Lesson: Polymorphism

Description: Discover how polymorphism makes code more dynamic and flexible.

Polymorphism is the ability of an object to take on many forms. In OOP, polymorphism allows methods to perform different tasks based on the object calling them. This makes the code more flexible and easier to extend.

Types of Polymorphism:
- **Compile-time Polymorphism** (Method Overloading): Multiple methods in the same class with the same name but different parameters.
- **Run-time Polymorphism** (Method Overriding): A method in the child class overrides a method in the parent class.

Example:

```
class Shape:
    def area(self):
        pass

class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return 3.14 * self.radius * self.radius

class Rectangle(Shape):
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def area(self):
        return self.length * self.width

shapes = [Circle(5), Rectangle(4, 6)]
for shape in shapes:
    print(f'Area: {shape.area()}')
```

Exercises:

1. Implement polymorphism using a base class for employees and derived classes for different job roles.
2. Write a program to demonstrate method overloading in Python.