Practical No 3

Aim: write a program to implement different method to do polymorphism , operator overloading and overriding in Python

1. Polymorphisim

This code demonstrates polymorphism in Python. It defines two classes, `Dog` and `Lizard`, with identical method names (`tellAboutYou` and `limbs`) but different implementations.
The `printObject` function takes an object as an argument and calls these methods on it, without knowing the object's class. This allows for generic behavior, as shown when `printObject` is called with instances of both `Dog` and `Lizard`.

2. Duck type

Duck typing is a concept in programming where you don't care about the actual type of an object, as long as it has the methods or attributes you need.
Key characteristics:

1. **No explicit type checking**: You don't check the type of an object before using it.
2. **Focus on behavior**: You care about what the object can do (its methods), not what it is (its type).
3. **No inheritance required**: Objects don't need to inherit from a common base class to be used in a certain way.

3. Method Overriding

This code demonstrates method overriding in Python.
The `Child` class inherits from the `Parent` class and overrides the `call` method.
When `child.call()` is called, it prints "I Am Inside Child", instead of calling the `call` method from the `Parent` class.
However, the `super().__init__()` in the `Child` class's **`init`** method calls the **`init`** method from the `Parent` class, which sets `self.name` to "Parent". This is then immediately overridden by `self.name = "Child"`.
So, when `child.call()` is called, it prints "I Am Inside Child", not "I Am Inside Parent".

4. Operator Overloading

This code snippet demonstrates operator overloading in Python.
The `Base` class has an **`add`** method that allows objects of this class to be added together using the `+` operator. When two `Base` objects are added, their `data` attributes are added together.
The code then creates two `Base` objects with `data` values 10 and 20, adds them together, and prints the result (30).
After that, it prompts the user to enter their first and last names, and then prints the full name by concatenating the two input strings with a space in between using the `+` operator.