Tu es vraiment en train de **maîtriser la POO comme un(e) pro** — on entre dans un sujet **plus avancé et super important** :

**🧠 Concept #12 — Composition vs Héritage**

**🧬 L’Héritage (que tu connais déjà)**

On l’utilise quand une **classe enfant** hérite du **comportement d’une classe parent**.

python

CopyEdit

class Animal:

def manger(self):

print("Je mange.")

class Chien(Animal):

def aboyer(self):

print("Wouf !")

✔️ Utilise quand : **“est un”**

Un Chien **est un** Animal → ✅ Héritage

**🧱 La Composition**

La **composition**, c’est quand une classe **utilise une autre classe** **à l’intérieur d’elle-même**, comme **une pièce** dans une machine.

python

CopyEdit

class Moteur:

def demarrer(self):

print("Moteur démarré.")

class Voiture:

def \_\_init\_\_(self):

self.moteur = Moteur() # Composition

def demarrer\_voiture(self):

self.moteur.demarrer()

✔️ Utilise quand : **“a un”**

Une Voiture **a un** Moteur → ✅ Composition

**📊 Comparaison simple**

| **🧩 Critère** | **🧬 Héritage** | **🧱 Composition** |
| --- | --- | --- |
| Relation | "Est un" | "A un" |
| Couplage | Fort (plus rigide) | Faible (plus souple) |
| Réutilisation | Hérite des méthodes/attributs | Intègre un objet comme attribut |
| Modifications | Affectent tous les enfants | Limité à une seule classe |