# **Motivación**

1. Entender mejor los programas.

2. Que los programas sean más fáciles de escribir y mantener

3. Que sea más fácil reutilizar los programas

**Falla de la clase de DPOO:** Hacer entender la necesidad de un concepto como este. Dar un panorama básico de qué tipo de problemas resuelve este tipo de Estructuras.

Existen Conceptos que permiten desentender funcionalidades o naturalezas de datos de un programa en específico. Por ejemplo, los arraylist se usan en muchísimos contextos, pero se pueden modificar para un programa en específico para que funcione mejor: en vez de hacer más líneas de código, podríamos establecer una estructura que permita modificar la implementación de los Arraylist, ya definida dentro de un archivo que se puede invocar. Aquí los arraylist serían una superclase, y para cada funcionalidad se podría modificar esta clase (subclase) a través del concepto de Herencia [Clase Abstracta, Interfaz]. Puede darse, además, que un concepto se derive de otros a la vez (Polimorfismos), por ejemplo, si quisiéramos aprovechar la portabilidad de los arraylist con la eficacia para encontrar datos de los HashMaps (Como una especie de diccionarios), podríamos hacer una clase que venga de los dos y aprovechar ciertas funcionalidades de cada uno.

Detrás de estas estructuras también surgen conveniencias, como las relaciones entre las clases: Cómo interactúan entre sí. El concepto de herencia es solo una de ellas, pero hay más

**1. Asociación**

* **Definición**: Una clase tiene una referencia a otra.
* **Ejemplo**: Un Estudiante puede estar asociado con un Curso.
* **Tipos**:
  + **Uno a Uno** (1:1): Un Pasaporte pertenece a un único Ciudadano.
  + **Uno a Muchos** (1:N): Un Profesor puede enseñar a varios Estudiantes.
  + **Muchos a Muchos** (N:M): Varios Estudiantes pueden inscribirse en varios Cursos.

**2. Agregación (Has-A)**

* **Definición**: Es un tipo de asociación donde una clase "tiene" otra, pero ambas pueden existir independientemente.
* **Ejemplo**: Un Departamento tiene varios Empleados, pero si el Departamento se elimina, los Empleados pueden seguir existiendo.

**3. Composición**

* **Definición**: Es una relación más fuerte que la agregación, donde una clase "posee" otra y no pueden existir por separado.
* **Ejemplo**: Un Coche tiene un Motor. Si el Coche se destruye, el Motor también deja de existir.

**4. Herencia (Is-A)**

* **Definición**: Una clase (subclase) hereda atributos y métodos de otra (superclase).
* **Ejemplo**: Perro es un Animal (Perro extends Animal).

**5. Implementación (Interface)**

* **Definición**: Una clase implementa una o varias interfaces y proporciona su comportamiento.
* **Ejemplo**: Pájaro implementa Volador (class Pájaro implements Volador).

**6. Dependencia (Usa)**

* **Definición**: Una clase usa otra temporalmente para una tarea específica.
* **Ejemplo**: Un Cliente usa un CarritoDeCompras para comprar productos.

¿Por qué asociar entre clases?