**Object-Oriented Programming**

Laboratory Activity No. 1

**Review of Technologies**

*Submitted by:*

**Hermosura, Leigh B.**

**SAT 5:00pm – 8:30pm / BSCpE 1-A**

*Submitted to*

**Engr. Maria Rizette H. Sayo**

Instructor

*Date Performed:*

**18-01-2025**

*Date Submitted*

**18-01-2025**

I. Objectives

In this section, the goals in this laboratory are:

* To define the key terms in Object-oriented programming
* To be able to know the construction of OO concepts in relation to other types of programming such as procedural or functional programming

GeeksforGeeks (2024) defined paradigm as a method or set of knowledge of a certain way to solve tasks and problems. In essence, programming paradigm is a method or approach that we follow in order to solve problems using programming language, tools and techniques that are available to us. Object-Oriented Programming (OOP) is a programming paradigm under computer science that is reliant on the concept of classes and. It is utilized to organize a software program into straightforward, reusable code blueprints (sometimes referred to as classes) that are used to generate distinct instances of objects. Numerous programming languages are object-oriented, such as JavaScript, C++, Java, and Python.

Object-oriented programming offers a way to organize programs so that properties and behaviors are combined into individual objects. For instance, an object could represent a person with properties like name, age, and address as well as behaviors like walking, talking, breathing, and running. Also, it can represent an email containing properties like a recipient list, subject and body as well as behaviors like sending and attaching files (Amos, 2024).

Amos (2024) also highlighted that OOP is found in other programming languages as well, and it is frequently explained in terms of its four pillars, or tenants:

1. **Encapsulation**enables you to group data (attributes) and behaviors (methods) together within a class, forming a unified structure. By setting up methods to manage access to and modification of attributes, encapsulation ensures data integrity and supports the creation of modular and secure code.
2. **Inheritance**allows classes to be organized in a hierarchy, where a subclass can inherit attributes and methods from a parent class. This encourages code reuse and minimizes repetition.
3. **Abstraction**involves hiding the internal workings of an object and revealing only its necessary functions. By establishing a clear interface, abstraction makes it easier to interact with objects, allowing developers to focus on what an object does rather than how it works.
4. **Polymorphism**allows you to treat objects of different types as instances of the same base type, as long as they implement a common interface or behavior. Python’s duck typing make it especially suited for polymorphism, as it allows you to access attributes and methods on objects without needing to worry about their actual class.

II. Methods

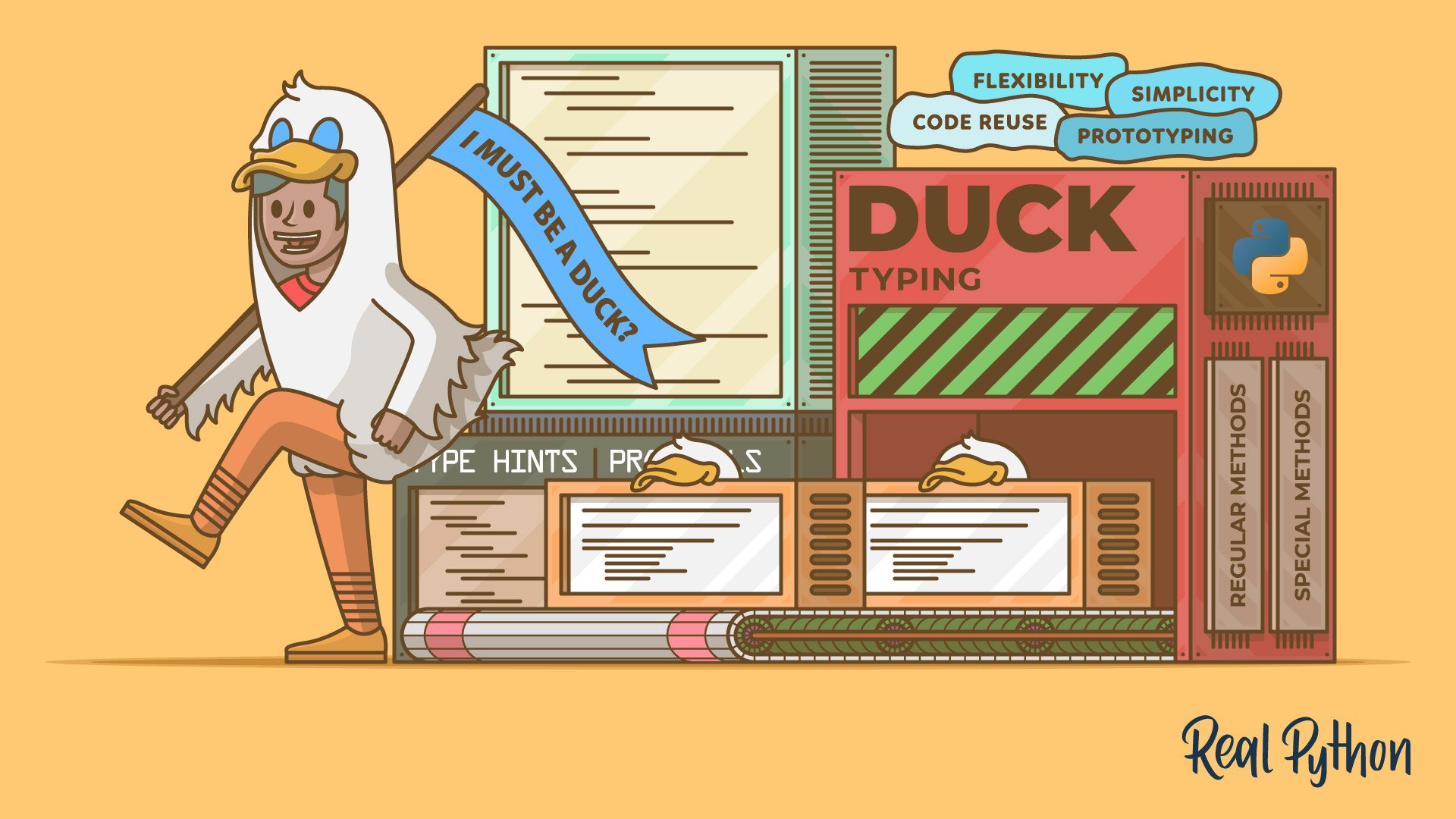
**Classes** Classes offer a way to group data and functionality into one unit. When a new class is created, it defines a new object type, enabling the creation of multiple instances of that type. Each instance can have attributes that represent its state, and it can also have methods (defined by the class) to alter its state (Python, n.d.).

**Objects** While a class serves as a blueprint, an instance is a specific copy of that class containing actual values. Python is an object-oriented programming language that focuses on objects, meaning it emphasizes functions. Python objects are essentially a combination of data variables and methods that operate on that data, bundled together into a single entity (GeeksforGeeks, 2023).

Classes are made up of **fields** (attributes) and **methods** (behaviors). For instance, in a superhero class like Iron Man, fields could include name and suit color, while methods could be actions like flying and shooting lasers (Ahmad 2024).

In object-oriented programming, **properties** can also be viewed as functions. A property contains a function that allows its procedures or variables to be modified without directly editing the code. Properties can be updated or changed based on user input, enabling the creation of highly interactive programs and applications.

III. Results



**Figure 1**. Duck typing in Python

Retrieved January 18, 2025 from https://realpython.com/duck-typing-python/

Duck typing is a type system in which an object is deemed compatible with a particular type if it possesses all the necessary methods and attributes required by that type. Hence the saying, “If it walks like a duck and it quacks like a duck, then it must be a duck” (Ramos 2024).

IV. Conclusion

In conclusion, Object-Oriented Programming (OOP ) provides an effective way to organize and manage code by emphasizing the use of classes, objects, and their relationships. By combining data and behaviors within individual objects, OOP makes it easier to build modular, reusable, and scalable programs. The four main principles of OOP encapsulation, inheritance, abstraction, and polymorphism further enhance the flexibility and efficiency of software development. Additionally, concepts like duck typing allow for dynamic code that interacts with objects based on their behavior rather than their specific type. Ultimately, OOP streamlines the development process and serves as a foundation for building complex, interactive applications.

**Reference**

Book

[1]

Website

[2]

<https://www.geeksforgeeks.org/introduction-of-programming-paradigms/>

<https://realpython.com/python3-object-oriented-programming/>

<https://docs.python.org/3/tutorial/classes.html>

<https://www.geeksforgeeks.org/python-object/>

<https://www.techtarget.com/whatis/definition/method>

<https://www.designgurus.io/blog/object-oriented-programming-oop>

<https://realpython.com/duck-typing-python/>