LBYCPA1

**Programming Logic and Design Laboratory**



**Laboratory Module 2**

Basic Syntax, Variables, Data Types, Numbers, Casting

By

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

# INTRODUCTION

This module is mostly concerned with programming grammar, variable naming, and declaration. Additionally, the session discusses and teaches about data types, numbers, and casting. Together with module 1, this module will serve as a basis for programming. As with Module 1, the materials and tools for this module will be Jupyter notebooks and Pycharm.edu.

As mentioned in the form, the objective of a lab report is to provide an opportunity for learning rather than to contribute to field knowledge. With that stated, the act of coding, such as flowcharts and pseudocodes. Python enables ease of use when it comes to writing; users, such as myself, can easily grasp the code and identify syntax or logical issues. With the help of Jupyter notebook, you may review your code while you're still working; the platform will also identify any issues in your code once it's run. It is a simple necessity to study the fundamentals; regardless of your field of endeavor, returning to the fundamentals will influence us, students in some way. The module covers variable declaration and naming, as well as statements, values, and expressions. Additionally, the module would teach the many Python data types, including strings, numbers, tuples, and many others. In Python, type conversion, or casting, would also be included in the second module.

1. Objectives
2. To understand the different data types in Python
3. To understand and be familiarize with various arithmetic operators.
4. To solve advanced mathematical problems using Python
5. To properly run the various arithmetic operators
6. To manipulate a certain function.
7. To execute and run Karel the Bot
8. Materials and Tools
9. Module 2
10. Karel the Bot
11. Jupyter Notebook
12. Pycharm.edu
13. Diagrams.net
14. W3schools
15. StackOverflow

# PROCEDURES (*Individual*) / EXPERIMENTAL PLAN

Each exercise requires understanding the problem, identify the expected input and outputs, and formulate the process of a program. Proper planning is a must in coding, I used flowcharts, algorithms, and pseudocodes before I can execute our codes:

1. Exercise 1: GPA Calculator
   1. This program is a GPA Raw Calculator, and it requires the user to enter their grades of the subject in COEDISC, LBYCPA1 and CALENG1; then the program formulates and will show the GPA. In formulating the GPA, I simply followed the guide on the notes once again and declared a variable gpa.

Text

Description automatically generatedChart, diagram

Description automatically generated

1. Exercise 2: Triangle Area calculator
   1. This program is basically a calculator, and it requires the users to input 3 sides to find the area of the triangle. I followed the guide from my notebook, and I imported the built-in function math and imported a specific function called square root. In this problem, it was specified that we will use the function of a square root, so I searched a formula and I found Heron’s Formula that fits with the criteria of this activity.

Diagram

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. Exercise 3: Decimal points calculator
   1. The program requires the users to input a number with 2 decimal points and the program gets the value of the 2 decimal points and transform it into a whole number. The thought process in this activity is to subtract the input with 2 decimal points, minus the whole number of that specific input (3.14 – 3 = 0.14). Once we get the result, I multiplied the result with 100 to transform it to a whole number. Text

      Description automatically generated

Diagram

Description automatically generated

1. Exercise 4: 3-digit Reverse
   1. This program requires an input of a 3-digit number and the program reverses the position of the number. By using modulo and division integer, my initial plan here is to get every place value and manipulate them (Ones Value into Hundredths), by getting the remainder and by using the division integer 10 or 100 (to simplify and it can end with 0) then multiply it with a specific place value. Since we have the value of each place value, we just need to add them all to combine them all together.

Table

Description automatically generatedChart, diagram

Description automatically generated

1. Exercise 5: Karel the Bot
   1. I downloaded the Karel the Bot python package and simply followed the guide on how the bot can be manipulated with specific syntax to create and draw my middle initial.

A picture containing scatter chart

Description automatically generated

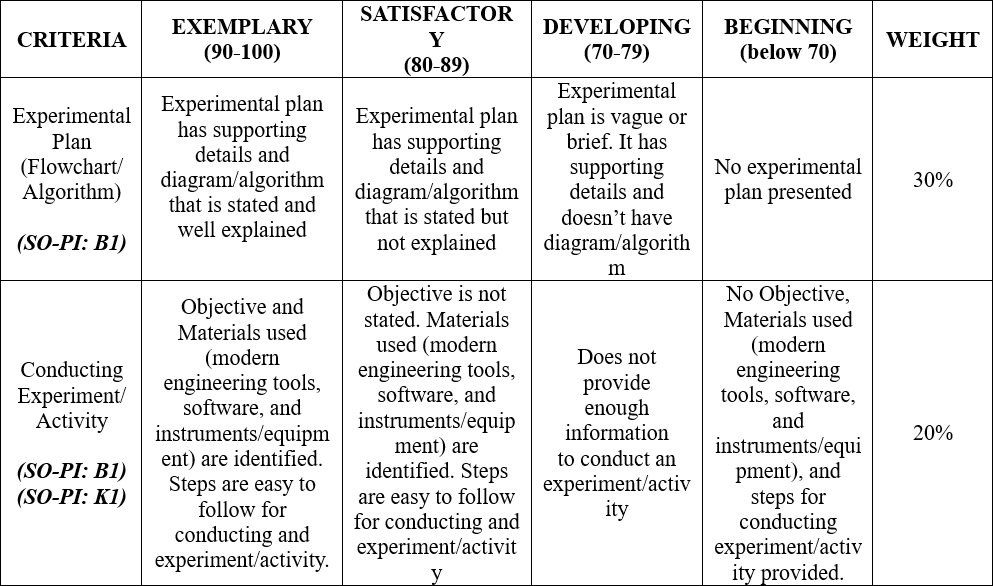
Text

Description automatically generated with medium confidenceText

Description automatically generatedText

Description automatically generatedWriting Middle initial (S.) using beepers: PSUEDOCODE:

**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:** **GPA Calculator**

**Explanation:**

In this Activity, we were assigned to develop a program that will calculate the GPA on 3 specific courses. The input of the program is expressed as a float since, we will be inputting the grades each course, and it is in a form of a decimal number. I categorized each subject in the variable (gpa1, gpa2, gpa3) separately. Before I write the code of the formula for computing the GPA, I checked first our course flowchart to know and identify the number of units of COEDISC, LBYCPA1, and CALENG1. The code is executed, and its formula would be the SUBJECT multiply by how many UNITS, apply this to the three subjects, get the sum and divide it with the total number of UNITS to calculate the raw GPA. Line 1 defined the function cal\_GPA3 and the value would return on “return gpa” and it also indicates the end of the function. The final section of the code prints the output of the program which is the expected output or the calculated final GPA grade. Text

Description automatically generated

**2. Familiarization Exercise 2 Result: Triangle Area Calculator**

**Explanation:**

In this Problem, we are tasked to develop a Triangle Area Calculator; in which the user must enter the triangle’s sides, and in this exercise, we were tasked to use the function of a square root to find the area of the triangle. I searched a formula on google and its name is Heron’s Formula in which, we get the sum of the sides of the triangle divided by 2 to get the Perimeter of the triangle, and to find the Area we will use the function of the square root to find the Area: Graphical user interface, text, application

Description automatically generated

Importing the built-in function of python called math so we can use the operator of the square root; write the formula inside the function and as the value returns from the function, all we need to do is to print out the expected output or the final answer of the area and declaring the final answer as a float so the result would be exact.

Graphical user interface, text, application

Description automatically generated

**3. Familiarization Exercise 3 Result: Decimal Point Calculator**

**Explanation:**

This is a tricky problem since we will only use arithmetic operators and type conversion functions. The program is expected for the user to input a number with 2 decimal places and the output would be the 2 decimal places that the user has inputted. The thought process of this problem is first, I subtracted the user’s input minus the user’s input in Integer form (Whole Number), the result would just be the 2 decimal places that the user has inputted and to make it a whole number; get the value of the result and multiply to transform the decimal into a whole number. Finally print the expected output and set the function into an integer so the output would only be a whole number.

Graphical user interface, text, application, email

Description automatically generated

**4. Familiarization Exercise 4 Result: 3-Digit Reverser**

**Explanation:**

The user is required to input any 3-digit number and the program will reverse the placements of each digit. The program should perform only with arithmetic functions to reverse the digits and the result must be a string type. I divided my code into 3 parts so I can identify and take out the value of each place value.

* digit\_1 = num % 10 \* 100
  + The input would be divided by 100 and get the remainder of the **Ones’ value** then multiply it by 100 to transform into a **Hundredths’ place value**
* digit\_2 = num % 100 // 10 \* 10
  + The input would be divided by 100 and get the remainder of the **Tens’ value**; use division integer and divide the result by 10, and multiply it by 10 to transform into a **Tens’ place value**
  + You get the remainder of the tens value then // 10 to get the tens value and multiply by 10
* digit\_3 = num % 1000 // 100 \* 1
  + The input would be divided by 100 and get the remainder of the **Hundredths’ value**; use division integer and divide the result by 10, and multiply it by 10 to transform into a **Ones’ place value**
* result = str (digit\_1 + digit\_2 + digit\_3)
  + In this line of code, it is basically getting the sum of all digits and making sure that the data would be saved.

Finally, as the function returns the value; The program would print out the expected output / reversed digits.

Text

Description automatically generated

**5. Familiarization Exercise 5 Result: Karel the Bot**

**Explanation:**

In this activity, I first had to install the Karel Python package through the Anaconda and used Pycharm.edu to create and input the code for Karel the Robot. We used the Karel functions to accomplish the activity to spell out my middle initial of S in the Karel 2D environment. The functions are the following:

* move()
  + This function instructs the Karel Bot to move one tile on what direction is the bot facing.
* put\_beeper()
  + This function instructs the Karel Bot to move one tile on what direction is the bot facing.
* turn\_left()
  + This function instructs the Karel Bot to rotate their direction to 90 to the left / counterclockwise
  + A picture containing scatter chart

    Description automatically generated

**The Results and Discussions constitutes the data criteria in the Lab Report Evaluation Rubric**:

Text

Description automatically generated

# CONCLUSION:

The second module focuses on the fundamentals of Python. The best programming language is foreign to my understanding, and I’ve enjoyed learning it and experimenting on it. I wish to have a better understanding and develop an appreciation for programming. Like every module’s objective, we should assure ourselves that we understand and execute the programs smoothly.

I gained a lot of understanding about Python, and I am getting used to it. The module aided me tremendously throughout the coding process, from accessing my notes to pointing up errors in my code, and even creating a cell in the Jupyter Notebook to experiment what those functions can do and how can I manipulate it. I’ve learned to be patient because there were a lot of errors as I process and execute my codes. It’s a trial-and-error basis for me, and I hope in the future I can easily find and analyze my code right away. Overcoming the error would need me to be more vigilant and precise with my coding and spelling. I feel that it is something that I can work on and improve upon. Apart from that, I loved this problem set and can honestly claim that I learnt a lot. This module helped me think outside the box and trying to think simple solutions to solve these problems, just like the problem in which we will get the value of the decimal value, it trains us to be critical thinkers and to analyze the problems better. There are a lot to improve upon. Apart from that, I honestly claim that I learnt a lot.

My advice to individuals attempting this module for the first time is that you must truly listen to the lectures and comprehend the module. Third-party resources such as w3 schools and stackOverflow can assist you if you are having difficulty understanding the modules and the syntax. Another piece of advice is that when it comes to coding, you must be conversant with variable naming and most especially the indentions, Python is very sensitive with indentions and syntax.

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

**Text

Description automatically generated with medium confidence**

# REFERENCES (*Enumerate references in APA format*)

* w3school (n.d.). Retrieved from. https://www.w3schools.com/python/python\_intro.asp
* stackOverflow (n.d.). Retrieved from. <https://stackoverflow.com/>
* Math is Fun. (n.d.). Heron’s Formula. Retrieved from. <https://www.mathsisfun.com/geometry/herons-formula.html>

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

1. Exercise 1: GPA Calculator

def calc\_GPA3(gpa1, gpa2, gpa3):

gpa = ((gpa1 \* 1) + (gpa2 \* 2) + (gpa3 \* 3)) / 6

return gpa

gpa1 = float(input("What is your COEDISC GPA? "))

gpa2 = float(input("What is your LBYCPA1 GPA? "))

gpa3 = float(input("What is your CALENG1 GPA? "))

gpa = calc\_GPA3(gpa1, gpa2, gpa3)

print("Your GPA is: ", gpa)

1. Exercise 2: Triangle Area calculator

FORMULA\_NAME = "triangle\_area"

import math

from math import sqrt

def triangle\_area(s1, s2, s3,):

# --- Heron's Formula

P = (s1 + s2 + s3)/2

A = sqrt (P \* (P - s1) \* (P - s2) \* (P - s3))

return P

return A

s1 = float(input("Side 1: "))

s2 = float(input("Side 2: "))

s3 = float(input("Side 3: "))

A = triangle\_area(s1, s2, s3,)

print("Triangle Area is: ", A)

1. Exercise 3: Decimal points calculator

def tenths\_hundredths(num\_float):

whole\_num = num\_float - int(num\_float)

result = whole\_num \* 100

return result

num\_float = float(input("Enter a number with a decimal: "))

result = int(tenths\_hundredths(num\_float))

print("Decimal Value:", result)

1. Exercise 4: 3-digit reverser

def reverse3(num):

digit\_1 = ((num % 10) \* 100)

digit\_2 = (((num % 100) // 10) \* 10)

digit\_3 = (((num % 1000) // 100) \* 1)

result = str( digit\_1 + digit\_2 + digit\_3 )

return result

num = int(input("Enter a Number: "))

result = (reverse3(num))

print(num, " == ", result)

1. Exercise 5: Karel

Diagram

Description automatically generated

from stanfordkarel import \*  
  
def main():  
 *"""Karel code goes here!"""* # move() - move  
 # turn\_left() - rotate 90 degrees  
 # put\_beeper()  
 move()  
 move()  
 put\_beeper()  
 move()  
 put\_beeper()  
 move()  
 put\_beeper()  
 move()  
 put\_beeper()  
 move()  
 turn\_left()  
 move()  
 put\_beeper()  
 turn\_left()  
 move()  
 move()  
 move()  
 move()  
 move()  
 put\_beeper()  
 turn\_left()  
 turn\_left()  
 turn\_left()  
 move()  
 turn\_left()  
 turn\_left()  
 turn\_left()  
 move()  
 move()  
 move()  
 move()  
 put\_beeper()  
 turn\_left()  
 move()  
 turn\_left()  
 move()  
 put\_beeper()  
 move()  
 turn\_left()  
 turn\_left()  
 turn\_left()  
 move()  
 put\_beeper()  
 move()  
 turn\_left()  
 move()  
 put\_beeper()  
 move()  
 turn\_left()  
 turn\_left()  
 turn\_left()  
 move()  
 put\_beeper()  
 move()  
 turn\_left()  
 turn\_left()  
 turn\_left()  
 move()  
 put\_beeper()  
 move()  
 put\_beeper()  
 move()  
 put\_beeper()  
 move()  
 put\_beeper()  
 move()  
 turn\_left()  
 turn\_left()  
 turn\_left()  
 move()  
 put\_beeper()  
 turn\_left()  
 move()  
 turn\_left()  
 move()  
 turn\_left()  
 turn\_left()  
 turn\_left()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 run\_karel\_program()