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:
Gonzales, Aaron Gabriel D.
Saturday 4:30-8:30 PM / BSCpE-1A

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

II. Methods

General Instruction:

A. Define and discuss the following Object-oriented programming concepts:

1. Classes – It is a custom data type created by users that serves as a blueprint for organizing data and behavior. It defines attributes (like a superhero's name or suit color) and methods (like flying or shooting lasers). Objects are instances created from this blueprint, each with its own unique data but sharing the same structure and functionality. Essentially, a class outlines the shared properties and actions of a group of objects.

2. Objects – It is a core concept in Object-Oriented Programming, representing real-world entities or abstract ideas. It is created as an instance of a class, which acts as its blueprint. While defining a class doesn't use memory, creating an object does. Each object has an identity, state (attributes), and behavior (methods). Objects store data and include code to work with it, and they can interact by exchanging messages without needing to know each other's internal details. For example, ironman could be an object created from the IronMan class.

3. Fields – It is also called attributes or variables, are used to store information within an object or class. They define the qualities or features of an object and can hold different types of data, like numbers, text, or custom objects. Fields represent the data that objects work with or change.

4. Methods - They are functions within a class that define how objects behave. They perform actions, manipulate data, and enable objects to communicate, making systems more organized and modular. Objects use these methods to carry out tasks or interact with other objects. For example, in an IronMan class, a fly() method lets an object like ironMan perform the action of flying without changing its attributes. Methods can also modify attributes, like an updateArmorStrength() method increasing Iron Man's armor strength. By keeping functionality within objects,

methods improve reusability and simplify debugging. Attributes like _armorStrength are often protected, meaning they should only be changed through dedicated methods.

5. Properties - There are 4 Pillars of OOP:

   Encapsulation, one of the main principles of object-oriented programming (OOP), focuses on keeping an object's internal details hidden from the outside. Instead of directly accessing an object's data, you interact with it through specific methods. This not only safeguards the object's internal structure from outside interference but also makes the program more secure, easier to maintain, and less prone to bugs.

   Inheritance is a key idea in object-oriented programming (OOP). It lets one class, known as the subclass or child class, take on the properties and methods of another class, called the superclass or parent class. This makes it easier to reuse code and creates a connection between the parent and child classes.

   Abstraction is an important principle in object-oriented programming (OOP) that simplifies things by hiding the inner workings and showing only what’s essential. It helps reduce complexity by breaking the application into different layers. The idea is to focus on how objects work together to perform a task, without needing to know all the details of how they do it.

   Polymorphism, which comes from the Greek words for "many" (poly) and "forms" (morph), is a key idea in object-oriented programming (OOP). It allows objects from different classes to be treated as if they belong to the same class, thanks to inheritance. In simple terms, polymorphism lets one interface handle a variety of actions. A common example is using a reference from a parent class to work with objects of a child class. This adds flexibility and dynamic behavior to OOP.

III. Results

Python is a versatile language used in web development with frameworks like Django and Flask, creating desktop applications with tools like Tkinter and Kivy, and powering scientific and numeric calculations with libraries like SciPy and Numpy. It's at the core of data science, enabling data analysis, machine learning, and visualization with libraries like Pandas, TensorFlow, and Matplotlib. Python simplifies software development by automating workflows and managing testing, while also excelling in education as an easy-to-learn language. Businesses use Python for ERP systems and platforms like Reddit and YouTube. It's also popular for game development, 3D graphics, network programming, and database access with tools like PyMySQL and PyMongo. Python's simplicity, flexibility, and vast libraries make it indispensable in many fields.

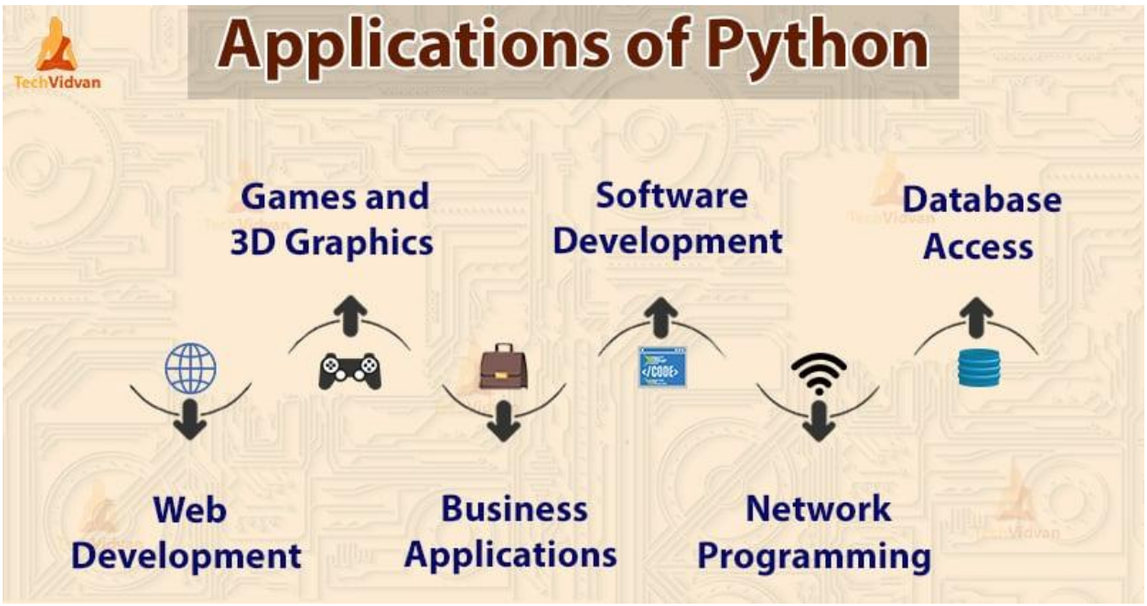

Figure 1: Applications of Python

Sources: Python Applications - Know what exactly you can do with Python - TechVidvan

IV. Conclusion

Object-Oriented Programming (OOP) is a way of organizing code that feels natural and intuitive because it mirrors how we think about the world. It uses **classes** as blueprints to create **objects**, which are like real-life examples with their own data and actions. This makes it easier to break big problems into smaller, manageable parts.

The four main ideas of OOP—**encapsulation**, **inheritance**, **abstraction**, and **polymorphism**—help keep code clean and flexible. **Encapsulation** hides the messy details, so you only deal with what's necessary. **Inheritance** lets you reuse existing features without starting from scratch. **Abstraction** focuses on what matters, ignoring the rest, and **polymorphism** allows different objects to behave in similar ways, making your code more dynamic.

Python is especially great for OOP because it's simple and flexible. Whether you're building a website, analyzing data, creating a game, or automating tasks, OOP helps you write code that's easier to understand, update, and reuse. It's a powerful way to build solutions for real-world problems while keeping things organized and efficient.

Reference

Website

-Beautiful, T. I. (2023, March 5). 5 Characteristics of Object-Oriented Programming (OOP) and how to use it. *Medium*. https://medium.com/@techisbeautiful/5-properties-of-object-oriented-programming-oop-and-how-to-use-it-f38609d7b85d

-Design Gurus, "Beginner's Guide to Object-Oriented Programming (OOP)," *Design Gurus: One-Stop Portal for Tech Interviews.* https://www.designgurus.io/blog/object-oriented-programming-oop

-A. Goodarzi, "Object Oriented Programming - part 2: Field, Producer, Class and Object," Nov. 23, 2023. https://www.linkedin.com/pulse/object-oriented-programming-part-2-field-producer-class-goodarzi-y0rif#:~:text=Fields%2C%20also%20known%20as%20attributes,%2C%20strings%2C%20or%20custom%20objects.

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

-"What is object-oriented programming (OOP)?" https://www.tutorialspoint.com/What-is-object-oriented-programming-OOP