### Exercise 1.5: Object-Oriented Programming in Python

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

In OOP, everything is treated as object which refers to class which is like a template for creating instances of individual objects. OOP helps to keep the code non-repetitive, clean and organized.

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

In Python, everything is an object containing different data types and methods to interact with the data. Each objects belong to a class, like a template describing internal structure. For example, ‘Phone’ is a class and it can have individual models or objects containing its own data and interacting methods such as model name, released year, operations on pressing a certain button and more.

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

|  |  |
| --- | --- |
| **Method** | **Description** |
| Inheritance | Data attributes and methods in a class can copied over another class, this is called ‘inheritance’ in OOP. A class that’s inherited from is called parent class and the class that’s inheriting is called sub class or child class. Inheritance works in only one direction, from parent to child |
| Polymorphism | Polymorphism is where given data attributes or methods have the same name across different classes or data types but performs different operations depending on where it was defined. |
| Operator Overloading | Operator overloading is a process for enabling use of operators such as + and – on a custom class by defining own methods with the name Python already reserves for the symbol, surrounding with double underscores |