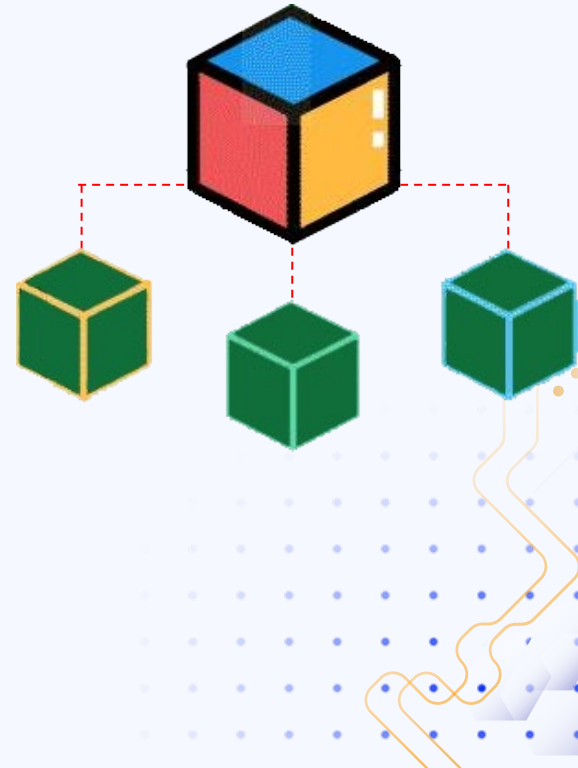
The background features a light blue grid with various colored lines (purple, orange, blue) and dots. There are three 3D cubes: a small one in the top left, a medium one in the top center, and a large one in the bottom left. All cubes have red, blue, and yellow faces. A yellow parallelogram box is on the right side, containing the title text. A blue line connects the large cube to the yellow box.

Programación Orientada a Objetos (Herencia)

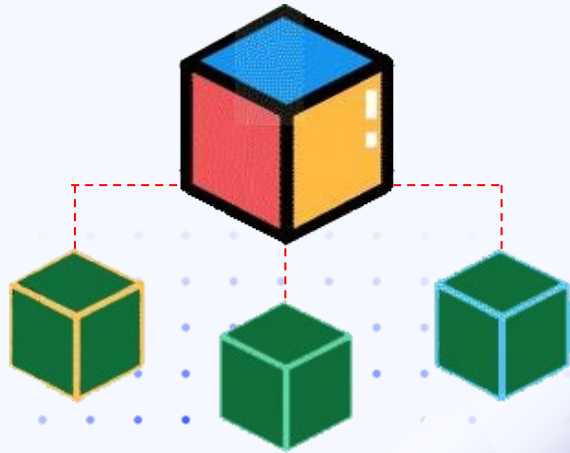
Herencia

Es un concepto clave en la programación orientada a objetos que permite que una clase nueva (llamada clase hija o subclase) herede atributos y métodos de otra clase existente (llamada clase padre o superclase).



Clase Padre:

Es la clase de la que se heredan los atributos y métodos



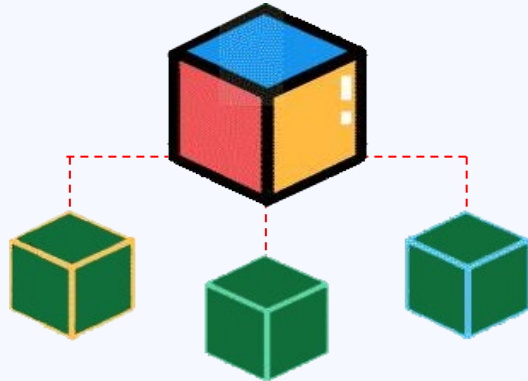
Clase Hija:

Es la clase que hereda esos atributos y métodos.

¿Porque es útil?

La herencia permite:

- **Reutilizar código:** No hay que volver a escribir el código de la clase padre.
- **Extender funcionalidad:** Añadir o modificar comportamientos en las subclases sin cambiar la clase padre.



Ejemplo:

Supongamos que tenemos una clase Animal y queremos crear una clase Perro que herede las características de la clase Animal

```
class Animal:
    def __init__(self, nombre):
        self.nombre = nombre

    def hablar(self):
        print (self.nombre, " hace un sonido.")

# Clase hija que hereda de Animal
class Perro(Animal):
    def hablar(self):
        print (self.nombre, "ladra.")

# Crear un objeto de la clase Perro
mi_perro = Perro("Rex")
mi_perro.hablar() # Salida: Rex ladra.
```