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 coding that helps keep your code non-repetitive, non-redundant, and efficient. With this you are benefiting in making your code cleaner and easier to read. You will be able to code faster and be able to find any errors within the code more quickly.

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

Objects consist of data contained in Python. Classes are templates containing the objects. A real-world example would be to consider a wristwatch to be a class. A Rolex or Omega would be considered an object within that class.

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

Method	Description

Inheritance - Whenever you write code it is possible that certain properties of class you have created can be used by another class. Inheritance makes this possible. It lets you "inherit" methods from one class and use it in another. Inheritance only works in one direction from a parent to a subclass.

Polymorphism - Whenever you name an attribute or method it performs a certain operation within that class. With polymorphism that method or attribute can have the same name across different classes or data types but perform a different operation for

each one. So even though multiple classes can have a method with the same name, each method will perform its own implementation for its respected class.

Operator Overloading – Python does not support the use of an operator (+, -, *, /) in a custom class. Due to this you need to define your own method for them if you plan on using. Fortunately, Python has already set aside names for each operator. All you need to do is surround the name Python has reserved with double underscores. For example to add your code would look like "__add__()". To subtract your code would look like "__sub__()".