Practical Python Exercise — Functions and Classes

Exercise 1: Basic Function

Create a function **power(base, exp)** that returns **base**$^{\text{exp}}$.
Then write another function **sum_of_powers(numbers, exp)** that uses it to return the **sum** of all numbers raised to **exp**.

Exercise 2: Using Lambda and map/filter

Use **lambda** and **map()** to convert a list of temperatures in **Celsius** to **Fahrenheit**.

Formula: $F = \left(C \times \frac{9}{5}\right) + 32$

Exercise 3: Closure Function

Create a closure called **multiplier(n)** that returns a function which multiplies its input by **n**.

Exercise 4: Define a Simple Class

Create a class **Student** with:

- **Attributes**: name, age, grade
- **Method**: display_info() to print all details

Exercise 5: Add Methods and Behavior

Add a method **update_grade(new_grade)** that modifies the student's grade and confirms the update.

Exercise 6: Create Parent and Child Classes

- Create a parent class **Person** with attributes: name, age.
- Create a subclass **Teacher** that **inherits** from **Person** and adds an attribute subject.
- Use super().__init__(name, age) in Teacher.
- Add a method introduce() in both classes that displays information differently.

Exercise 7: Override and Extend

Add a method **work()** in **Person** that prints "Doing something generic...".
Override it in Teacher to print "Teaching students...".

Call **work()** for both classes.

Exercise 8: Shared Interface, Different Behavior

Create three classes: **Dog**, **Cat**, and **Bird**, each with a **speak()** method.

Exercise 9: Polymorphism with Shapes

Define a base class **Shape** with an **area()** method (that raises an exception).
Then create subclasses:

- Circle(radius)
- Rectangle(width, height)

Each should override **area()** correctly.
Then create a list of shapes and print all their areas using one loop.

Exercise 10: School Management System

Design a simple structure for a school:

1. Create a base class Person with attributes name, age.
2. Create two subclasses:
    - Student(Person) with extra attribute grade
    - Teacher(Person) with extra attribute subject
3. Each class should have an introduce() method with its own print style.
4. Create a list containing both teachers and students, and loop through calling introduce().