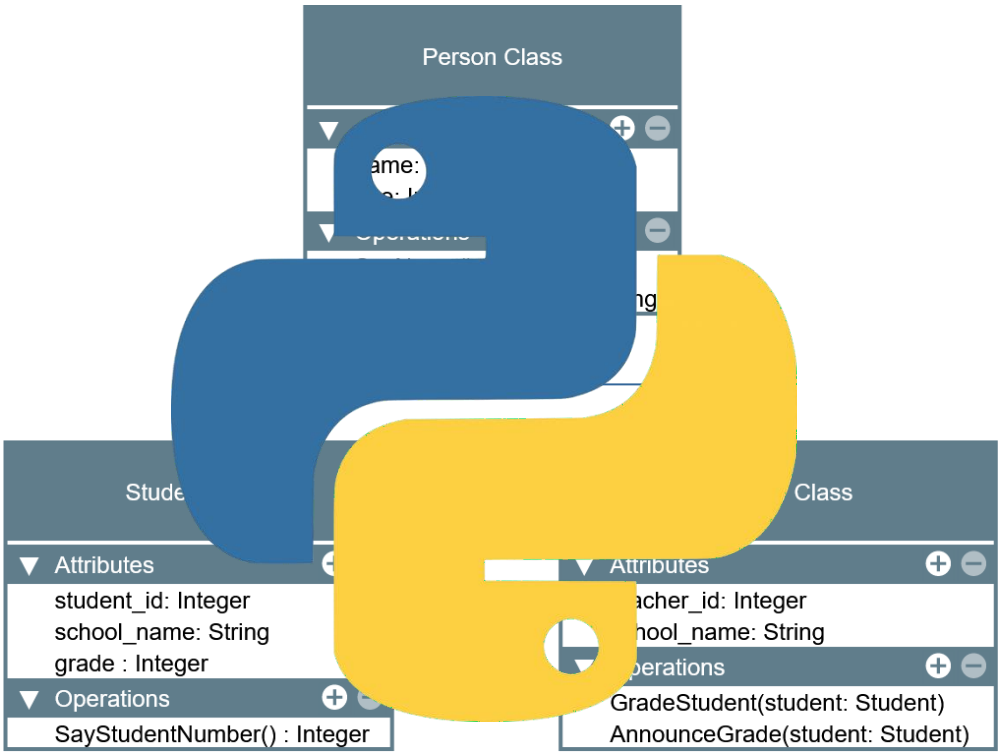
UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

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



LABORATORY MANUAL

Object-Oriented Programming
(CPE 103)

UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

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

Laboratory Activity No. 2.1	
Literals, Operators, and Variables	
Course Code: CPE103	**Program:** BSCPE
Course Title: Object-Oriented Programming	**Date Performed:** 1 / 25 / 2025
Section: 1A	**Date Submitted:** 1 / 25 / 2025
Name: Asugas, Kenneth R.	**Instructor:** Engr. Maria Rizette Sayo

1. Objective(s):

This activity aims to familiarize students in the various data types of Python, assign values to variables, and perform operations in a Python program.

2. Intended Learning Outcomes (ILOs):

The students should be able to:
2.1 Assign different values to variables in Python
2.2 Perform different operations available with variables in Python

3. Discussion:

The **Python** programming language is an interpreted language meaning the lines are evaluated line -by-line at runtime because there is no compile time at Python. This means that Python can dynamically allocate memory to variables as needed depending on the line of code that it interprets that is why Python is also referred to as a Dynamically typed language.

Like other programming languages such as C/C++ and Java, Python can also assign values to specific blocks of memory through variables as well as perform operations such as but not limited to Addition, Subtraction, Multiplication, Division, and Modulo(remainder). This activity will focus on assigning values and performing operations in Python.

Recall that a **variable** is a name that points to a specific location in memory where the data is stored. A variable can be allocated memory based on the data type it is assigned with which in Python can be: **Integer**, **Float**, **Complex Number**, **Boolean**, and **String**. In Python, **lists**, **tuples**, and **dictionaries** are also referred to as data types specifically sequences. More information can be found here (https://docs.python.org/3.8/reference/datamodel.html?highlight=data%20type#objects -values-and-types). These will be discussed further in lab activities.

Variables in Python are assigned in the following manner:

```
variable_name = value
```

Literals refers to the raw data given in a variable or constant. Literals can be some of the following: Numeric, Complex, String, Boolean, Special. Other literals are list, tuple, dict, set, and Unicode literals.

4. Materials and Equipment:

Desktop Computer with Anaconda Python /Python Colab
Windows Operating System

5. Procedure:
Perform the activity using the Jupyter Notebook This activity can be done either locally on Anaconda's Jupyter Notebook or online through Google Collaboratory which offers a free Jupyter Notebook environment for Google Users. IPython Notebook files (.ipynb) that are saved in the Google Drive can be opened on Google Collaboratory. Additional guides are available on the IPython Notebook template file that is provided wit h this activity. If the template is not present, these are the valuable links for reference:

UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

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

https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Notebook%20Basics.html
https://colab.research.google.com/notebooks/welcome.ipynb
https://colab.research.google.com/notebooks/markdown_guide.ipynb

Assigning variables of different data types in Python

1. In an empty cell, declare a variable **value** and assign it the value of 5 then display its value using print().
2. Create a new cell and type the command: type(value) then run the cell. The output should be like the image below.

```
In [3]: type(value)

Out[3]: int
```

3. In a new cell, use the same variable **value** and assign it the value of 5.0 then print the value.
4. Repeat step 2.
   Note: You may choose to decide how you execute the code in the cells for the next tasks in the procedure.
5. Repeat these steps for the following values:
   a. 2+3j
   b. 'Hello World'
   c. "Hello World"
   d. True
   e. False
   f. [1,2,3,4,5]
   g. (1,2,3,4,5)
   h. { 'name': 'Your_name' }
   i. None
6. Re-assign the **value** variable to be equal to 5.
7. Declare a new variable named **value2** to be equal to -6.

For the program please refer to this link: CPE-103-OOP-1-A/Procedure.ipynb at main · Kenneth-Asugas/CPE-103-OOP-1-A

UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

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

Performing Operations with Python

1. Using **value** and **value2**. Type the command: print(**value**+**value2**)
2. Repeat step 1 for the following values of **value** and **value2**:
   Hint: You may try using this assignment ***value, value2 = 5, -6*** in the Notebook for the following steps:
   a. value, value2 = 5.0, 6
   b. value, value2 = -5, 6.1
   c. value, value2 = “Hello”, ‘world’
      Note: Modify the code so that hello and world would be separated.
   d. value, value2 = [1,2,3], [4,5,6]
   e. value, value2 = (1,2,3), (4,5,6)
   f. value, value2 = {"name":"Royce"}, {"age":2}
      Note: Observe the outputs carefully and try repeating them using subtraction.

For the program please refer to this link: CPE-103-OOP-1-A/Procedure.ipynb at main · Kenneth-Asugas/CPE-103-OOP-1-A

3. Using value, value2 = 30, 4. Type the commands:
   a. print(value*value2)
   b. print(value2**2)
   c. print(value2**3)
   d. print(value*value2+value2**2+1)
   e. print(value/value2)
   f. print(value%value2)

Receiving Input Data using Python

Data can be received through keyboard input in Python by using the input() function. The input function has the following syntax:

input("Message Name")

The "Message Name" is an optional String parameter that can be customized to prompt the user for a message instead of having to print a message prompt separately. The default return value of the input() function is a String containing the value received from the keyboard. This value can be assigned to a variable shown in the example below:

name = input("Enter your name: ")

For the program please refer to this link: CPE-103-OOP-1-A/Procedure.ipynb at main · Kenneth-Asugas/CPE-103-OOP-1-A

UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

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

Assigning Input Data to a Variable

Finding a person's BMI (metric)

1. Declare a new variable named **name** and assign it the value **input("Enter your name")**
2. Create another variable named **weight** and assign it the value **input("Enter your weight(kg): ")**
3. Create another variable named **height** and assign it the value **input("Enter your meters(m): ")**
4. Declare another variable called **bmi** and assign it the formula $bmi = \frac{weight}{height^2}$
5. Address the errors displayed step#4. You can accomplish this by converting the String input to another data type. An example would be:

   weight = input("Enter your weight(kg)")
   weight = float(weight)

   Or simply **weight** = float(input("Enter your weight(kg): "))

   There are many functions available that can convert one data type to another. Some of which are the following:
   int(), float(), str()

   Other functions which maybe used in the later lab activities are: complex(real, imaginary), list(), tuple(), set(), dict(), ord(), bin(), hex(), oct().

6. Print the persons's name, weight, height, and bmi
   Name: John Ray
   Weight: 60
   Height: 1.6764
   BMI = 21.3499

   Guide: 5.5 feet ~ 1.6764 m

For the program please refer to this link: [CPE-103-OOP-1-A/Procedure.ipynb at main · Kenneth-Asugas/CPE-103-OOP-1-A]

Hint: You can combine two values by converting the output value to String and Concatenating (Addition) the operator on two strings.

print("Value: "+str(12))

You may explore many other methods to format values onto the print() function in Python. Another example is the following:

print("Value: ", 12)

For the program please refer to this link: CPE-103-OOP-1-A/Procedure.ipynb at main · Kenneth-Asugas/CPE-103-OOP-1-A

UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

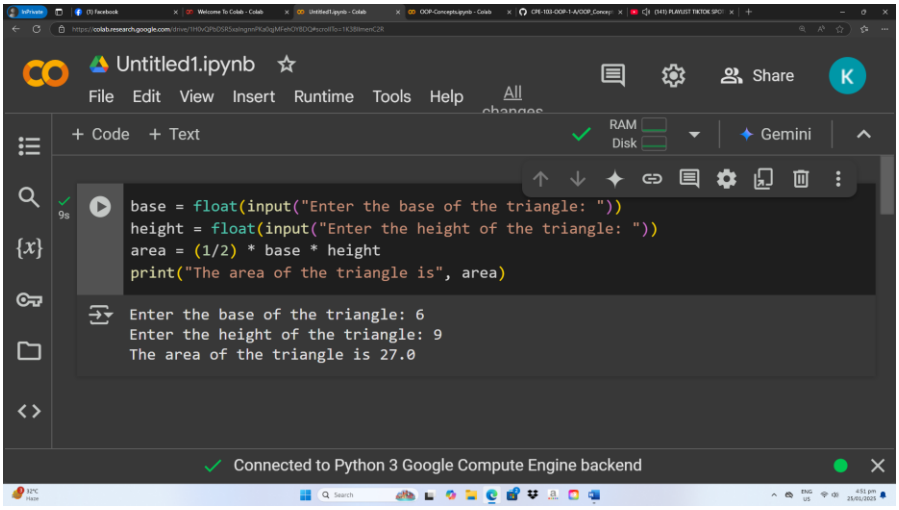
COLLEGE OF ENGINEERING
Computer Engineering
2nd Semester, School Year 2024-2025

6. Supplementary Activity:

Tasks

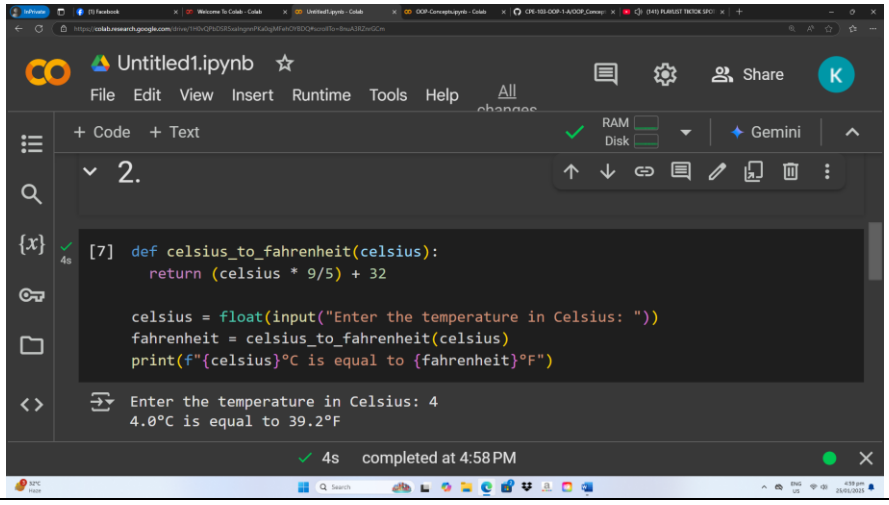
1. **Write the Python equivalent code of the following C code:**

```
int main(){
    float base = 0, height = 0, area = 0;
    printf("Enter the base of the triangle: ");
    scanf("%f", &base);
    printf("Enter the height of the triangle: ");
    scanf("%f", &height);
    area = (1/2)*base*height;
    print("The area of the triangle is %f", area);
}
```



2. **Write a program that would convert Celsius to Fahrenheit given the formula: F = (C × 9/5) + 32**
Example of conversion:

0°C = 32 °F
-20°C = -4 °F



UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

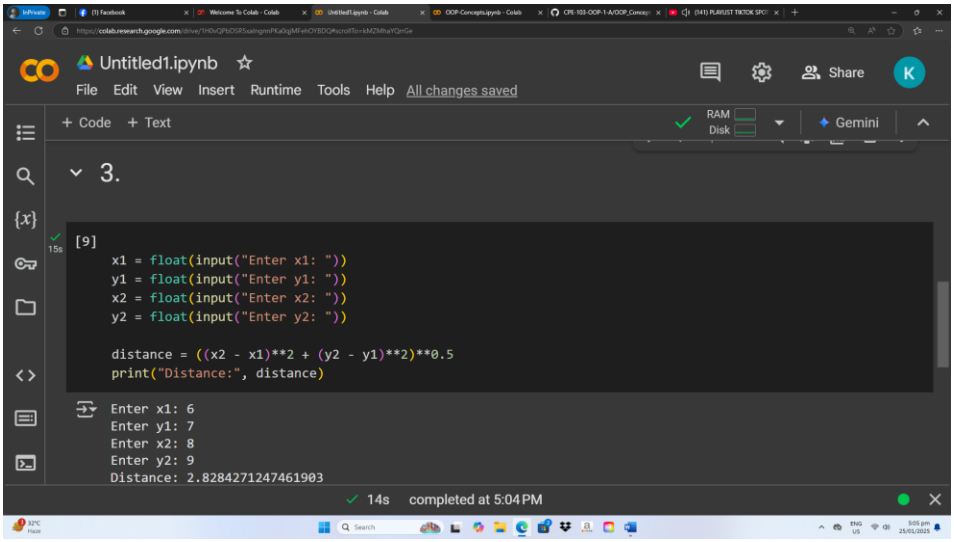
COLLEGE OF ENGINEERING
Computer Engineering
2nd Semester, School Year 2024-2025

3. **Write a program that can determine the distance between two points given the coordinates using the formula:**

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Hint/Rule: No library or package is needed to implement this equation.

Example: x2, y2 = -3, 3 and x1, y1 = 2, 2 d = 5.099019514



For the program of number 1,2,and 3 please refer to this link: [CPE-103-OOP-1-A/Tasks.ipynb at main · Kenneth-Asugas/CPE-103-OOP-1-A]

UNIVERSITY OF CALOOCAN CITY
Caloocan, 1400 Metro Manila, Philippines

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

Questions:

1. **Give one major difference in syntax that Python has with other languages such as C?**

   One major difference in syntax between Python and C is that Python doesn't use semicolons to end a line, while C requires them. Python also uses indentation to define blocks of code instead of curly braces.

2. **How does variable assignment differ in Python compared with other languages such as C?**

   In Python, variables don't need a type to be declared beforehand; you can just assign a value to them, and Python will figure out the type. In C, you have to declare the type of a variable before using it.

3. **Try assigning variable names that start with numbers, and special characters. Is the assigning of variables that start with numbers accepted by Python? For Special Characters? Is there an exception for variables special characters?**

   In Python, you can't name variables starting with numbers or special characters (like @ or $). However, you can start a variable name with an underscore (_), which is often used for temporary or private variables.

4. **Do the assignment operators (+, -, *, /, %, **) work for all data types? Why or Why not?**

   Not all assignment operators like +, -, *, /, %, and ** work with every data type. + and * can work with strings or lists but % and / only work with numbers. It depends on the data type and what operation it supports.

5. **How does the * operator differ from the ** operator?**

   The * operator is for multiplication or repeating things like strings or lists. Meanwhile the ** operator is for exponentiation meaning raising a number to a power.

7. Conclusion:

In this laboratory activity, we explored the basics of Python programming, including literals, operators, and variables. We learned how Python dynamically assigns data types to variables without requiring explicit type declarations, which sets it apart from languages like C. Through hands-on exercises, we successfully assigned different values to variables, performed various operations, and analyzed their outputs. Additionally, we implemented a simple program to calculate BMI, demonstrating how input data can be processed and utilized in Python.

Overall, this activity helped solidify our understanding of Python's syntax, dynamic typing, and its ability to handle different data types and operations efficiently. The practical tasks emphasized Python's simplicity and versatility, making it an excellent language for beginners and advanced programming tasks.

8. Assessment Rubric: