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:
Gabijan, Rhovic M.
Saturday 4:30 – 8:30 PM / BsCpE 1-A

Submitted to
Engr. Maria Rizette H. Sayo
Instructor

Date Performed:
18-01-2024

Date Submitted
18-01-2024

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

Python is an object orient programming and classes are very essential in this language since it provides bundling data and functionality. Class instances also have methods for modifying its state. Unlike other programming languages, Python’s mechanism adds classes with a minimum of new syntax and semantics. Additionally, class plays a trick with namespaces. It needs to know how the scope and namespace work, to understand what is happening. A namespace is mapping from names to objects. Most namespaces are in the dictionary of python, one of the examples of namespace is the set of built-in-names such as abs (). In addition, namespaces are created in different moments and lifetimes. The namespace containing built-in-names are permanent in python when the interpreter startups. This is an example demonstrating how to reference the different scopes and namespaces:

```
def scope_test():
    def do_local():
        spam = "local spam"

    def do_nonlocal():
        nonlocal spam
        spam = "nonlocal spam"

    def do_global():
        global spam
        spam = "global spam"
    spam = "test spam"

    do_local()

    print ("After local assignment:", spam)

    do_nonlocal()

    print ("After nonlocal assignment:", spam)

    do_global()

    print ("After global assignment:", spam)


scope_test()

print ("In global scope:", spam)
```

2. Objects

Python objects are instances of classes, it is essential for creating blueprints and objects. Since python is an object-oriented programming is emphasized the use of objects and codes. Here’s an example of creating class and objects in python:

```
class Person:

def __init__ (self, name, age):

self.name = name

self.age = age


p1 = Person ("John", 36)

print(p1.name)

print(p1.age)
```

3. Fields

The field function in python is a powerful tool to customize the behavior of the class attributes. It is used to control the class attributes present in the dataclass, such as providing defaults values, excluding/including attribute _repr_, or any class attribute _init_ method.

dataclasses.field(*, default, default_factory, repr, hash, init, compare, metadata)

4. Methods

In using python, the method is defined within a class and associated with an instance of that class. Also, it is used to define the behavior of an object. Furthermore, when you define function in class, it’s purely a function, in contrast when you call the function via an instance of class, the function becomes method.

Example:

```
class Student:
def study(self):
return "Student is studying..."
student1 = Student()
print(student1.study()) # Output: Student is studying
```

5. Properties

The property of python is a built-in function that allows you to create managed attributes in your classes. Meaning you will be able to control access to an attribute by defining getter, setter, and delete. Because of this approach it helps in maintaining stable public API (Application programming interface) while allowing internal changes to implementation.

Example:

```
class Circle:
def __init__(self, radius):
self._radius = radius

def _get_radius(self):
return self._radius

def _set_radius(self, value):
self._radius = value

def _del_radius(self):
del self._radius
```

```
        radius = property(_get_radius, _set_radius, _del_radius, "The radius property.")
```

III. Results

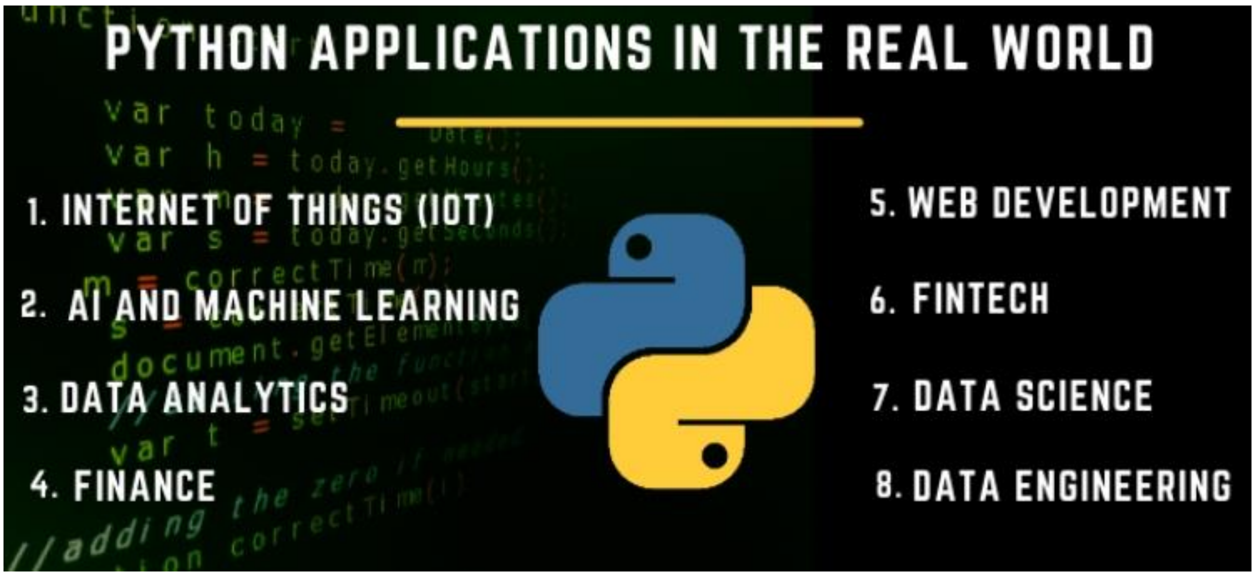
PYTHON APPLICATIONS IN THE REAL WORLD
1. INTERNET OF THINGS (IOT)
2. AI AND MACHINE LEARNING
3. DATA ANALYTICS
4. FINANCE
5. WEB DEVELOPMENT
6. FINTECH
7. DATA SCIENCE
8. DATA ENGINEERING

Figure 1.0: Python Applications & Real World Uses

What is Python Used For? Python Applications & Real World Uses - Invedus

Python is a versatile programming language as it has a wide variety of applications in various domains. Its simplicity, readability, and extensive libraries make it more popular for developers. Web development is widely used in python. These frameworks help developers to build robust and scalable web applications. Next is Artificial intelligence. The libraries of python facilitate data manipulation, mathematical computations, and development of complex algorithms making it more useful in developing artificial intelligence. Another is game development, since python is object-oriented programming, it is used in game development creating both 2d and 3d games. There are still a lot of applications of python in our world.

IV. Conclusion

In conclusion, Python has a lot of uses in the technology industry because of its simplicity, accessibility, and various applications in the real world. Python is also an object-oriented programming language that has classes, objects, fields, methods, and properties. A class is a blueprint that defines methods and fields, it doesn’t hold data directly however, it provides the structure to create objects with desired properties and behaviors. Object is an instance of class and represent real world entities with attributes and behaviors based on the class. Further, Fields also known as attributes are variable belongs to object. They are defined within the class and used to store data typically initialized _int_() method. In addition to that, Methods are function within the class and they describe behaviors and actions that an object perform often refers to the current instance of an object. Lastly, Properties is a especial kind of attribute that allows you to getter and setter, it is commonly used to add logic or constraints when setting or getting an attribute’s value.

Reference

Website

[1] "9. classes," *Python Documentation*. https://docs.python.org/3/tutorial/classes.html

[2] "W3Schools.com." https://www.w3schools.com/python/python_classes.asp

[3] GeeksforGeeks, "Python classes and objects," *GeeksforGeeks*, Dec. 13, 2024. https://www.geeksforgeeks.org/python-classes-and-objects/

[4] SmritiSatyan, "Python Dataclass (field Function) - Part 3," *Studytonight*, Jun. 13, 2023. https://www.studytonight.com/post/python-dataclass-field-function-part-3

[5] Python Tutorial, "Python Methods," *Python Tutorial - Master Python Programming for Beginners From Scratch*, Nov. 28, 2021. https://www.pythontutorial.net/python-oop/python-methods/

[6] "Python @property Decorator (With Examples)." https://www.programiz.com/python-programming/property

[7] P. Geeks, "Applications of Python - Python Geeks," *Python Geeks*, Jun. 26, 2021. Available: https://pythongeeks.org/applications-of-python/

[8] "Top 10 Python Applications in Real World," *GeeksforGeeks*, Oct. 22, 2021. https://www.geeksforgeeks.org/python-applications-in-real-world/

[9] Ritika, "Frequently asked questions (FAQs) related to python," *Blogs*, Dec. 30, 2021. https://invedus.com/blog/what-is-python-used-for/ (accessed Jan. 18, 2025).