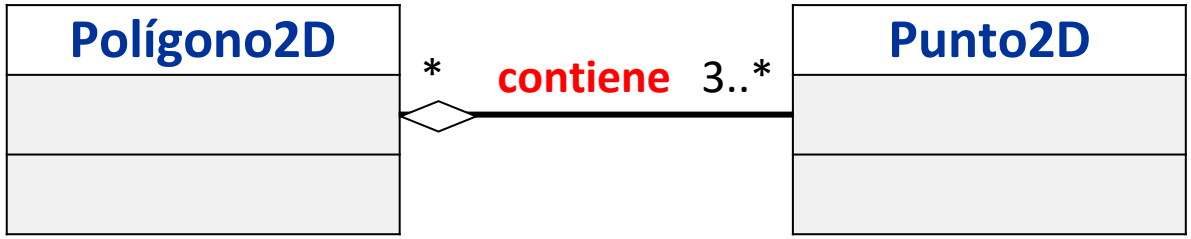
```
class A{
private:
        int x;
public:
        A(){
                x=0;
        }
};
```

```
class B{
private:
        int y;
        A obj1;
public:
        B(){
                y=0;
        }
        void uso (A obj){
                y=obj.x;
        }
};
```

Repaso


Relaciones entre clases
Asociación
Dependencia/Uso
Agregación
Composición
Generalización
Herencia

Asociacion: Caso Especial de Agregación



- **Cada Polígono2D está formado por 3 o + Puntos2D.**

```
class Poligono2D {
  private Punto2D vertices[];
  …
}
```

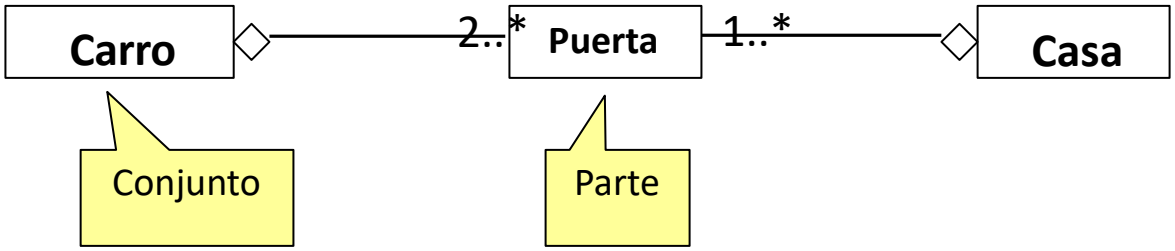
```
class Punto2D {
   …

}
```

- Se utiliza para representar la relación de que un objeto es parte de otro. **El objeto que es parte del otro puede existir o no independiente del objeto que lo agrega a su definición.**

Agregación

- Una especial forma de asociación que modela una relación entre un agregado (el todo) y sus partes.
- Modela la relación **"es una parte de"**.



Agregación

- La frase "parte de" utilizada para describir la relación
  - **Una puerta es parte de un carro**
- Algunas operaciones en conjunto se aplican automáticamente a sus partes
  - **Mueve el coche, mueve la puerta**.
- Algunos valores de atributos son propagados a todas o algunas de sus partes
  - **El coche es azul, por lo tanto la puerta es azul**.
- Existe una **asimetría** intrínseca a la relación donde una clase está subordinada a la otra
  - Una puerta es parte de un coche. Un coche no es parte de una puerta.

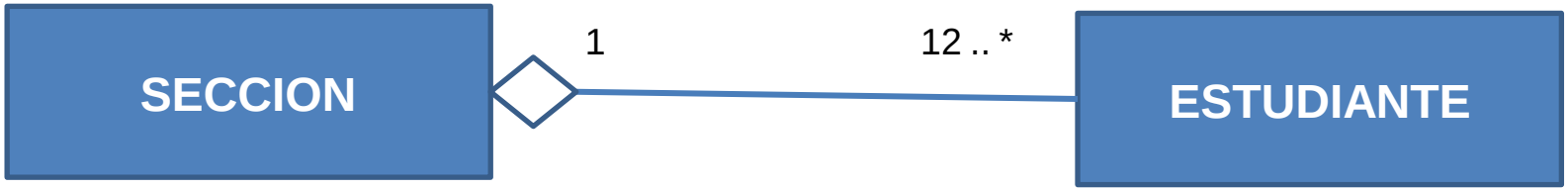
Ejemplo: Implementación en c++

```
class A{
private:
        int x;
public:
        A(){
                x=0;
        }
};
```

```
class B{
private:
        int y;
        A obj1;
public:
        B(A newobj){
                y=0;
                obj1=newobj;
        }
        ~B(){
                y=0;
                obj1=NULL;
        }
};
```

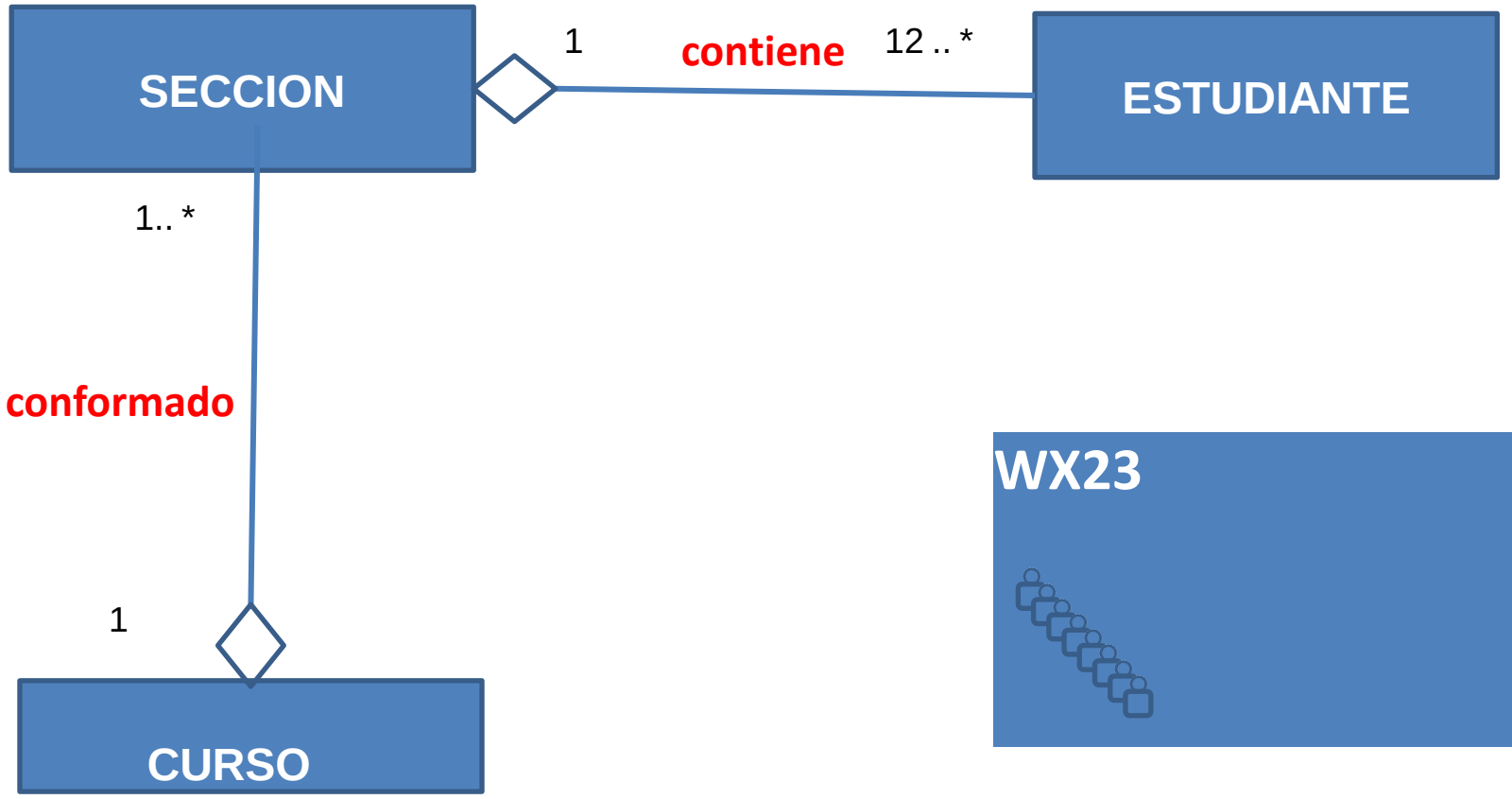
Ejercicio:

- Diagrame en UML e Implemente la relación de agregación entre Sección y Estudiante.



Ejercicio:

- Diagrame en UML e Implemente la relación de agregación entre Sección y Estudiante.



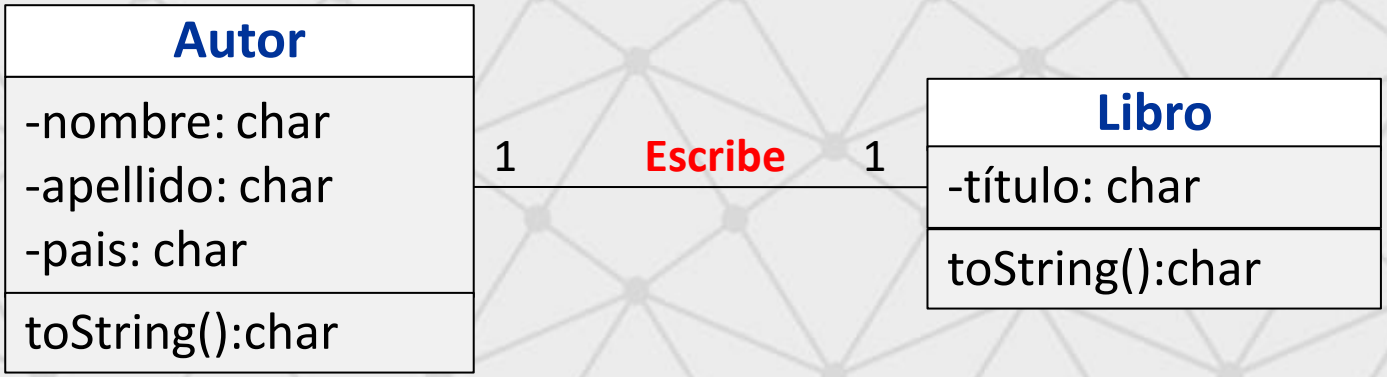
Ejemplo1

- La Biblioteca de la UPC maneja información de libros y de sus autores; **suponga que**:
  - De cada libro se conoce el título.
  - De los autores se conoce: su nombre, apellido y país de procedencia.
  - Un libro es escrito por uno y solo un autor
- Se pide:
  - Elaborar el diagrama de clases.
  - Implementar el diagrama de clases.
  - Implementar los métodos de servicio, si se sabe que ambas clases poseen un método que devuelve la información contenida en las propiedades de la clase.
  - Elabore una clase de prueba para probar el funcionamiento de las instancias de la clase.

Ejemplo1

DIAGRAMA DE CLASES

- Un **Autor** escribe **Un** Libro
- Un **Libro** es escrito por **Un** autor



Ejemplo 1

autor.h

```
#ifndef _AUTOR_H_
#define _AUTOR_H_
#include<string>
class Autor{
        private:
        char* nombre, apellido, pais;
        public:
        Autor(char* n, char* a, char* p){
                nombre = n;
                apellido = a;
                pais = p;
        }
        ~Autor(){}

        //Metodos de servicio
    char* toString(){
        string val = “nom:” +
(string)nombre +  “ape:” +
(string)apellido +
        “pais:” + (string)pais;
        char* valret = &*val.begin();
        return (valret);
        }
};
#endif
```

libro.h

```
#ifndef _LIBRO_H_
#define _LIBRO_H_
#include<sstream>
class Libro{
        private:
        char* titulo;
        Autor* autor;
        public:
        Libro(char* t, Autor* a){
                titulo = t;
                autor = a;
        }
        ~Libro(){}
        //Metodos de servicio
    char* toString(){
        string val;
        val+= “titulo:” + (string)titulo;
        val+= (string)(autor-
>toString());

        char* valret = &*val.begin();
        return (valret);
        }
};
#endif
```

Ejemplo 1: prueba

```
#include<iostream>
#include “autor.h”
#include “libro.h”
using namespace std;

int main() {
        Autor* a = new Autor("Luis", "Joyanes", "España");
        Libro* l = new Libro(“C++”, a);
        boolean res = a->agregarLibro(l);
        cout<<“Objeto Autor: ” <<a->toString()<<endl;
        cout<<“Objeto Libro: ” <<l->toString()<<endl;
        cin.get();
        return(0);
}
```

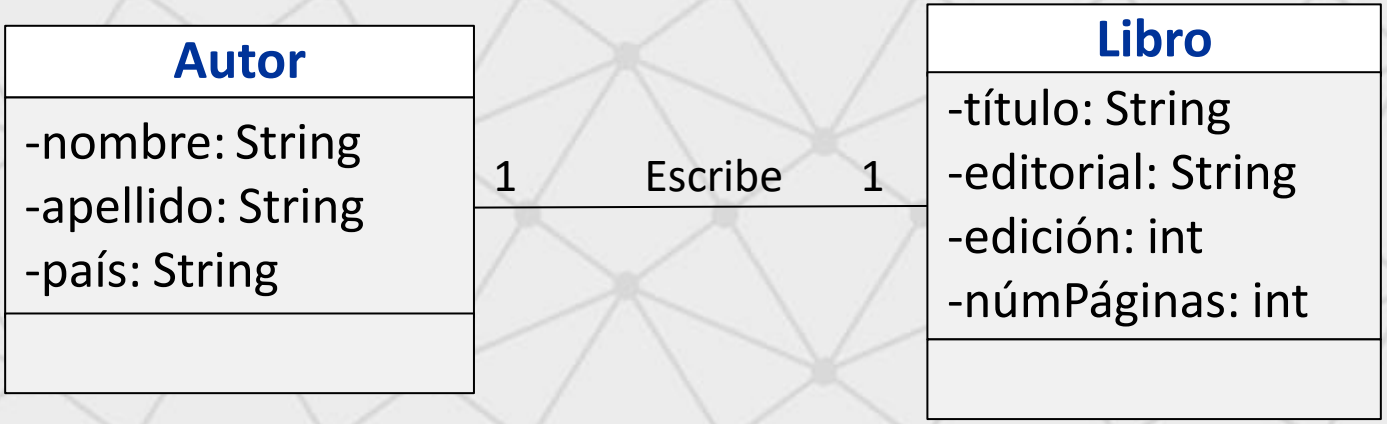
Ejemplo 2

- La Biblioteca de la UPC maneja información de muchos libros y de sus autores; **suponga que**:
  - De cada libro se conoce: **título, editorial, edición y número de páginas**.
  - De los autores se conoce: **su nombre, apellido y país de procedencia.**
  - Un libro es escrito por uno y solo un autor
- Se pide:
  - Elaborar el diagrama de clases.
  - Implementar el diagrama de clases.
  - Implementar los métodos setter/getter

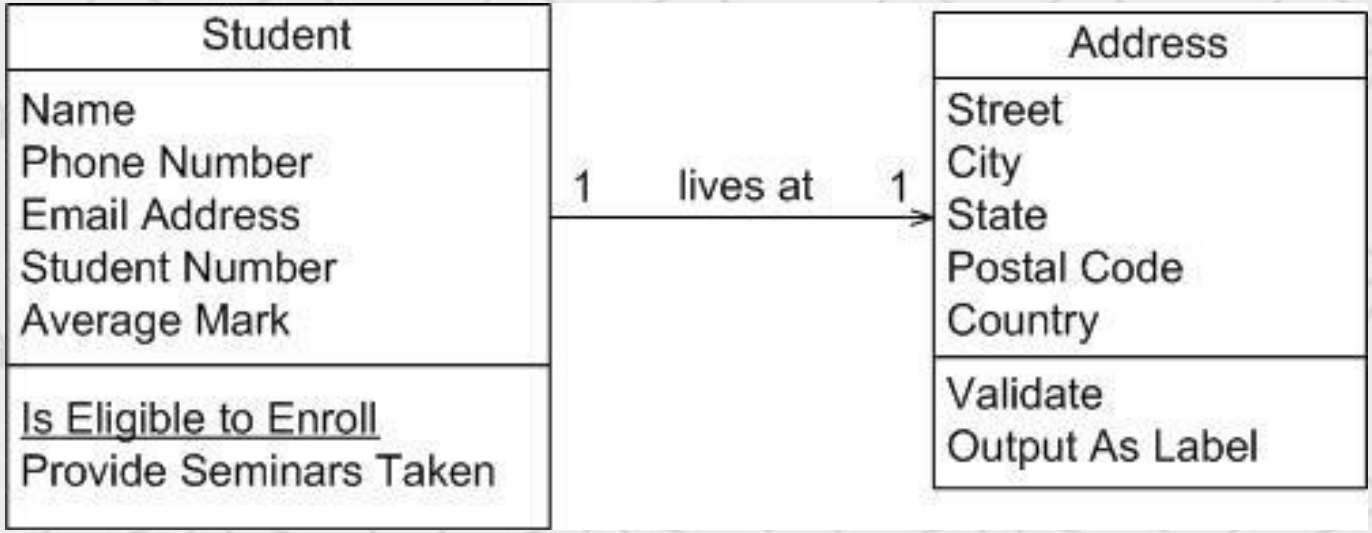
Ejemplo 2

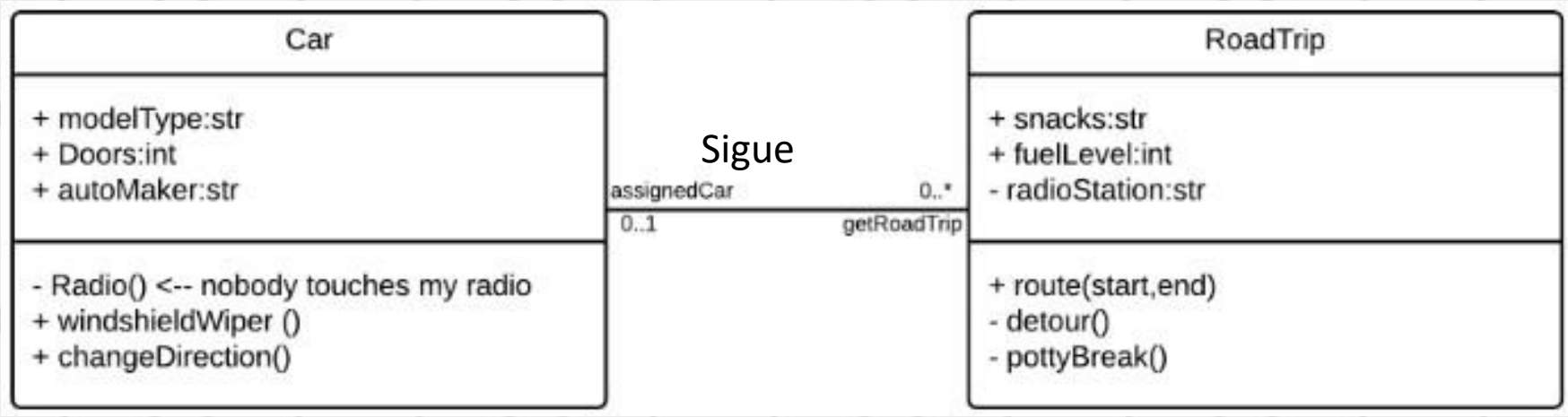
DIAGRAMA DE CLASES

- Un **Autor** escribe **Un** Libros
- Un **Libro** es escrito por **Un** autor

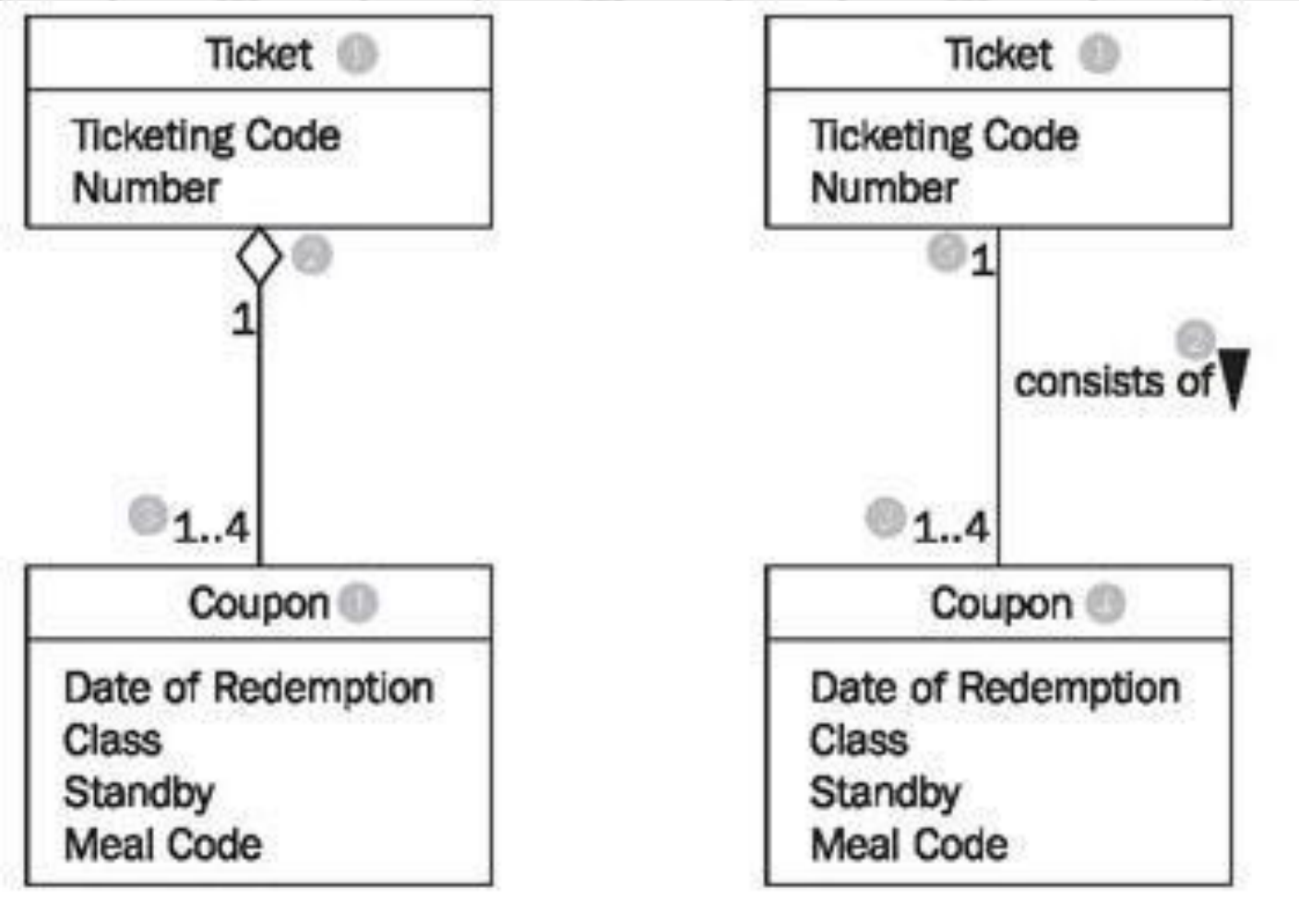


Ejercicios

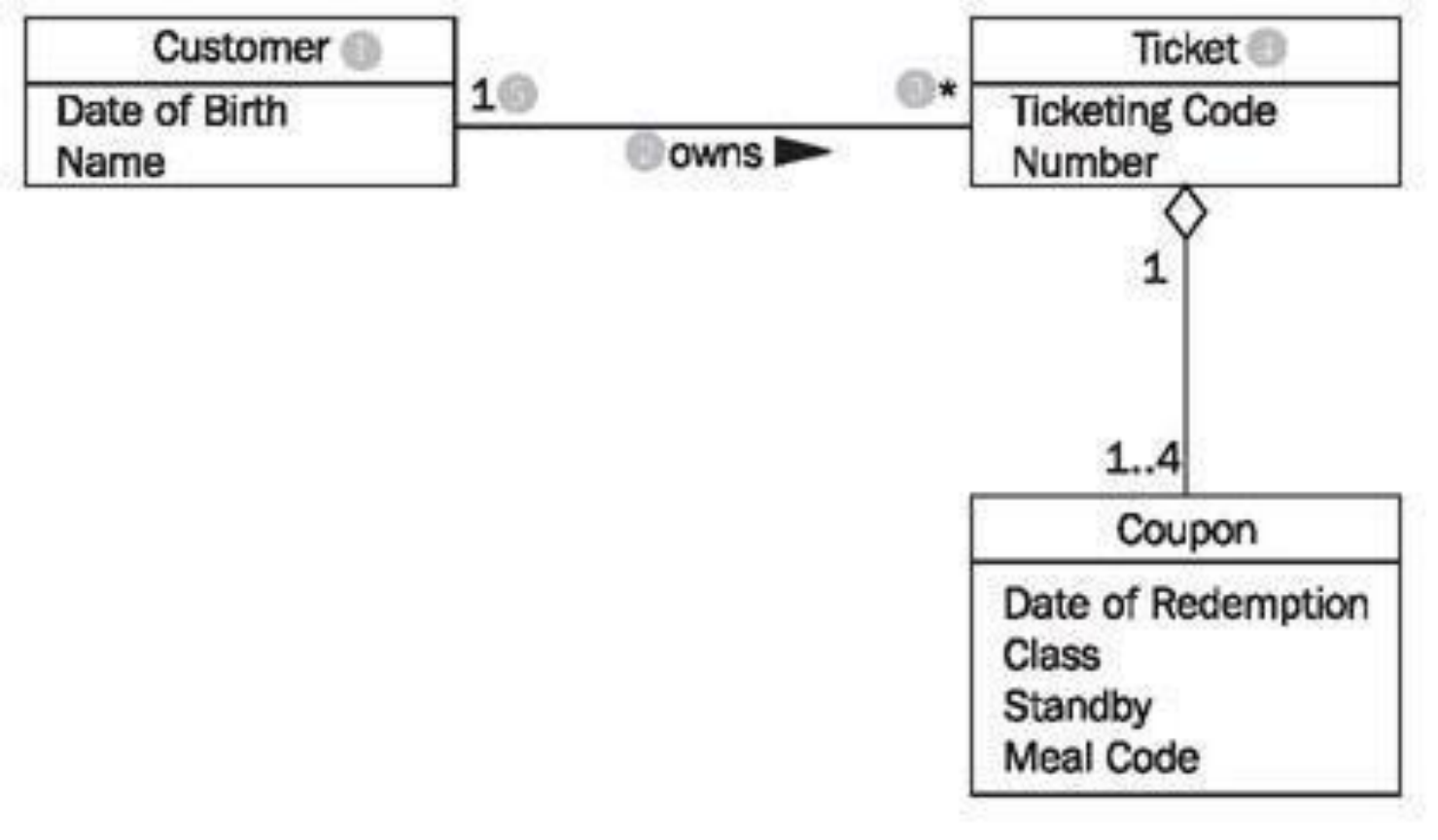

Student
Name
Phone Number
Email Address
Student Number
Average Mark
Is Eligible to Enroll
Provide Seminars Taken
1
lives at
1
Address
Street
City
State
Postal Code
Country
Validate
Output As Label


Car
+ modelType:str
+ Doors:int
+ autoMaker:str
- Radio() <-- nobody touches my radio
+ windshieldWiper ()
+ changeDirection()
Sigue
assignedCar
0..*
0..1
getRoadTrip
RoadTrip
+ snacks:str
+ fuelLevel:int
- radioStation:str
+ route(start,end)
- detour()
- pottyBreak()

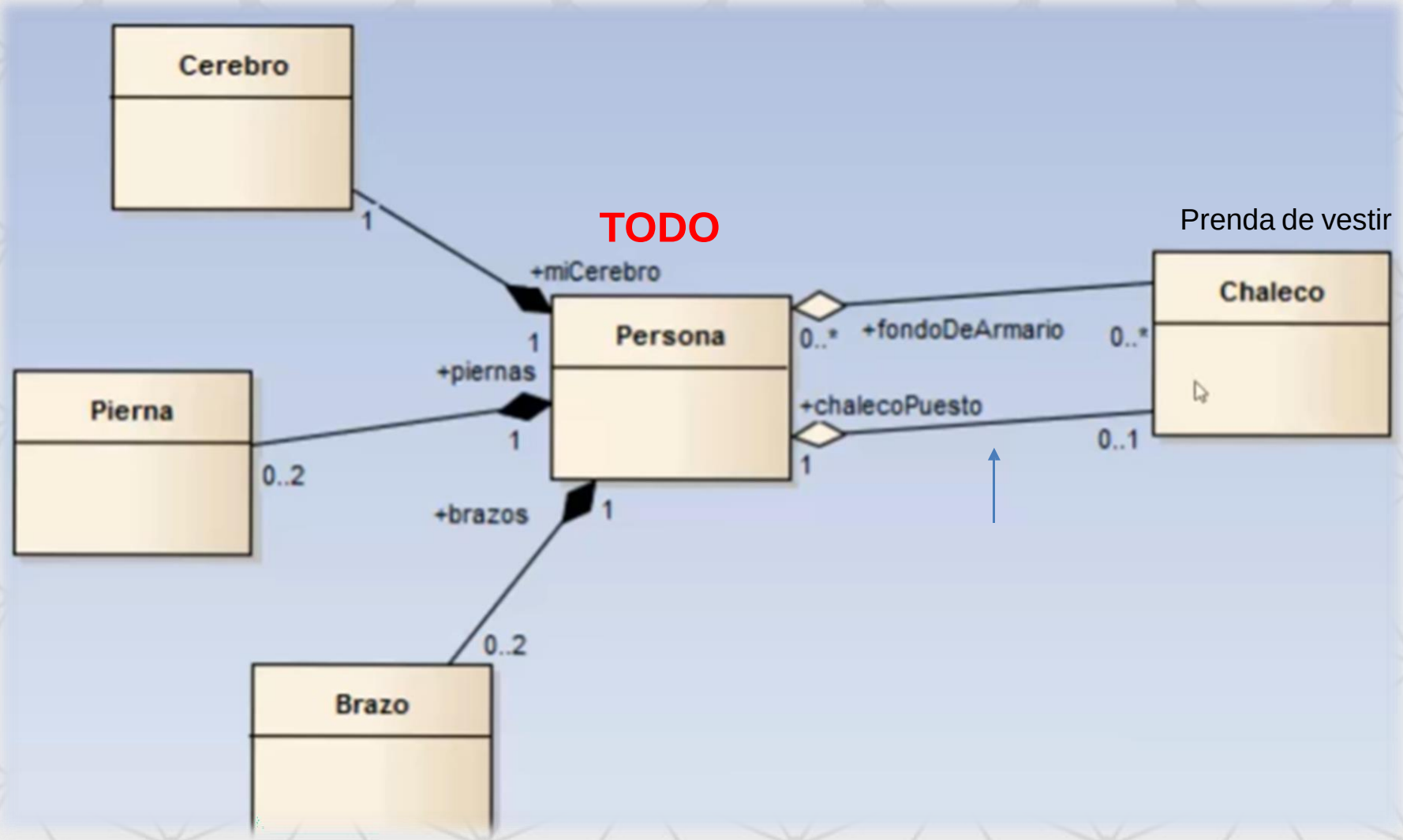
Ejercicios


Ticket
Ticketing Code
Number
1
1..4
Coupon
Date of Redemption
Class
Standby
Meal Code
Ticket
Ticketing Code
Number
1
consists of
1..4
Coupon
Date of Redemption
Class
Standby
Meal Code

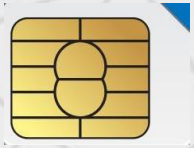
Ejercicios

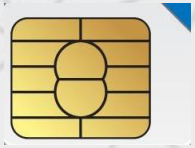

Customer
Date of Birth
Name
1
owns
*
Ticket
Ticketing Code
Number
1
1..4
Coupon
Date of Redemption
Class
Standby
Meal Code

Ejemplos


Cerebro
1
+miCerebro
1
Persona
+piernas
Pierna
1
0..2
+brazos
1
0..2
Brazo
TODO
Prenda de vestir
Chaleco
0..*
+fondoDeArmario
0..*
+chalecoPuesto
1
0..1

Agregación



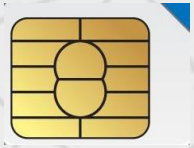


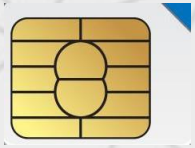
Imaginemos un Smartphone. Al fabricar un Smartphone no se fabrica con un chip o tarjeta SIM

Agregación

Pero podemos agregar uno o dos chips.
El celular funcionará con o sin chip. Porque el chip no es parte del celular

Agregación

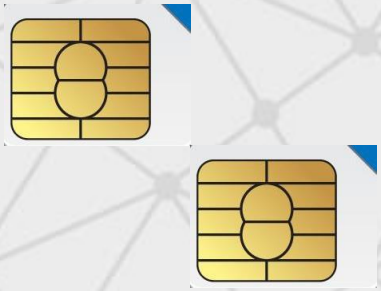




Podemos decir que estas clases pueden funcionar uno independientemente del otro.

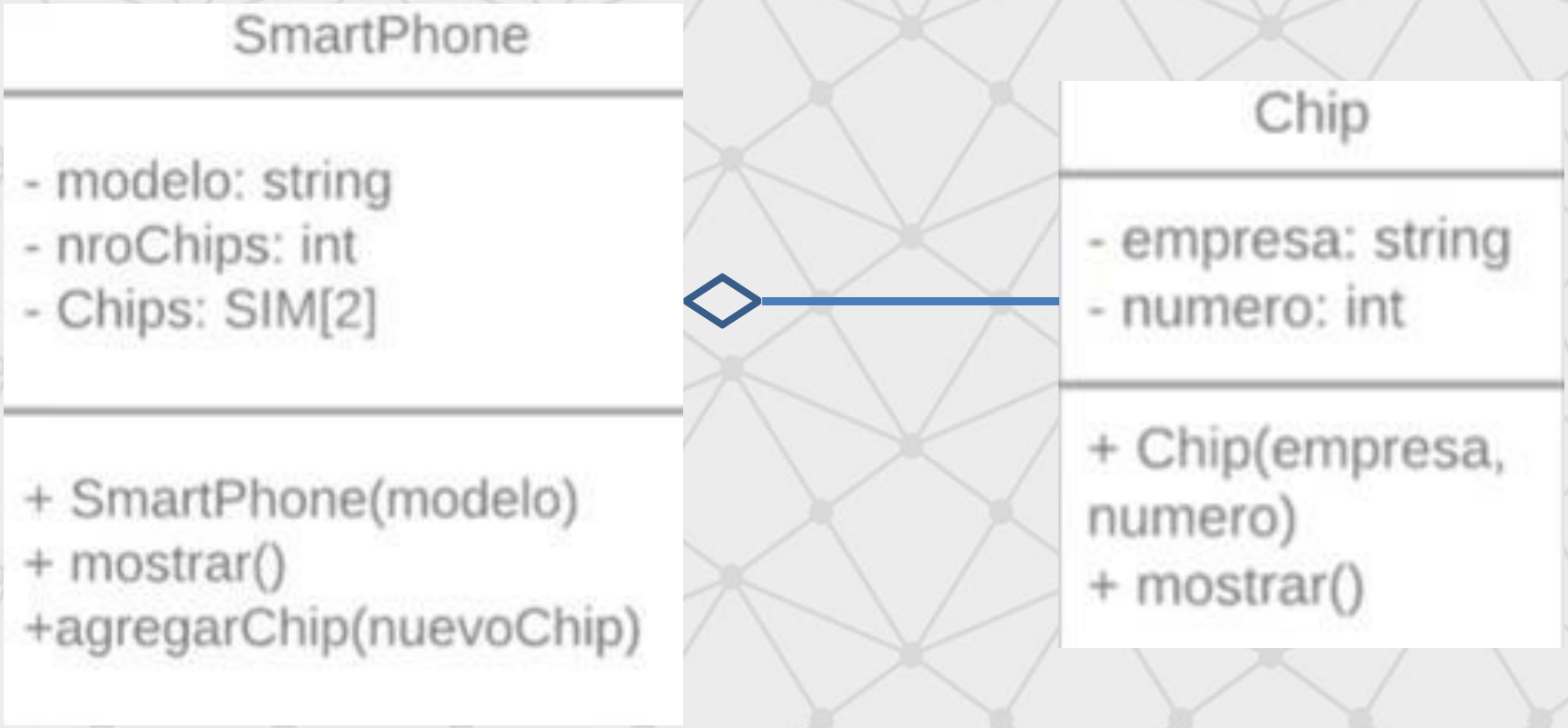
Agregación

CONTENEDOR



Si la clase contenedora es destruida los elementos seguirán existiendo

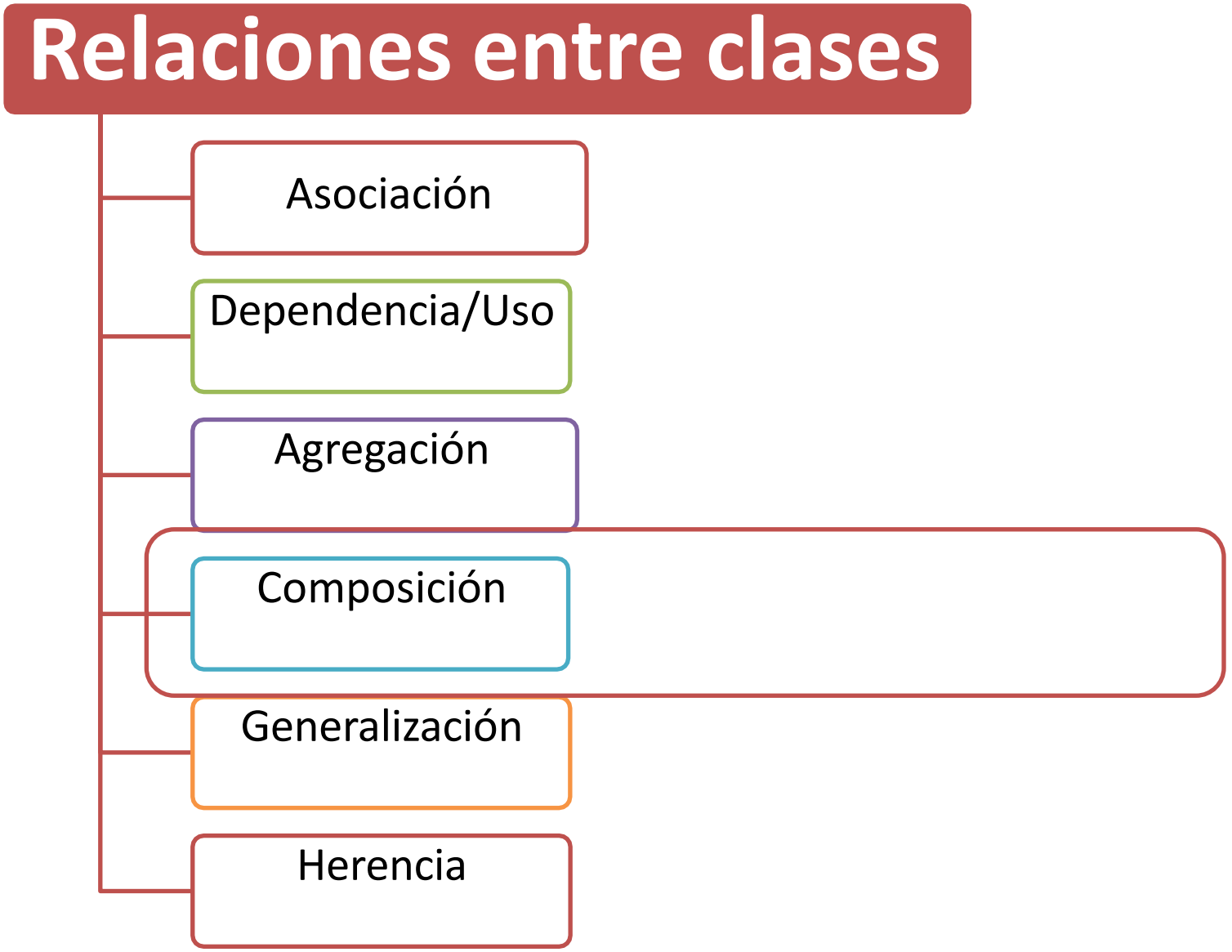
Agregación


SmartPhone
- modelo: string
- nroChips: int
- Chips: SIM[2]
+ SmartPhone(modelo)
+ mostrar()
+agregarChip(nuevoChip)
Chip
- empresa: string
- numero: int
+ Chip(empresa, numero)
+ mostrar()

Agregación

Un celular tiene un Chip
Un chip esta en un celular

Repaso


Relaciones entre clases
Asociación
Dependencia/Uso
Agregación
Composición
Generalización
Herencia

Composición

TODO

PARTE

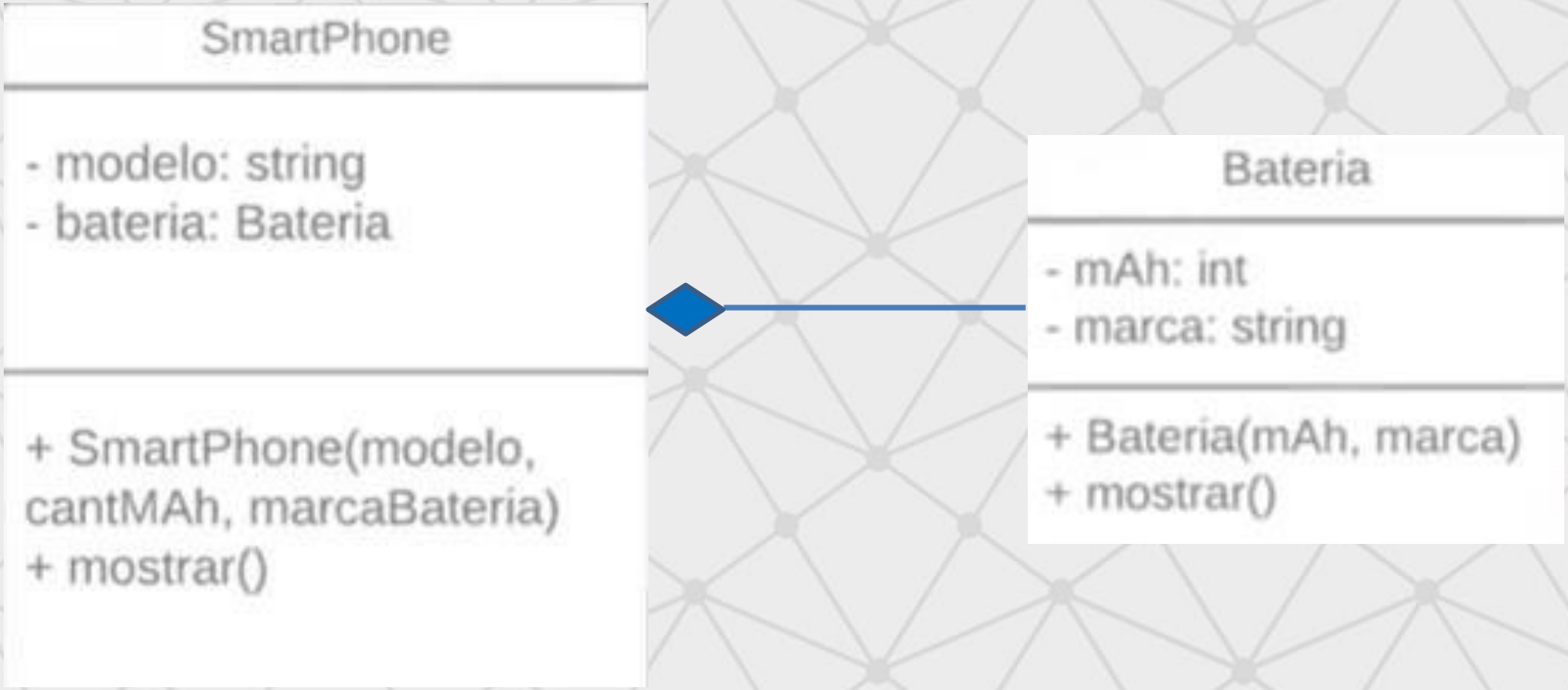
Un smartphone no puede funcionar sin batería

Composición

TODO

Si el objeto todo es destruido también se destruyen sus partes

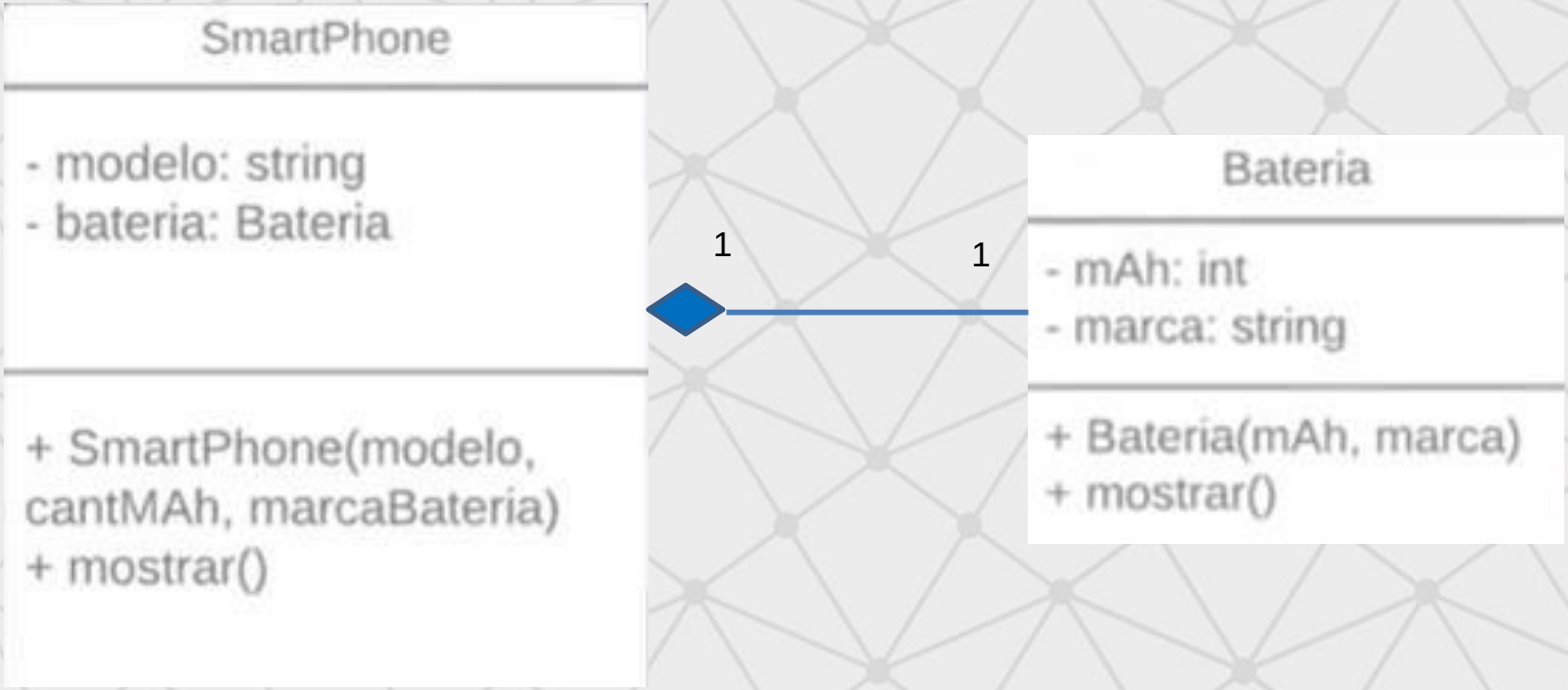
Composición



Composición

"Se compone de"
"compone a"

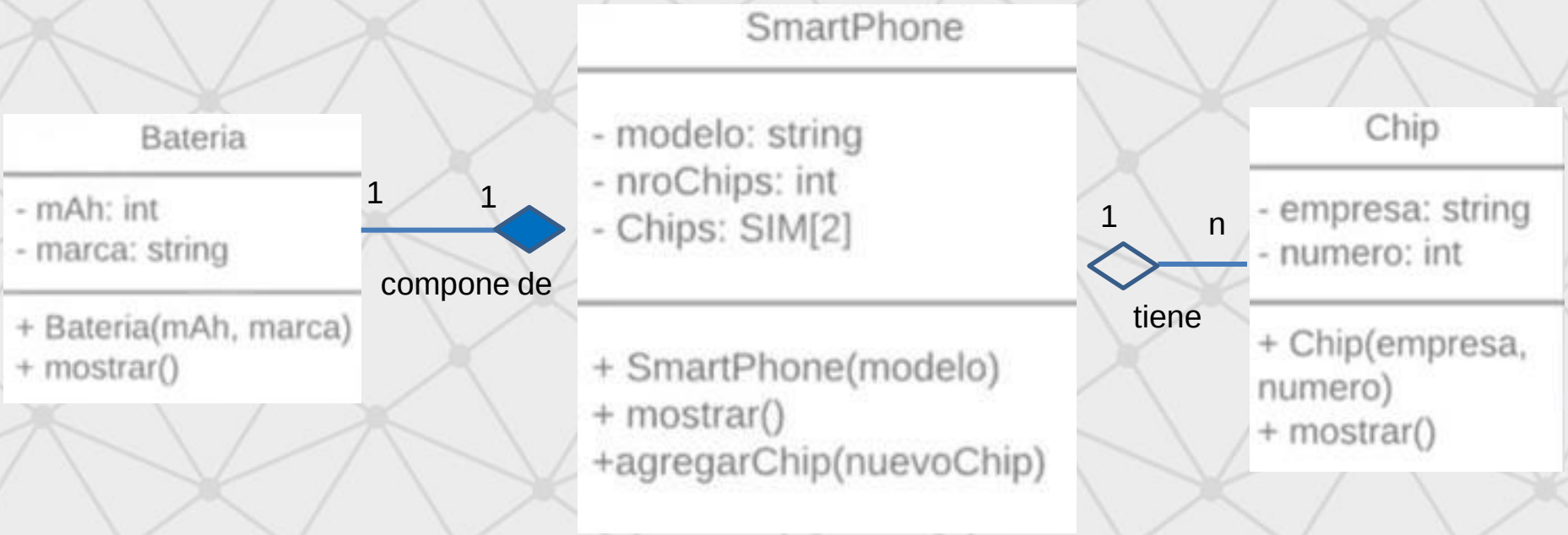
Composición



Un smartphone se compone de una batería
Una batería compone a un smartphone

Ejemplo de Aplicación

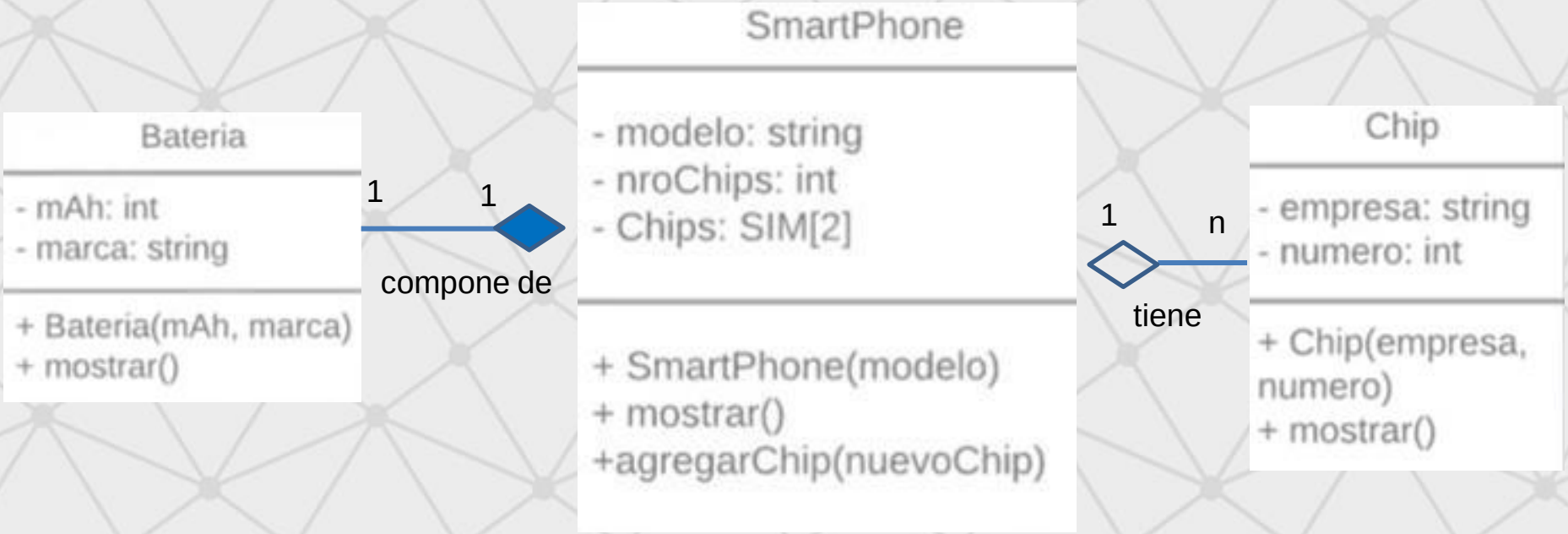
Diagrama de Clase



Ejercicio: Agregar la clase airpods y case.

Ejemplo de Aplicación

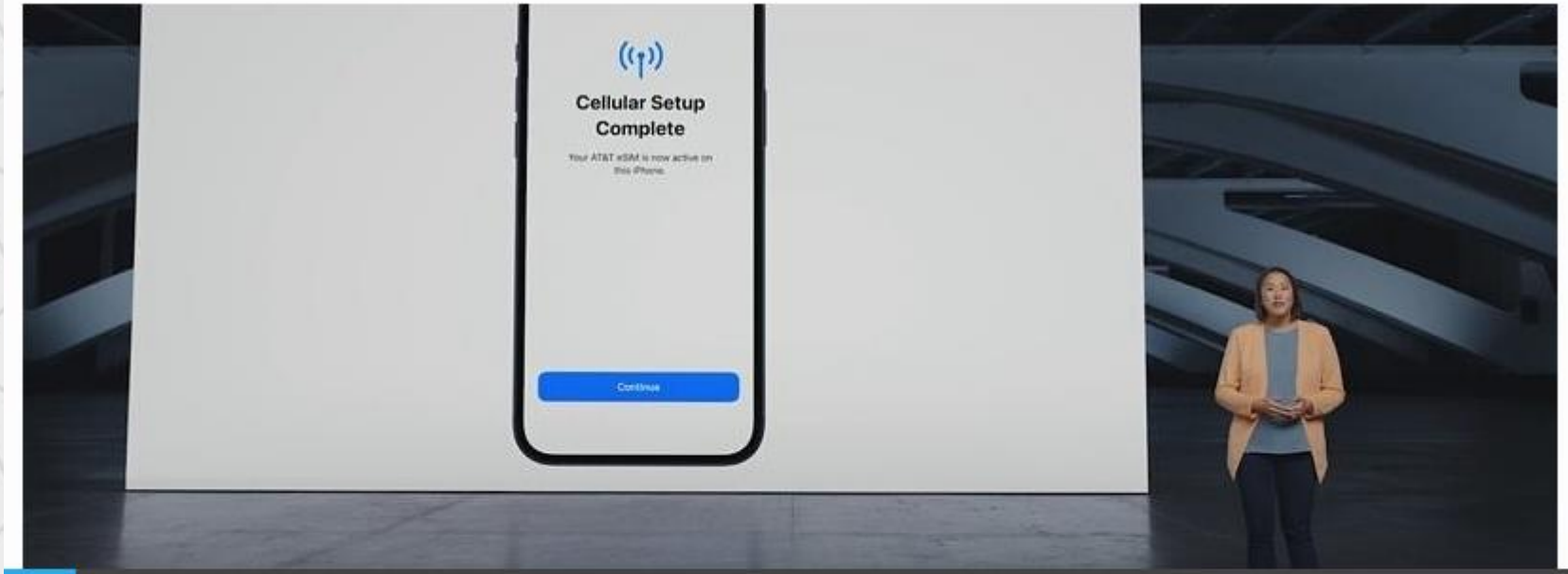
Diagrama de Clase



Aplicación

Ejemplo de Aplicación

07.09.2022



September 7, 2022 2:07pm Comment João Carrasqueira

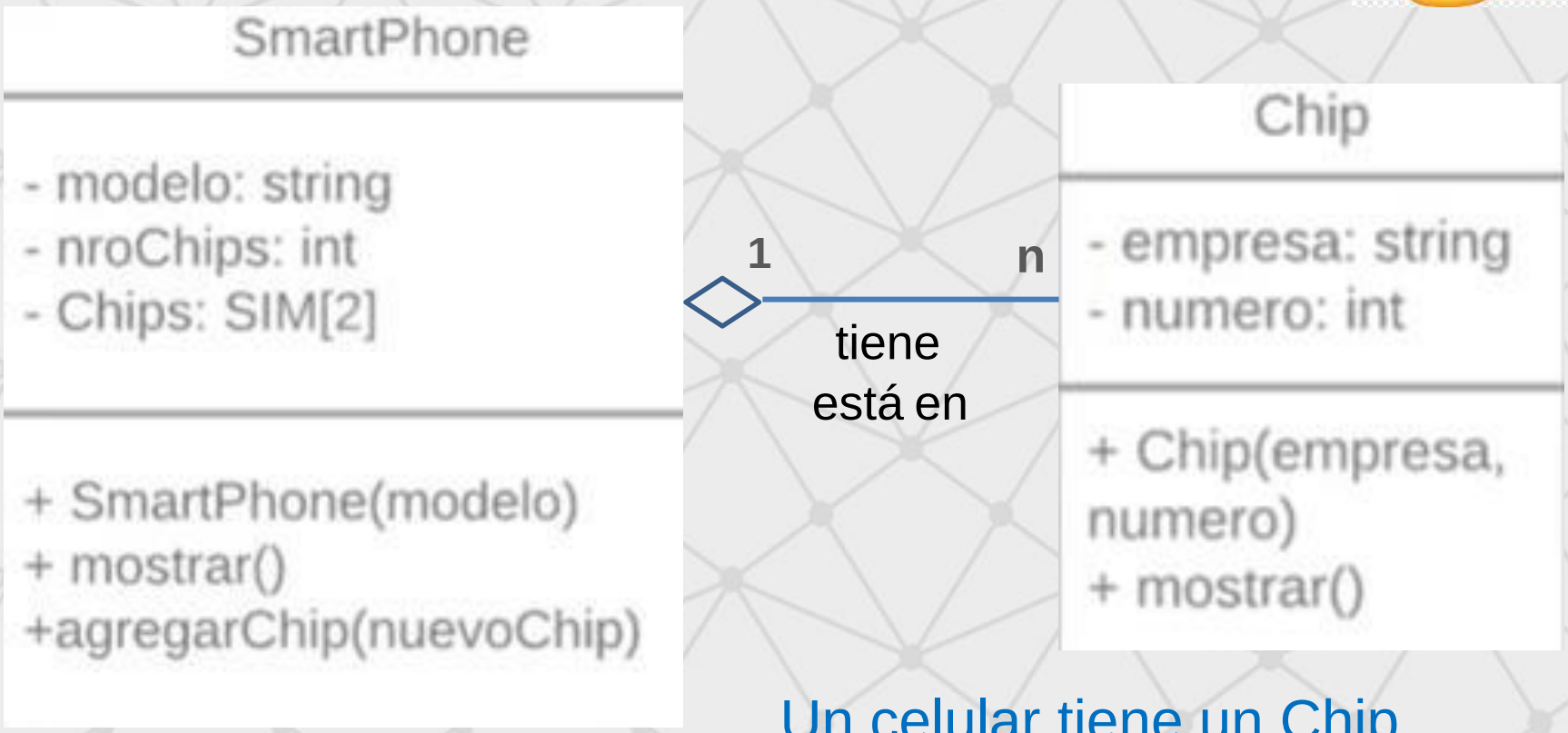
The iPhone 14 no longer comes with a SIM slot in the US

Ejemplo de Aplicación

07.09.2022

Agregación

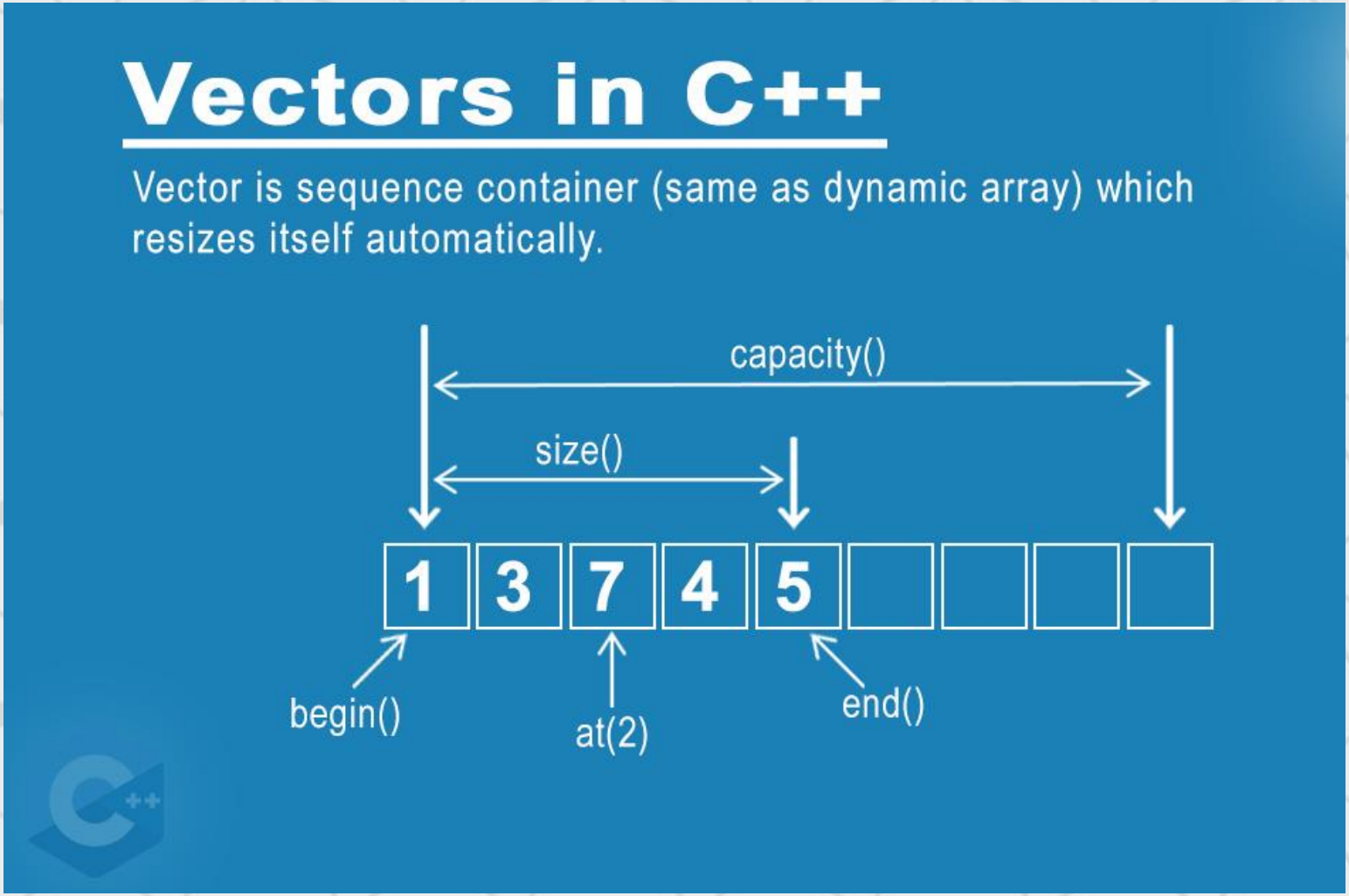
Entonces….¿Cómo cambia el diagrama?



Un celular tiene un Chip
Un chip esta en un celular

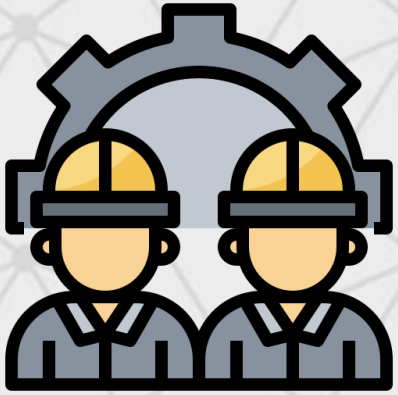
Ejemplo de Aplicación

Contenedor



https://cplusplus.com/reference/vector/vector/

Ejercicio de Aplicación



Una empresa marca ACME, decide cambiar su política de retención del talento, con dos actividades:

1.Celebraciones presenciales de los cumpleaños del mes.

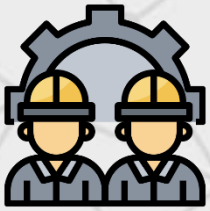
2.Agazajar a los empleados que cumplen aniversario en la empresa.

Se solicita un diagrama de clases que permita representar estos cambios realizados por el área de RRHH y dos reportes:

a. Cantidad de empleados contratados en Marzo 2017.

b. Empleados nacidos en Abril.

Ejercicio de Aplicación



Empleado

Dia

ListaEmpleado