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 computer programming model that organizes code around objects that contain data. The benefits of OOP include an easier method of code organization, easier maintenance and debugging, enhanced code reusability, and increased productivity as inheritance can be used.

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

A class is a user defined blueprint or prototype that is then used to create objects in Python. The class provides a way to bundle data and functionality together, once created classes and then able to create a new type of object following the blueprint. The Object created by a class is an instance of that class, it no longer has theoretical data, but it has actual values. Say you worked at a dog grooming salon and wanted to keep track of the dogs your clients brought in. You could create a class that allows you to store information about each dog, defining attributes such as breed, age, weight, treat preference, and typical haircut. Everytime a new dog came in, you could use the class to create an object containing all the important facts about the dog. Once created, this dog now becomes a searchable object that can be reviewed whenever needed in the future. Allowing your staff to identify each dog and ensure that they execute the right haircut every time.

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

Method	Description
Inheritance	Mechanism of basing an object or class upon another. Allows you to use existing properties and behaviors that have been written into a parent class/object and apply them to a child without re-writing code. You can further customize the child but don't need to re-write all of the initial code to establish the object.
Polymorphism	The ability of any data to be processed in more than one form. This is a key power of object-oriented programming. It allows objects to take on many different forms in

	different instances and is achieved through inheritance, interfaces, and method overriding.
Operator Overloading	It is the process of defining your own methods within a custom class and associating them with a specific operator. It is necessary to use operators on a custom class in Python, if not defined, you will run into a TypeError with the operators stated that they are not supported.