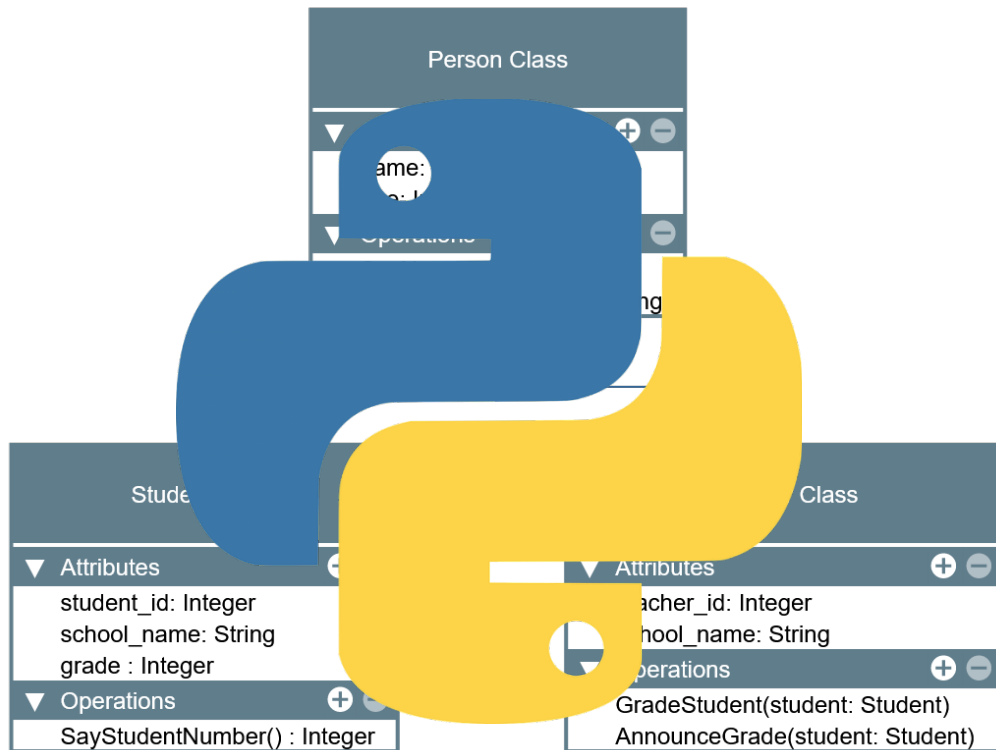


TECHNOLOGICAL INSTITUTE OF THE PHILIPPINES
COMPUTER ENGINEERING DEPARTMENT
Quezon City



LABORATORY MANUAL

Object-Oriented Programming
(CPE 009)

Prepared by:
Engr. Royce Chua

Activity No. 2

Literals, Operators, and Variables

Course Code: CPE009

Program: BSCPE

Course Title: Object-Oriented Programming

Date Performed: 11/14/2019

Section: CPE12FB2

Date Submitted: 11/14/2019

Name: Julongbayan, Dan Angelo A.

Instructor: Sir Razon

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
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 with this activity. If the template is not present, these are the valuable links for reference:

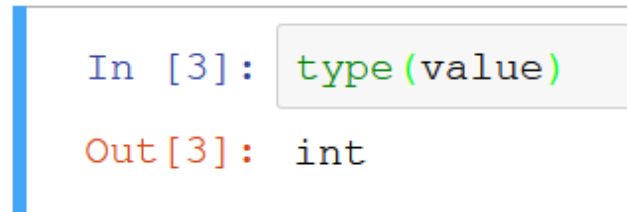
<https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Notebook%20Basics.html>

<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.

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.

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: ")`

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

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)
```

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;
    printf("The area of the triangle is %f", area);
}
```

2. Write a program that would convert Celsius to Fahrenheit given the formula: $F = (C \times 9/5) + 32$

Example of conversion:

0°C = 32 °F

-20°C = -4 °F

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: $x_2, y_2 = -3, 3$ and $x_1, y_1 = 2, 2$ $d = 5.099019514$

Questions:

1. Give one major difference in syntax that Python has with other languages such as C?
Python is very simplistic and only requires indentations instead of having to add terminators such as a semicolon(;), and nesting with curly brackets({})
2. How does variable assignment differ in Python compared with other languages such as C?
Python does not require you to declare the variable before its use. It just reads the data inside the variable and automatically identifies its data type. You can also assign multiple values to multiple variables in a single line.
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?
Variables that start with a number and special characters return syntax errors. The underscore(_) special character however, is supported and works fine.

4. **Do the assignment operators (+, -, *, /, %, **) work for all data types? Why or Why not?**
It does not, generally data types that support character inputs can only be added, as for lists, tuples and sets, they support only some operators. On the other hand, the 'dict' data type does not support any operator.
5. **How does the * operator differ from the ** operator?**
The * operator indicates multiplication, and the ** operator indicates that the left operand is being raised to the power of the right operand.

7. Conclusion:

After this experiment, I learned that in Python, the declaration of data types can be omitted. Instead of spending 3 or so lines declaring variables, their types, and their literal values, you can accomplish all that in python for only one line.

Python supports multiple value assignments for multiple variables, given that you separate each by commas (.). It also automatically identifies what data type a variable is the moment it reads the value for that variable. Say you assign a 1 to a variable x, python recognizes that as an integer variable. Likewise, assigning an 'a' to a variable x will give it a character or string data type.

Aside from recognizing data types that we're already familiar with from C language, Python also introduces the complex, boolean and dictionary data types. Complex can hold values like 4+3j while boolean data types hold "True" or "False" values. Dictionaries which hold keys to values.

Aside from data types of variables and literals, Python also applies the same properties of operators with the special addition of the ** operator, which eases the addition of exponents. Although I still used the pow(x,y) function in the activity knowing this operator exists and works? Definitely helps.

8. Assessment Rubric:

JULONGBAYAN_Laboratory Activity 2

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().

```
In [1]: value = 5  
print(value)  
5
```

2. Create a new cell and type the command: type(value) then run the cell.

```
In [2]: type(value)  
Out[2]: int
```

3. In a new cell, use the same variable value and assign it the value of 5.0 then print the value.

```
In [3]: value = 5.0  
print(value)  
5.0
```

4. Repeat step 2.

```
In [4]: type(value)  
Out[4]: float
```

5. Repeat these steps for the following values:

a. $2+3j$

```
In [5]: value = 2+3j
        print(value)
(2+3j)
```

```
In [6]: type(value)
Out[6]: complex
```

b. 'Hello World'

```
In [7]: value = 'Hello World'
        print(value)
Hello World
```

```
In [8]: type(value)
Out[8]: str
```

c. "Hello World"

```
In [9]: value = "Hello World"
        print(value)
Hello World
```

```
In [10]: type(value)
Out[10]: str
```

d. True

```
In [11]: value = True
         print(value)
True
```

```
In [12]: type(value)
Out[12]: bool
```

e. False

```
In [13]: value = False
         print(value)
False
```

```
In [14]: type(value)
Out[14]: bool
```

f. [1,2,3,4,5]

```
In [15]: value = [1,2,3,4,5]
         print(value)
[1, 2, 3, 4, 5]
```

```
In [16]: type(value)
Out[16]: list
```

g. (1,2,3,4,5)

```
In [17]: value = (1,2,3,4,5)
         print(value)
(1, 2, 3, 4, 5)
```

```
In [18]: type(value)
Out[18]: tuple
```

h. {'name': 'Your_name'}

```
In [19]: value = {'name': 'Julongbayan_Dan_Angelo'}
         print(value)
{'name': 'Julongbayan_Dan_Angelo'}
```

```
In [20]: type(value)
Out[20]: dict
```

i. None

```
In [21]: value = None  
         print(value)
```

None

```
In [22]: type(value)
```

```
Out[22]: NoneType
```

6. Re-assign the value variable to be equal to 5.

```
In [23]: value = 5
```

7. Declare a new variable named value2 to be equal to -6.

```
In [24]: value2 = -6
```

Performing Operations with Python

1. Using value and value2. Type the command: print(value+value2)

```
In [25]: print(value+value2)
```

-1

2. Repeat step 1 for the following values of value and value2:

a. value, value2 = 5.0, 6

```
In [26]: value, value2 = 5.0, 6  
         print(value+value2)  
         print(value-value2)
```

11.0

-1.0

b. value, value2 = -5, 6.1

```
In [27]: value, value2 = -5, 6.1  
         print(value+value2)  
         print(value-value2)
```

1.0999999999999996

-11.1

c. value, value2 = "Hello", ' world'

```
In [28]: value, value2 = "Hello", ' world'  
         print(value+value2)  
         print(value-value2)
```

Hello world

```
-----  
TypeError                                Traceback (most recent call last)  
<ipython-input-28-a65f9bc4025b> in <module>  
      1 value, value2 = "Hello", ' world'  
      2 print(value+value2)  
----> 3 print(value-value2)  
  
TypeError: unsupported operand type(s) for -: 'str' and 'str'
```


d. value, value2 = [1,2,3], [4,5,6]

```
In [29]: value, value2 = [1,2,3], [4,5,6]
print(value+value2)
print(value-value2)
```

```
[1, 2, 3, 4, 5, 6]
```

```
-----
TypeError                                 Traceback (most recent call last)
<ipython-input-29-ab18783f8eab> in <module>
      1 value, value2 = [1,2,3], [4,5,6]
      2 print(value+value2)
----> 3 print(value-value2)

TypeError: unsupported operand type(s) for -: 'list' and 'list'
```

e. value, value2 = (1,2,3), (4,5,6)

```
In [30]: value, value2 = (1,2,3), (4,5,6)
print(value+value2)
print(value-value2)
```

```
(1, 2, 3, 4, 5, 6)
```

```
-----
TypeError                                 Traceback (most recent call last)
<ipython-input-30-aa2816bb4f79> in <module>
      1 value, value2 = (1,2,3), (4,5,6)
      2 print(value+value2)
----> 3 print(value-value2)

TypeError: unsupported operand type(s) for -: 'tuple' and 'tuple'
```

f. value, value2 = {"name": "Royce"}, {"age": 2}

```
In [31]: value, value2 = {"name": "Royce"}, {"age": 2}
print(value+value2)
print(value-value2)
```

```
-----
TypeError                                 Traceback (most recent call last)
<ipython-input-31-80047b7d9541> in <module>
      1 value, value2 = {"name": "Royce"}, {"age": 2}
----> 2 print(value+value2)
      3 print(value-value2)

TypeError: unsupported operand type(s) for +: 'dict' and 'dict'
```

Note: Observe the outputs carefully and try repeating them using subtraction.

Observation:

Operand type (+) is unsupported for data type 'dict'

Operand type (-) is unsupported for data types: 'str' 'list' 'tuple' and 'dict'

3. Using value, value2 = 30, 4. Type the commands:

```
In [32]: value, value2 = 30,4
```

```
print(value*value2)
```

```
In [33]: print(value*value2)
```

```
120
```

```
print(value2**2)
```

```
In [34]: print(value2**2)
```

```
16
```

```
print(value2**3)
```

```
In [35]: print(value2**3)
```

```
64
```

```
print(value*value2+value22+1)
```

```
In [36]: print(value*value2+value2*2+1)
```

```
129
```

e. `print(value/value2)`

```
In [37]: print(value/value2)
```

```
7.5
```

f. `print(value%value2)`

```
In [38]: print(value%value2)
```

```
2
```

Assigning Input Data to a Variable

For finding a person's BMI;Body-Mass Index(Metric)

1. Declare a new variable named name and assign it the value input("Enter your name")

```
In [41]: name = str(input("Enter your name: "))
```

```
Enter your name: Dan Angelo Julongbayan
```

2. Create another variable named weight and assign it the value input("Enter your weight(kg): ")

```
In [42]: weight = float(input("Enter your weight(kg): "))
```

```
Enter your weight(kg): 65.5
```

3. Create another variable named height and assign it the value input("Enter your meters(m): ")

```
In [43]: height = float(input("Enter your height(m): "))
```

```
Enter your height(m): 1.82
```

4. Declare another variable called bmi and assign it the formula

$$bmi = \frac{weight}{height^2}$$

```
In [44]: bmi = weight/(pow(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()`.

```
In [45]: print('Name:', name)
         print('Weight:', weight)
         print('Height:', height)
         print('BMI:', bmi)

Name: Dan Angelo Julongbayan
Weight: 65.5
Height: 1.82
BMI: 19.774181862093947
```

JULONGBAYAN_Laboratory Activity 2

SUPPLEMENTARY ACTIVITY

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;
printf("The area of the triangle is %f", area);
}
```

```
In [1]: base = float(input('Enter the base of the triangle: '))
height = float(input('Enter the height of the triangle: '))
area = (1/2)*base*height

print('The area of the triangle is', area)
```

```
Enter the base of the triangle: 5
Enter the height of the triangle: 8
The area of the triangle is 20.0
```

2. Write a program that would convert Celsius to Fahrenheit given the formula: $F = (C \times 9/5) + 32$

Example Conversion: 0C = 32F ; -20C = -4F

```
In [2]: celcius = float(input('Enter temperature in C: '))
fahrenheit = (celcius*(9/5))+32

print(fahrenheit)
```

```
Enter temperature in C: -20
-4.0
```

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

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

Example: $x_2, y_2 = -3, 3$ and $x_1, y_1 = 2, 2$ $d = 5.099019514$

```
In [4]: x1 = float(input('x1: '))
x2 = float(input('x2: '))
y1 = float(input('y1: '))
y2 = float(input('y2: '))
import math
d = math.sqrt(pow(x2-x1, 2)+pow(y2-y1, 2))
print(d)
```

```
x1: 2
x2: -3
y1: 2
y2: 3
5.0990195135927845
```

In []: