Exercise 1.5: Object-Oriented Programming in Python

Learning Goals

- Apply object-oriented programming concepts to your Recipe app

Reflection Questions

1. **In your own words, what is object-oriented programming? What are the benefits of OOP?**
   Object-oriented programming (OOP) is a programming paradigm in computer science that relies in on the concept of objects and classes.
   OOP is used to structure a software program into simple, reusable pieces of code blueprints (usually called classes), which are used to create individual instances of objects.
   Python is one of the many object-oriented programming languages, although is also allows other programming “styles”.
   Some of the benefits of OPP are:
   - Modularity for easier troubleshooting.
   - Reuse of code through inheritance.
   - Flexibility through polymorphism.
   - Effective problem solving.

2. **What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work.**
   Almost everything in Python it’s an object, with its properties and methods. An object is a collection of data (variables) and functions (methods). A class is a blueprint of objects. We can think of a class as a prototype of a product that defines the details the product may have.
   For example, we can define a class named Car that defines the details about the car (*e.g.* brand, model, color, year, etc.). From the class we can create an object named car1. Since many cars share the same details , we can define many objects from a class (*e.g.* car2, car3, etc.).

3. **In your own words, write brief explanations of the following OOP concepts; 100 to 200 words per method is fine.**

Method	Description
Inheritance	In Python a class can inherit properties from another class. The class being inherited from is called the parent class or base class, and the class to which all data attributes and procedures are copied over is known as subclass or inherited class. Inheritance only works in one direction (from parent to subclass). The subclass will contain its own unique attributes, but also receives attributes from the parent class (through inheritance).

Polymorphism	A given data attribute or method has the same name across different classes or data types, but performs different operations depending on where it was defined.
Operator Overloading	To use special operators supported by Python’s built-in classes on a custom class you have to define your own methods. All it requires is defining a function with a name that Python already reserves for your operator, and surrounding it with double underscores (*e.g.* __sum__()).