**UNIVERSITY OF CALOOCAN CITY**
*Caloocan, 1400 Metro Manila, Philippines*

**COLLEGE OF ENGINEERING**
**Computer Engineering**
*2nd Semester, School Year 2024-2025*

# Object-Oriented Programming

Laboratory Activity No. 1

## Review of Technologies

*Submitted by:*

**Directo, Hannah Thea B.**

**Saturday/ BSCPE 1-A**

*Submitted to*

**Engr. Maria Rizette H. Sayo**

Instructor

*Date Performed:*

**18-01-25**

*Date Submitted*

**18-01-25**

## 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

## II. Methods

### 1. Classes

In Python, we use classes to create objects. A class is a tool, like a blueprint or a template, for creating objects. It allows us to bundle data and functionality together. A class defines a set of attributes and methods that the created objects (instances) can have. (P. Team, 2021). A class is a user-defined data type. It consists of data members and member functions, which can be accessed and used by creating an instance of that class. It represents the set of properties or methods that are common to all objects of one type. (GeeksforGeeks, 2023).

### 2. Objects

It is a basic unit of Object-Oriented Programming and represents real-life entities. An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated. An object has an identity, state, and behavior. Each object contains data and code to manipulate the data. Objects can interact without having to know details of each other's data or code, it is sufficient to know the type of message accepted and type of response returned by the objects. (GeeksforGeeks, 2023). The state of an object is represented by its attributes, reflecting its characteristics. The behavior is shown through the object's methods, which define how it interacts with other objects. Identity is established by assigning a unique name to the object, allowing it to engage with other objects in the system. (P. Team, 2021)

## 3. Fields

Field is a language-specific term for instance variable, that is, an attribute whose value is specific to each object. There are various types of fields to represent the data that we want to store. First, Init-only fields are those that can only be assigned a value during the initialization of the object, ensuring that certain attributes remain immutable. Mutable default fields, on the other hand, have a default value that can be modified later, offering flexibility in how the object is configured after initialization. Lastly, class-level fields are shared across all instances of the class, making them useful for storing information that is common to every instance of the class. (Sling Academy, 2023)

## 4. Methods

In object-oriented programming (OOP), a method is a programmed procedure that is defined as part of a class and is available to any object instantiated from that class. Each object can call the method, which runs within the context of the object that calls it. This makes it possible to reuse the method in multiple objects that have been instantiated from the same class. Method defined as a function inside a class. A method is an action that an object can perform. (R. Sheldon, 2023)

## 5. Properties.

In programming, properties are a mechanism used to manage the values of variables within objects, also referred to as fields or attributes. Instead of directly accessing a variable, properties provide a way to set or retrieve its value through special methods known as getters and setters. A getter is a method that retrieves the value of a field, while a setter is a method that sets or modifies the value of that field. Properties provide an abstraction for accessing and modifying an object's state, helping enforce rules and making the code cleaner, more maintainable, and easier to understand. [1]. Property is an attribute of an object. We can also say the variables inside a class are called properties. (Shoutsdev, 2020)

## III. Results

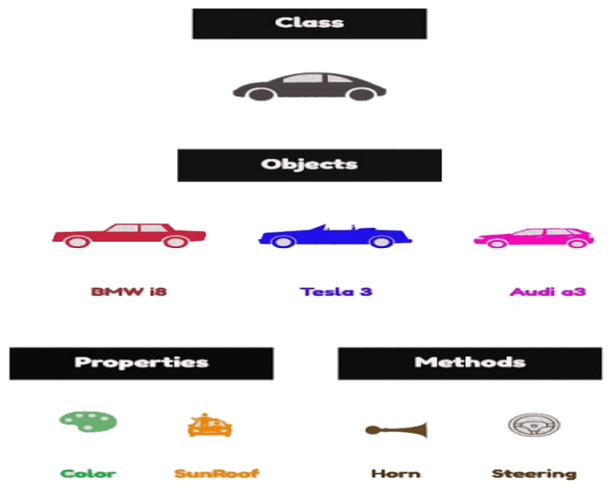


*Figure 1.* Understanding OOP Concepts: Properties and Methods - Shouts.dev

The image shows a visual representation of the concept of Object-Oriented Programming (OOP) using the analogy of a car. At the top, the "Class" section, represented by a generic car icon, symbolizes a class, which serves as a blueprint or template for creating objects. A class defines the properties and behaviors that its objects will have. Below this, the "Objects" section showcases specific instances of the class, such as BMW i8, Tesla 3, and Audi a3, each represented in different colors. These objects are individual examples created from the car class, each possessing unique attributes while sharing the same overall structure. After that, the figure highlights "Properties" and "Methods," which are essential aspects of objects in OOP. Properties, represented by icons such as a paint palette and a sunroof, describe the characteristics of an object, such as its color and whether it has a sunroof. Finally, methods, depicted by a horn and a steering wheel, define the actions or behaviors that an object can perform, such as honking the horn or steering the vehicle. In summary, this figure illustrates how a class provides the framework for objects, which have specific properties and methods that define their behavior and characteristics.

## IV. Conclusion

The fundamental concepts of Object-Oriented Programming (OOP) are crucial for developing efficient and organized software. Classes serve as blueprints for creating objects, encapsulating data and behavior that define an entity (P. Team, 2021). Objects, as instances of classes, embody real-world entities with distinct identities, states, and behaviors (GeeksforGeeks, 2023). Fields, also known as instance variables, store object-specific data and can be categorized based on their mutability and scope, offering flexibility and control over data management (Sling Academy, 2023).
Methods provide functionality to objects, enabling them to perform actions and interact with other objects in a systematic manner (R. Sheldon, 2023). Properties, managed through getters and setters, facilitate controlled access to an object's internal state, promoting data encapsulation and integrity (Shoutsdev, 2020).

The visual representation of these concepts through the analogy of a car highlights the relationship between classes, objects, properties, and methods. It represents how a class defines a general structure while objects show specific implementations with unique attributes and behaviors.

In summary, mastering OOP principles, including classes, objects, fields, methods, and properties, empowers developers to build scalable, maintainable, and modular software systems.

## Reference


GeeksforGeeks, “Python OOPs Concepts,” *GeeksforGeeks*, Dec. 13, 2024. https://www.geeksforgeeks.org/python-oops-concepts/

GeeksforGeeks, “Introduction of object oriented programming,” *GeeksforGeeks*, Feb. 09, 2023. https://www.geeksforgeeks.org/introduction-of-object-oriented-programming/

“Python: Defining Fields in Dataclass - Sling Academy.” https://www.slingacademy.com/article/python-defining-fields-in-dataclass/

P. Team, “Classes in Python with Examples,” *Python Geeks*, Jun. 2021. https://pythongeeks.org/classes-in-python/

R. Sheldon, “method (in object-oriented programming),” *WhatIs*, Feb. 22, 2023. https://www.techtarget.com/whatis/definition/method

“Understanding OOP Concepts: Properties and Methods,” *Shouts.dev*, Jan. 21, 2023. https://shouts.dev/articles/understanding-oop-concepts-properties-and-methods/#properties