Learning Journal 1.5

1. In your own words, what is object-oriented programming? What are the benefits of OOP?

   Object-oriented programming is a type of coding that organizes code around objects, which are instances of classes. It allows for the development of modular and reusable code structures in which the class acts as a template to define common attributes of a group of objects. The benefits of OOP are code reusability, modularity, encapsulation, inheritance, and polymorphism all which promote better code organization, flexibility, and maintainability.

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 an instance of a class. A class is a template that defines the structure of objects. It specifies the attributes and methods that the objects created from the class will have.

```
class Car:
        def __init__(self, make, model, year):
                self.make = make
                self.model = model
                self.year = year
                self.is_running = False

        def start_engine(self):
                self.is_running = True
                print(“The engine is now running.”)

        def stop_engine(self):
                self.is_driving = False
                print(“The engine has stopped running.”)

my_car = Car(“Toyota”, “Highlander”, 2022)
my_car.start_engine()
print(my_car.is_running)  #True

your_car = Car(“Honda”, “Accord”, 2018)
your_car.start_engine()
print(your_car.is running)  #True

my_car.stop_engine()
print(my_car.is_running)
print(my_car.is_running)  #False
```

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

Method	Description
Inheritance	Inheritance is a fundamental concept in OOP that allows a class to inherit properties and behaviors from another class known as the base class. The class that inherits these characteristics is known as the subclass. Inheritance allows code reuse and promotes the concept of generalization and specialization. The subclass can inherit from the superclass and add its own attributes or override the inherited methods. This promotes modularity, reduces duplicate code, and allows for better organization and code maintenance.
Polymorphism	Polymorphism is a feature in OOP that allows objects of different classes to be treated as objects of a common superclass. Polymorphism allows for the use of a single interface to represent different types of objects. It enables methods to be written in a generic way that can work with objects of multiple classes, if they implement the same set of methods or have a common superclass. This provides flexibility and extensibility in the code, as new classes can be added without modifying existing code.
Operator Overloading	Operator Overloading is the ability to customize the behavior of operators such as +, -, *, /, ==, <, > for objects of a class. It allows objects to respond to operators in a way that is meaningful in the context of the class. By overloading operators, we can define how objects of a class should behave when certain operators are used on them. Operator Overloading provides a natural syntax for working with objects and allows me to write more expressive and readable code.