Inheritance is a fundamental concept in object-oriented programming (OOP) that allows you to create a new class (subclass or derived class) based on an existing class (base class or parent class). The derived class inherits attributes and methods from the base class and can also have its own additional attributes and methods. In Python, you can implement inheritance in several ways, including single inheritance, multiple inheritance, and multi-level inheritance. Let's explore each type with examples:

1. Single Inheritance:
- Single inheritance involves one subclass inheriting from a single base class. This is the most common form of inheritance.

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

    def speak(self):
        pass

class Dog(Animal):
    def speak(self):
        return f"{self.name} says Woof!"

class Cat(Animal):
    def speak(self):
        return f"{self.name} says Meow!"

dog = Dog("Buddy")
cat = Cat("Whiskers")

print(dog.speak())  # Output: Buddy says Woof!
print(cat.speak())  # Output: Whiskers says Meow!
```

2. Multiple Inheritance:
- Multiple inheritance involves a subclass inheriting from more than one base class. In Python, this is supported, but it can lead to complex class hierarchies.

```python
class Parent1:
    def method1(self):
        return "Method 1 from Parent1"

class Parent2:
```

```
    def method2(self):
        return "Method 2 from Parent2"

class Child(Parent1, Parent2):
    pass

obj = Child()
print(obj.method1())  # Output: Method 1 from Parent1
print(obj.method2())  # Output: Method 2 from Parent2
```

3. Multi-level Inheritance:
- Multi-level inheritance involves a subclass inheriting from another subclass, creating a chain of inheritance.

```python
class Grandparent:
    def grandparent_method(self):
        return "Method from Grandparent"

class Parent(Grandparent):
    def parent_method(self):
        return "Method from Parent"

class Child(Parent):
    def child_method(self):
        return "Method from Child"

child = Child()
print(child.grandparent_method())  # Output: Method from Grandparent
print(child.parent_method())       # Output: Method from Parent
print(child.child_method())        # Output: Method from Child
```

In these examples, you can see how inheritance works in Python with different types:

- **Single Inheritance:** The `Dog` and `Cat` classes inherit from the `Animal` base class, and they override the `speak` method to provide their own implementation.

- **Multiple Inheritance:** The `Child` class inherits from both `Parent1` and `Parent2`, so it has access to methods from both parent classes.

- **Multi-level Inheritance:** The `Child` class inherits from `Parent`, which in turn inherits from `Grandparent`. This creates a multi-level hierarchy where `Child` can access methods from both `Parent` and `Grandparent`.