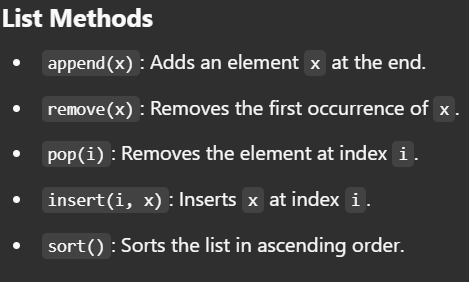
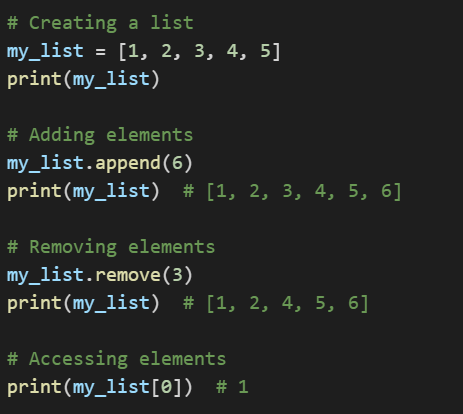
**Data Structures in Python**

Data structures are fundamental concepts used to store and manage data efficiently. Python provides several built-in data structures that help in organizing and manipulating data effectively.

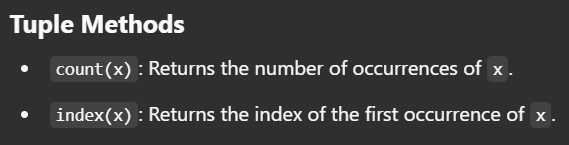
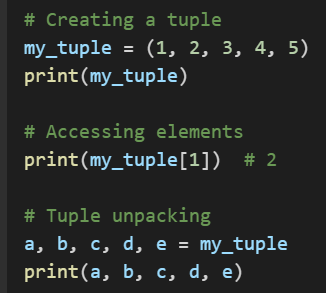
**1. Lists**

A list is an **ordered, mutable collection of elements**. Lists allow duplicate values and can store different data types.



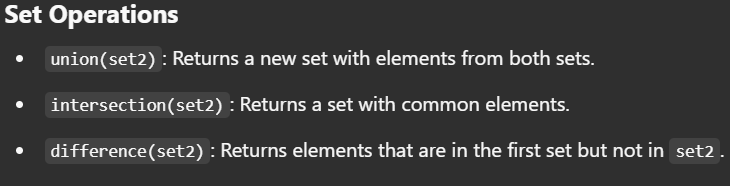
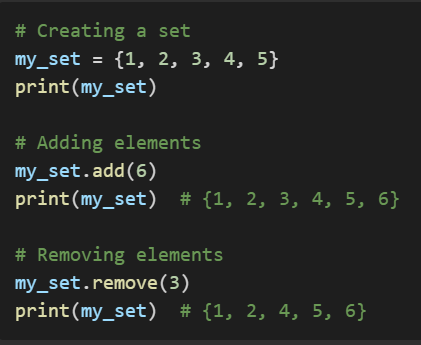
**2. Tuples**

A tuple is an ordered, immutable collection of elements. It allows duplicate values

****

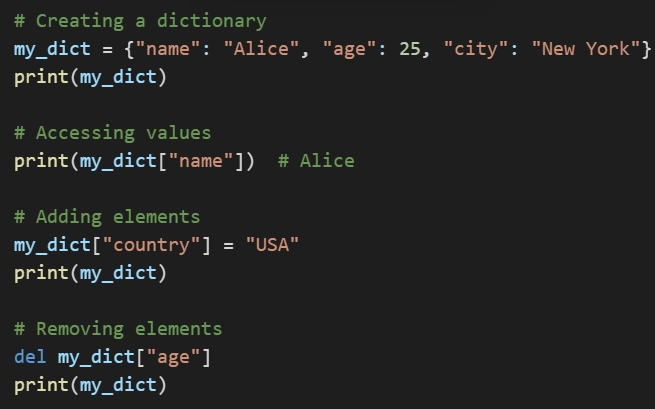
**3. Sets**

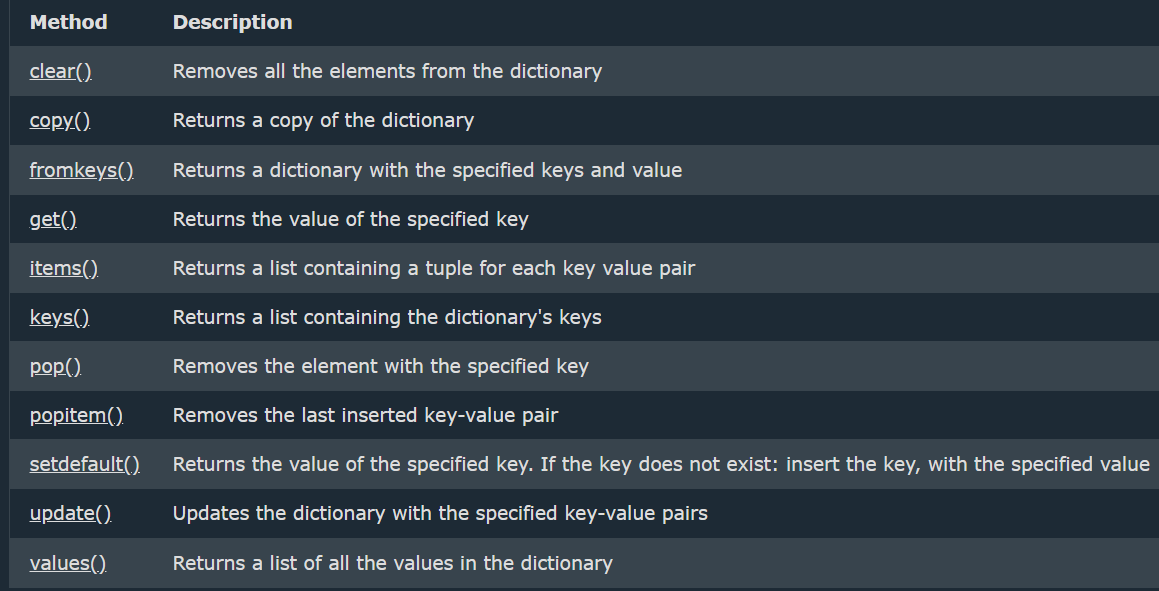
A set is an unordered, mutable collection of unique elements.

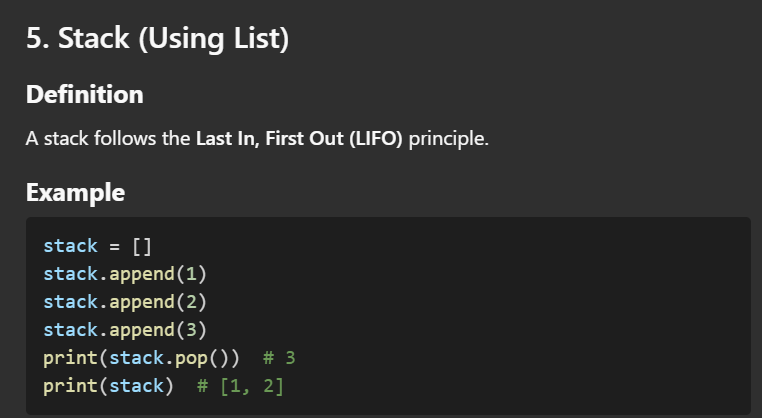


**4. Dictionaries**

A dictionary is an unordered collection of key-value pairs.





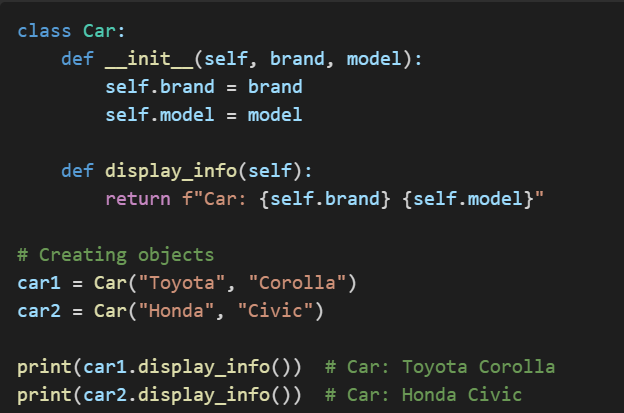


**Object-Oriented Programming (OOP) in Python**

Object-Oriented Programming (OOP) is a programming paradigm that organizes code into objects. Python supports OOP principles such as encapsulation, inheritance, polymorphism, and abstraction.

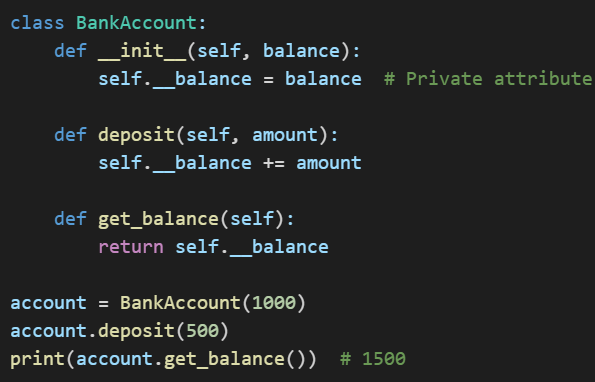
**1. Classes and Objects**

* **Class**: A blueprint for creating objects.
* **Object**: An instance of a class.



**2. Encapsulation**

Encapsulation restricts direct access to some object attributes, ensuring controlled modification.



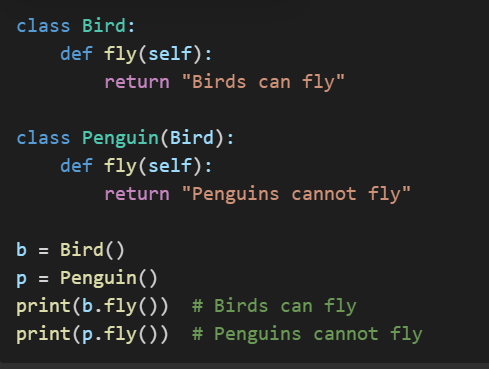
**3. Inheritance**

Inheritance allows a class to derive properties and methods from another class.



**4. Polymorphism**

Polymorphism allows different classes to define methods with the same name but different implementations.

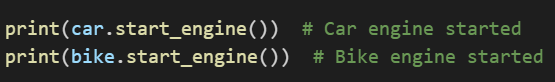
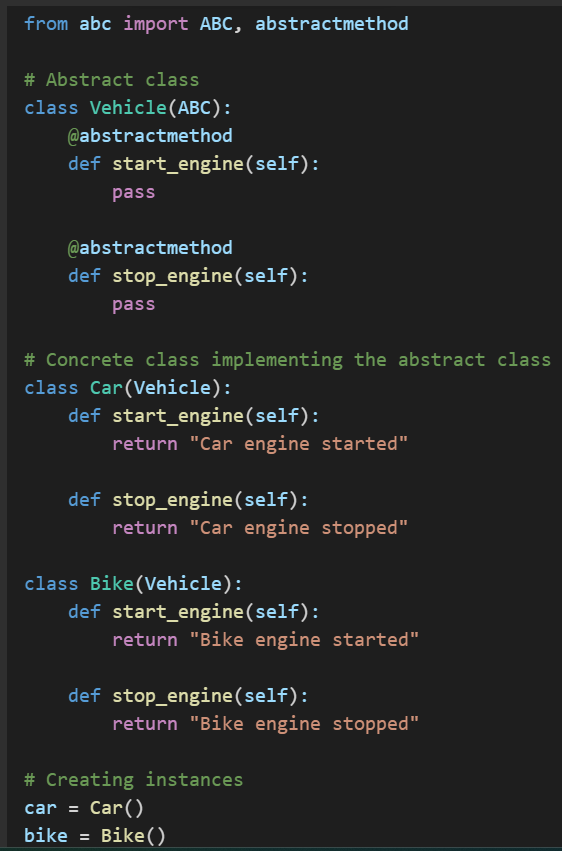


**5. Abstraction**

Abstraction is the process of hiding implementation details and exposing only the relevant features of an object. It is achieved in Python using abstract classes and methods, which are defined using the ABC module (Abstract Base Class).

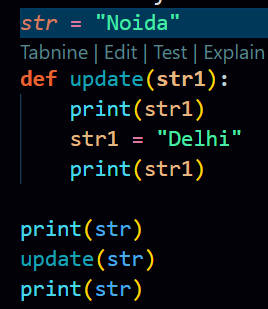
**Key Concepts**

* **Abstract Class**: A class that contains at least one abstract method.
* **Abstract Method**: A method declared but not implemented in the base class; it must be implemented in derived classes.



**Call by Value (Immutable Data Types)**

* Python does **not** support traditional call by value, but when passing immutable objects (like **int, float, str, tuple),i**t behaves similarly.
* A copy of the variable is passed, and modifications inside the function do not affect the original variable.



**Call by Reference (Mutable Data Types)**

When passing mutable objects (like list, dict, set), changes inside the function **affect the original object** because both the function parameter and the original reference point to the same memory location.