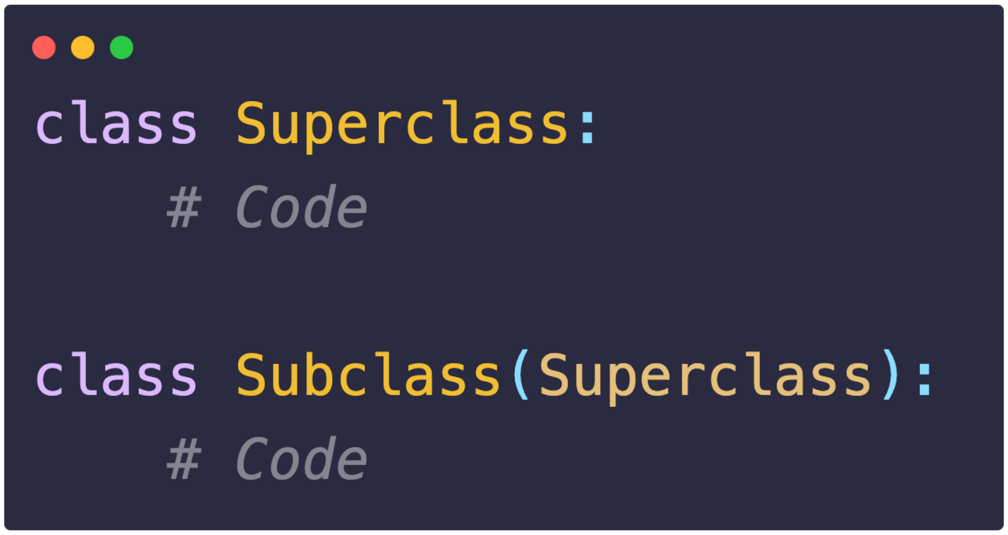
Inheritance in Python

Attributes

- ✦ **Inheritance** creates a hierarchical relationship between classes.
- ✦ It takes advantage of the natural hierarchies between objects and concepts by creating classes that "inherit" attributes and behaviors from other classes.
    - For example: A Car class could inherit from a Vehicle class. A Car "is a" type of Vehicle. Vehicle is more general and abstract than Car.
- ✦ A **Parent Class (Superclass)** is a class from which other classes inherit attributes and behaviors.
- ✦ A **Child Class (Subclass)** is a class that inherits attributes and behaviors from another class.
- ✦ A subclass "is a" type of its superclass.
- ✦ Subclasses automatically inherit the attributes of their superclasses (instance attributes and class attributes).
- ✦ The advantages of inheritance include reduced code repetition and more maintainable and scalable code.
- ✦ Subclasses can reuse code that was already written in the superclasses.
- ✦ You can create multi-level hierarchies in Python. This involves having multiple levels of classes that inherit from each other.
- ✦ Attributes are automatically inherited if you don't define the __init__() method in the subclass.
- ✦ To add a new attribute to the subclass, you need to define the __init__() method on the subclass and call the __init__() method of the superclass inside it (see Fig. 3).

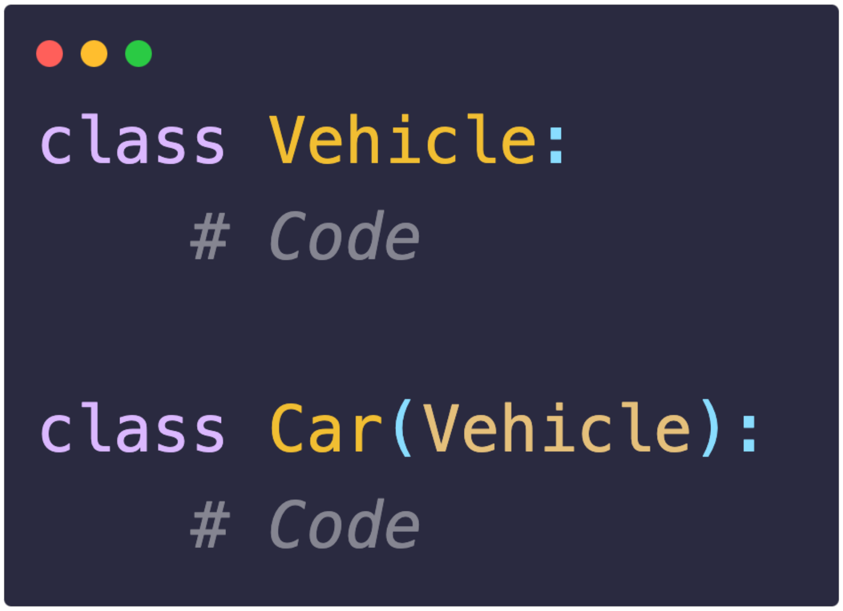
Fig. 1 Inheritance (General Syntax)



```python
class Superclass:
    # Code

class Subclass(Superclass):
    # Code
```

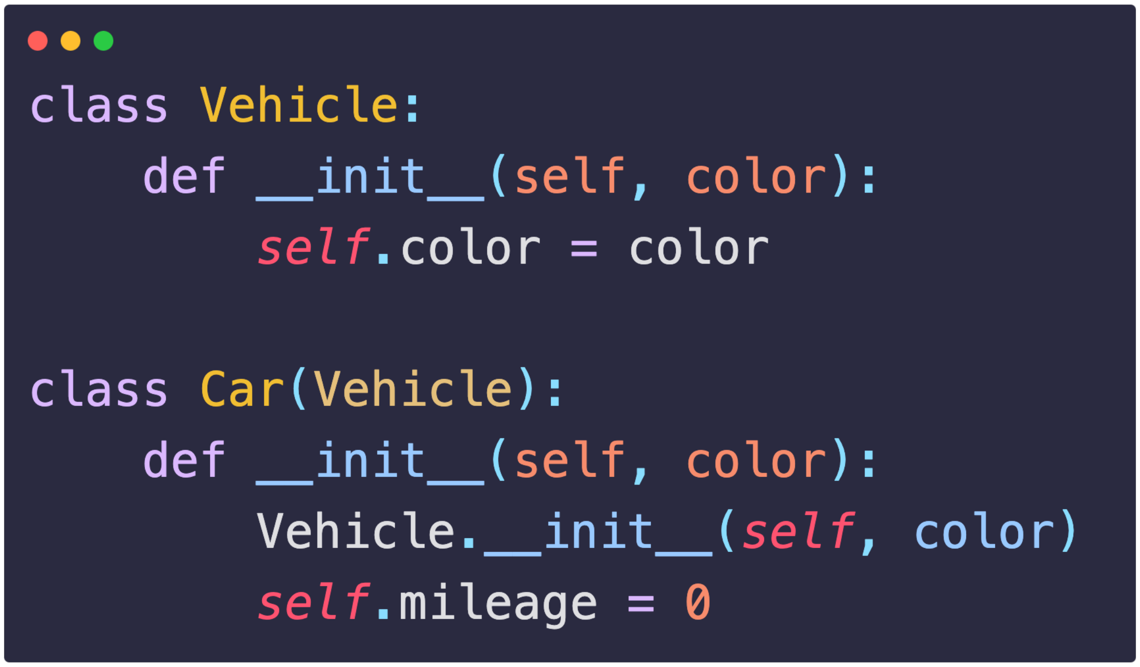
Fig. 2 Inheritance Syntax (Example)



```python
class Vehicle:
    # Code

class Car(Vehicle):
    # Code
```

Fig. 3 Inherit Attributes and Define New Attributes in the Subclass (Example)



```python
class Vehicle:
    def __init__(self, color):
        self.color = color

class Car(Vehicle):
    def __init__(self, color):
        Vehicle.__init__(self, color)
        self.mileage = 0
```