LBYCPA1

**Programming Logic and Design Laboratory**



**Laboratory Module 1**

Introduction to Python Programming

By

John Carlo Theo S. Dela Cruz || LBYCPA1-EQ1

# INTRODUCTION

The 1st module of LBYCPA1 is an introduction and utilization of ***Python, Jupyter Notebook*** and how can we utilize these programs to take this with us in our journey to learning Python Programming. We were first introduced with Jupyter Notebook, in this web-based platform we could perform Python live codes and equations as well as taking notes during discussions. This platform will be the playground for Computer Engineering / Science students, in which we conduct the experiment as well as documenting every analyzation regarding our modules.

The primary job of this Lab Report is for us to understand, coding and programming works. how Python is on the top programming languages, The reason why it is one of the top programming languages because it emphasizes readability, it makes coding easy in the language of Python, and it is more object-oriented. Python was introduced to us as an interactive shell and python script that we can use as a calculator, and it gives us the idea how arithmetic function works, and basics of Input and Output. The ***primary goal*** of starting something new is by being knowledgeable and practice the basics of coding and programming, this module is about getting to know about the language, what are the operations used in python? What is the function of Input and Outputs? How can we solve the following problems with the use of arithmetic equations only? Every problem has an answer and solution once the sets are finished.

1. **Objectives**
2. To understand basic knowledge about how programming works.
3. To implementation of programming knowledge to real life application.
4. To develop a program using Python.
5. To create a strong foundation in programming.
6. To be able to learn the different syntax to use with Python.
7. To learn how to use the Jupyter Notebook.
8. To learn about arithmetic operation in Python.
9. To understand the concept of Algorithms, Pseudocodes and Flowchart.
10. To solve various problems using Python.
11. To diagram our own flowcharts about our code.
12. **Materials and Tools**
13. Instructor's lecture notes
14. Modules
15. Jupyter Notebook
16. Flowchart Software – Diagrams.net
17. Google

# PROCEDURES (*Individual*) / EXPERIMENTAL PLAN

Before proceeding on performing or writing the code. It is important to read and understand the problem before initializing the code, it is like attacking without any weapon or preparation. A plan is a must every time we start coding, and it is the process of making Algorithms, Pseudocodes or Flowcharts depending on how we can interpret our plan better.

1. **Familiarization Exercise 1**: ‘aboutme.py’ – In this exercise, we will be familiarizing on how to execute a script in Anaconda Prompt.
   1. Copy the text/code given in the Jupyter Notebook (Preliminary Report 1).
   2. Open a blank notepad.
      1. Paste the code
      2. Modify the required text
      3. Save as ‘aboutme.py’, make sure to end the file with the ‘.py’ to format it into a python file
   3. Open CMD.exe prompt on the Anaconda Navigator
   4. Find the location of your ‘aboutme.py’ file.
      1. Copy the directory
   5. Use the command cd to change the directory and paste the directory of where the ‘aboutme.py’ is located.



* 1. Type “python” and the file name “aboutme.py”

Text

Description automatically generated

1. **Familiarization Exercise 2: Input and Output**
   1. Creating a simple input and output program that requires the user to input their name, birthyear and the program prints out the name that the user inputted and their age.

Diagram

Description automatically generated

1. **Familiarization Exercise 3:** 
   1. Creating a program that calculates the various Mathematical Equations; Cube – Cylinder – Cone – Sphere.

Diagram

Description automatically generated

1. **A picture containing diagram

   Description automatically generatedFamiliarization Exercise 4:**

**Diagram

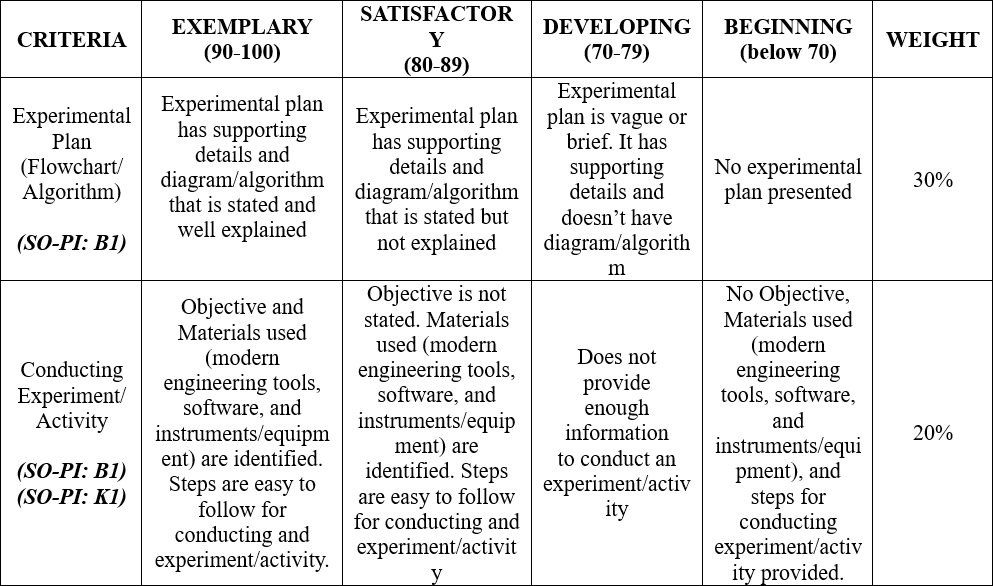
Description automatically generated**

1. **Familiarization Exercise 5:**
2. **Familiarization Exercise 6:**

Diagram

Description automatically generated with low confidence

**The Introduction together with individual Procedures and plan comprises the Experimental Plan and Conducting Experiment/ Activity criteria in the Final Laboratory Report Rubric**:



# RESULTS AND DISCUSSION/COMPUTATIONS (*Include the program output screenshots, and discussions per problem solution*)

**1. Familiarization Exercise 1 Result: ‘aboutme.py’**

* The program’s goal is to print out my personal details and execute that script in the Anaconda Prompt. First thing to do is to copy the Python code and paste it into a blank notepad page. Once you have pasted the code, modify the required text and I inputted my personal information inside the quotation marks. After inputting that information, I saved that file into ‘aboutme.py’. We need to indicate the ‘.py’ to convert it into a python file. Find the location of the 'aboutme.py' and I copied the directory in the address bar of my file explorer. Launch the CMD.exe prompt on the Anaconda Navigator and a black box will appear and all I need to do is to type 'cd' to change the directory and paste the directory that we copied a while ago. After pasting the directory, type "python 'aboutme.py'" to enable the python version and execute the file in the Command Prompt.
* **A screenshot of a computer

  Description automatically generatedCode:**
* **Result:**

**Text

Description automatically generated**

**Familiarization Exercise 2 Result: Input and Output**

* This is an example of a basic input-output program, where in the user is required to input their respective names and birthyear. The main variables are ‘name’, ‘birthyear’, and ‘age’, and these variables has their own unique functions, both ‘name’ and ‘birthyear’ requires the user to input the required fields while the ‘age’ function is dependent to the variable for ‘birthyear’. The find the age is just simply subtracting the year today which is 2022, to the inputted birthyear. The last section of the code is to display the user’s name and age. In printing a unique set of characters, we should always enclose them with quotation marks, because it indicates that they are not variables.
* Graphical user interface, text, application, chat or text message

  Description automatically generated Code and Result:

**Familiarization Exercise 3: Cube Equation**

* This problem is also an input-output set, but it is in a form of a mathematical problem. Formula should be precise for this program to work. In this program, the user is required input a value of the side of the cube which is written as variable ‘a’. The variable a is set as an input variable for the calculations, after defining the ‘volume\_cube’ below the line is where we set the variable V in which it will calculate the value of the cube.
  + The command return V will return the said variable as the computed volume is assigned.
  + Command a = float(input(“a = “)), it will require the user to input a number, and that input is set as a float. A float represents a real number and is written with decimal values.
  + Command V = volume\_cube(a) is used to confirm the declared variable from the start of the program, and it will be moved to V to make sure that the value of V is computed.
* Last few lines of the code are about printing out the value of the cube. The reason why the output has 2 significant digits to the right only is because of the string formatting. {0.2f} tells Python to put as much whole numbers but only 2 significant decimal values.
* Code and Result:

Graphical user interface, text, application

Description automatically generated

**4. Familiarization Exercise 4: Cylinder Equation**

* This problem is the same as exercise 3 but in this exercise, the solution requires the value of pi. Formula should be precise for this program to work, and we need to import the built-in function in Python called ‘math’. After importing the function ‘math’, I also imported the function ‘pi’, so I can easily declare the value of ‘pi’ by just inputting the word ‘pi’. Same with the previous Exercise, in which the user is required to input the radius and the height of the cylinder, which is written as variable r, and h.
  + The command return V will return the said variable as the computed volume is assigned.
  + Command a = float(input(“r = “)), it will require the user to input a number, and that input is set as a float. A float represents a real number and is written with decimal values.
  + Command V = volume\_cylinder(r, h) is used to confirm the declared variable from the start of the program, and it will be moved to V to make sure that the value of V is computed.
* Last few lines of the code are about printing out the value of the cylinder is the same as exercise 3. The reason why the output has 2 significant digits to the right only is because of the string formatting. {0.2f} tells Python to put as much whole numbers but only 2 significant decimal values.
* Graphical user interface, text, application, email

  Description automatically generatedCode and Result:

**Familiarization Exercise 5: Cone Equation**

* The problem in this exercise is almost the same in exercise 5 in terms of the formula used is similar to a cone. The only difference in the formula is that the equation should be multiplied by one-third. We need to import the built-in function ‘math’ because we will be needing the value of pi again. Same functions defining the volume cone, execute the code with the formula and returning the value of V after inputting the formula. In terms of output, we will be using the string formatting so there will be only 2 significant figures to the right.
* Code and Results:

Graphical user interface, text, application, email

Description automatically generated

**Familiarization Exercise 6: Sphere Equation**

* Asdasd
* Code and Result:

Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated

# CONCLUSION:

The Conclusion returns to the larger purpose of the lab, which is presented as the learning context in the Introduction: **to learn something about the scientific or computational concept that provides the reason for doing the lab**. This is where you demonstrate that you have indeed learned something by stating what it is you have learned. This is important because it helps you to understand the value of the lab and convinces the reader that the lab has been a success. It's important, then, to be specific, providing details of what you have learned about the theory or principle or procedure at the center of the lab. The following guides questions should be answered to create a good conclusion:

**Did you achieve your objectives for this module? Describe your achievement in this module one-by-one, objective by objective. Ideally, each objective has its own paragraph.**

**What have you learned? Both technically (focused on programming), and in general (soft skills, attitude-related, etc.)**

**What are the common pitfalls, mistakes, and confusion that you have encountered? How did you overcome them?**

**What are your recommendations for those who will try the activity for the first time and what can you suggest to improve this module?**

**The rest of the rubric criteria are as follows:**

**Text

Description automatically generated with medium confidence**

# REFERENCES (*Enumerate references in APA format*)

1. Parker James R. (2017). *Python : An Introduction to Programming*, Dulles, Virginia : Mercury Learning and Information.
2. Barry, Paul, (2016). [*Head First Python*](http://stackabu.se/head-first-python)*,* Second Edition*,* O'Reilly Media, Inc.
3. Lambert, Kenneth A., (2015). *Python Programming for Teens*, Boston, MA : Cengage Learning PTR.
4. Beazley, David and Jones, Brian K., (2013). [*Python Cookbook*. Third edition](http://stackabu.se/python-cookbook), O'Reilly Media, Inc.
5. Fangohr, H. (2015). *Python for Computational Science and Engineering*. Faculty of Engineering and the Environment University of Southampton.
6. Gowrishankar, S., & Veena, A. (2018). *Introduction to Python Programming*. Chapman and Hall/CRC.
7. Matthes, E. (2015). *Python crash course: a hands-on, project-based introduction to programming*. No Starch Press.
8. Miller, B. N., Ranum, D. L., & Anderson, J. (2019). *Python programming in context*. Jones & Bartlett Learning.
9. Sweigart, A. (2015). *Automate the boring stuff with Python: practical programming for total beginners*. No Starch Press.
10. LabWrite Project 2000, *“The Parts of a Laboratory Report”*. URL: https://projects.ncsu.edu/labwrite/res/res-studntintro-labparts.html (accessed 02/20/2020)

This is a list of the references that were cited in the lab report, including the lab manual, any module handouts accompanying the lab, the textbook, and sources from the scientific literature. The format for references differs in different fields and even within the same field. It's important that you check with your teacher or lab manual to find out what is expected of you.

# APPENDIX (*Attach all the source codes here per problem category*)

1. Familization Exercise 1:

*"""  
Author: Dino Ligutan  
Course: LBYCPA1  
Section: EQ1  
Date Created: March 21, 2022  
--------------------------------------------------  
This program simply displays "Hello World!"  
"""*print("Hello World!")

2. and so on …