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?

OOP is a programming paradigm that is based on the concept of "objects", which can contain data and code that manipulates that data. The objects are instances of classes, which are templates or blueprints for creating objects.
In my opinion, the most important benefit of using OOP is reusability: OOP allows for the reuse of code through the use of classes and objects, making development faster and more efficient.

2. What are objects and classes in Python? Come up with a real-world example to illustrate how objects and classes work.

In Python, objects are instances of classes. A class is a template or blueprint for creating objects, and it defines the properties and methods that the objects created from it will have.
Imagine you're creating a program to simulate a pet store. One class you could create is the "Pet" class, which would have properties such as name, age, and breed, and methods such as speaking and eating. Each pet in the store would be an object created from the "Pet" class. For example, you could create an object called "dog1" that represents a specific dog with the name of "Fido", an age of 3, and a breed of "Golden Retriever". The "dog1" object would have access to the properties and methods defined in the "Pet" class, such as the ability to speak("bark") and eat("dog food").
You could also create another object called "cat1" with different property values, but still having the same methods as "dog1" because they are both instances of the same class "Pet".

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

Method	Description
Inheritance	This is the mechanism of deriving a new class from an existing one. The child class inherits the properties and methods of the parent class.
Polymorphism	This is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
Operator Overloading	This is a feature that allows operators (such as +, -, *, etc.) to have different meanings depending on the context in which they are used. This is achieved by creating special methods in a class that are called when an operator is used with an instance of that class.