Concept	Exemple	Description
Classe	`class Voiture:`	Modèle pour créer des objets, définit les attributs et les méthodes.
Objet	`ma_voiture = Voiture()`	Instance d'une classe avec attributs et méthodes spécifiques.
Instance	`chien = Chien()`	Objet spécifique créé à partir d'une classe, partageant les caractéristiques de la classe.
Attribut	`voiture.couleur = "rouge"`	Variable associée à une classe ou à un objet, représentant leurs caractéristiques ou données.
Méthode	`chien.aboie()`	Fonction associée à une classe ou à un objet pour effectuer des opérations spécifiques.
Encapsulation	Getters et setters	Regroupe données et/ou méthodes dans une classe pour cacher les détails d'implémentation. On encapsule les informations privées avec des méthodes publiques.
Héritage	`class Dacia(Voiture):`	Permet à une classe d'hériter des attributs et méthodes d'une autre classe, favorisant la réutilisation du code.
classe-mère	`class Voiture:`	C'est une classe qui possède une ou plusieurs classes qui en hérite. Elle est la classe-mère de ces classes uniquement.
classe-fille	`class Dacia(Voiture):`	C'est une classe qui hérite d'une autre classe. Elle est la classe-fille d'uniquement cette classe.
Polymorphisme		Extension de l'héritage, il permet à des objets de différentes classes de répondre de manière similaire ou différente à une même méthode.
Abstraction		Simplifie des concepts complexes en les modélisant à l'aide de classes et d'objets, cachant les détails d'implémentation. Il sert surtout à mutualiser le code en le mettant à un niveau plus élevé pour faciliter la maintenance.
Constructeur	`def __init__(self):`	Méthode spéciale d'une classe pour initialiser les objets lors de leur création.
Destructeur	`def __del__(self):`	Méthode spéciale d'une classe pour libérer des ressources ou effectuer des opérations de nettoyage.
Annotation	`@abstractmethod`	Commence par @ et se met au dessus d'une méthode. Elle permet de lui donner un fonctionnement externe précis.

> [!Tip] Différence entre "instance de classe" et "objet" Ceux sont des termes équivalents pour décrire un exemplaire spécifique d'une classe en programmation orientée objet.

> [!Warning] Les interfaces Les interfaces sont des objets abstraits non instantiable qui servent de contrat avec ceux qui l'implémentent. Ils doivent posséder toutes les méthodes et attribut de cette dernière.

> En python, il n'existe pas ce mot clé `Interface`, on doit utiliser un classe et des méthodes abstraite.