Data Structure and Algorithm

Laboratory Activity No. 1

Object-oriented Programming

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Monoy, Justin Rhey A. | Engr. Maria Rizette H. Sayo |

July, 26, 2025

# Objectives

This laboratory activity aims to implement the principles and techniques in object-oriented programming specifically through:

* Identifying object-orientation design goals
* Identifying the relevance of design pattern to software development

# Methods

* Software Development
  + The design steps in object-oriented programming
  + Coding style and implementation using Python
  + Testing and Debugging
  + Reinforcement of below exercises
  1. Suppose you are on the design team for a new e-book reader. What are the primary classes and methods that the Python software for your reader will need? You should include an inheritance diagram for this code, but you do not need to write any actual code. Your software architecture should at least include ways for customers to buy new books, view their list of purchased books, and read their purchased books.
  2. Write a Python class, Polygons that has three instance variables of type str, int, and float, that respectively represent the name of the polygon, its number of sides, and its area. Your class must include a constructor method that initializes each variable to an appropriate value, and your class should include methods for setting the value of each type and retrieving the value of each type.

# A diagram of a computer program AI-generated content may be incorrect.Results

# 

Figure 1. Inheritance diagram

This diagram shows how a simple book reading app works, where users can buy and read books either as e-books or audiobooks. At the center of the system is the book, which contains details like the title, author, and content. There are two types of books: E-Book and Audio Book. E-books include additional information such as file size and format (like PDF), while audiobooks contain details like the duration, the narrator, and the audio file itself.

The book reader is the tool that helps users read books. It keeps track of which book is currently open and the page the user is on, and it allows them to move forward or backward through the pages. The user class stores important details about the person, including their name, email, and the books they have purchased. Users can buy books, view their collection, and read them through the app.

The bookstore is where all available books for purchase are stored. It displays the list of books and allows users to buy them. Once a user purchases a book, it is added to their collection. All parts of the app are connected: the user interacts with the bookstore to buy books and uses the book reader to read them. This design keeps the app simple, organized, and easy to use.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure 2. Source code

This Python code creates a simple class called Polygons, which is used to describe shapes that have many sides, like an octagon or a hexagon. Each shape has three main details: its name (like "Octagon"), how many sides it has (like 8), and its area (like 77.25). The class allows us to set or update these details and also get or show them when needed.

In the example, two shapes are created: an octagon and a hexagon. For each shape, the program prints its name, number of sides, and area. The result shows that the octagon has 8 sides and an area of 77.25, while the hexagon has 6 sides and an area of 93.53. This code is a good way to organize and handle information about different polygons using object-oriented programming.

# Conclusion

I learned that the book reading app is made in a simple way where users can buy and read books, either as e-books or audiobooks. The app has parts like the bookstore, the book reader, and the user, and they all work together to make it easy to use.

I also learned how to use Python to make a simple class for polygons like octagons and hexagons. The code helps store and show the name, number of sides, and area of each shape. Both the diagram and the code helped me understand how to organize and manage information better.

**References**

[1] *W3Schools.com*. (n.d.). <https://www.w3schools.com/python/python_inheritance.asp>

[2] *Python Inheritance (With examples)*. (n.d.). <https://www.programiz.com/python-programming/inheritance>  
[3] Codecademy. (n.d.). *Python Inheritance explained: Types and use cases*. Codecademy. https://www.codecademy.com/article/what-is-python-inheritance