ENGR 13300
Python 1: Introduction to Python

Recall the guidelines for team activities:

1. You should work as a team; **all** team members will be held responsible for all material. You may work together and contribute to one program and submit similar codes as long as the contributors to the development of the solution are documented.
2. Each student is responsible for submitting their own assignment.

Task 1 (of 7) [Team]

Learning Objectives: Create and execute simple scripts comprised of basic Python concepts; Describe the characteristic and purpose of modules in Python; Import modules in Python (i.e. math module); Employ the Spyder IDE to write, edit, and save Python code; Output data from a script to the screen in Python; Apply course code standard in development of Python scripts.

Background:

Modules are a simple way to structure a program. Generally speaking, there are standard modules in the standard library and there can be other Python files, or directories containing Python files, in the current directory (each of which constitutes a module). Standard modules in Python must be imported before use. For example,

```
import math
```

imports the math standard module. After importing this module, you can use the functions inside the math module. Here an example showing the syntax:

```
math.sqrt(10)
```

Write a Python script that computes the area of a paralellogram that has two sides of 8 cm and 6 cm and an angle of 90 degrees in between. You can use the following formula to calculate the area of a paralellogram:

```
area of paralellogram = a * b * sin(C)
```

The program should then output the result to the screen.
Note: The sine function in Python returns the sine of x assuming x is in radians.

Example:

Please note that the actual computation results below may not be correct. This example is intended to elucidate matters of format only. The output of your program should have the same format.

```
The area of a paralellogram is 48[cm^2] for a given length of sides
8[cm] and 6[cm], with the angle of 90 [degrees] in between.
```

Task 1 Files:

1) `Py1_Task1_`*`teamnumber`*`.py`

Task 2 (of 7) [Team]

Learning Objectives: Recognize and explain the differences between various data types in Python; Convert between data types (e.g. strings to integers, or floats to integers) in Python.

Background:
In engineering context, you need to deal with various data types to solve your problem. Some of the important categories of data types in Python include Integers, Floating Points, and Strings.

The following is a section of code that was entered into Python to calculate the volume of a cylinder:

```
>>>import math
>>>radius = 5
>>>height = 5
>>>volume = math.pi * height * radius ** 2
>>>print(f'The volume of a cylinder is {volume}[cm^3] for a given')
>>>print(f'diameter of {2*radius}[cm] and height of {height}[cm].')
```

Include the answers to the following questions in a PDF file: `Py1_Team_`*`teamnumber`*`.pdf`

1. What changes do you need to make to the code to get the following output?

```
The volume of a cylinder is 392 [cm^3] for a given
diameter of 25 [cm] and height of 5 [cm].
```

2. What changes do you need to make to the code to get the following output?

```
The volume of a cylinder is 393 [cm^3] for a given
diameter of 25 [cm] and height of 5 [cm].
```

Task 2 Files:

1) `Py1_Team_`*`teamnumber`*`.pdf`

Task 3 (of 7) [Team]

Learning Objectives: Perform arithmetic operations in Python (i.e. addition, subtraction, multiplication, division, and exponentiation), while keeping in mind order of operations; Employ the Spyder IDE to write, edit, and save Python code; Output Python data from script to screen.

Background:
Python 3 is a powerful programming language for performing computations and database management. As an engineering tool, it offers standard mathematical operations as part of its basic environment. Therefore, you can design standard computational models, process data, and generate reports. The language must work within the limits of the machine it is running on. As a result, there may be differences between what you compute by hand and what Python computes.

Part A:

1) In your previously created PDF document, create a table with three columns on a new page: problem number, hand calculation, and Python calculation.
2) Assuming that equations 1-7 shown below are computed sequentially in the order shown, calculate by hand the result of each expression.
3) Perform these calculations in the Python terminal.
4) Fill in the table.

Hand Calculation represents the results of the arithmetic as would typically be displayed on a calculator.
Python Calculation represents the results computed and displayed from Python3

Hint: Remember the order of operations. You may need to import the math module to do some of the calculations in Python (second line). Also, for hand calculations, interpret the double division symbol as a regular division, but type it exactly as shown in Python. Finally, if a calculation gives an error or a "weird" result, simply record that fact in the table.

1. A = 5
2. B = A^3
3. C = B – A * 7 / 5
4. D = 7 ^ (7 / 3)
5. E = 3 ^ (2 * 5)
6. F = 179 % 17
7. G = 17 // 4 + 11 / 6

Part B:
Answer the following questions and include the answers in your previously created PDF:

1. What differences did you notice between the hand calculations and the Python calculations?
2. What Python function can you use to output these variables to the screen?
3. What syntax differences exist between Python and your calculator? Be specific.

Task 3 Files:

1) `Py1_Team_`*`teamnumber`*`.pdf`

Task 4 (of 7) [Team]

Learning Objectives: Create valid identifiers, accounting for relevant Python rules (i.e. keywords) and code standard; Implement the use of expressions in Python to assign values to variables.

Background:
Variables (or indicators) are unique labels for values in memory. These variable names should be descriptive so it is easier to understand how they relate to your algorithm. The form of these must be consistent. Also, Python needs to define its own set of unique indicators that are called keywords. This activity has you debug a series of statements and correct them so that the program will run.

The following is a section of code that was entered into Python and contains numerous errors involving variables.

1) For each line of code that needs correction, you should record all of the corrections necessary to make the code valid.
2) Explain why the changes need to be made.

NOTE: Python has reserved variable words as well as restrictions on what characters may be used in a variable name.

Save your answers in your previously created PDF file.

```
1.  >>>a=7
2.  >>>B-3=1.33333333
3.  >>>ProducT= 12.59
4.  >>>
5.  >>>return = a + B_2 - ProducT
6.  >>>
7.  >>>names@133 = Product * A
8.  >>>
9.  >>>lg = a^2
10. >>>
11. >>>Sum = Name - Lg-a*2
12. >>>
13. >>>def = a^ProducT
14. >>>
15. >>>3var = B_7 - 7
```

Task 4 Files:
1) `Py1_Team_`*`teamnumber`*`.pdf`

Task 5 (of 7) [Team]

Learning Objectives: Use the Spyder IDE to write, edit, and save Python code; Output data from a script to the screen in Python; Apply course code standard in development of Python scripts; Modularize and comment code in Python for readability and reusability.

Background:
Modules are handy when they can be used by others. With proper information in the header on how to use them and appropriate comments and description text throughout the algorithm, other programmers will know how to use the new function appropriately. Therefore, comments are needed to quickly and easily support future users of a module. This also applies to essentially all other programming languages. Commenting can also make it easier to debug your own modules by reminding you why you did certain things in your code.

Complete the following activity:

Create a Python program that

- Takes the radius as input from the user (use the `input()` function; note that you will have to cast the returned string from the `input()` function to an `int` or `float`. Casting can be done by `int_number = int(int_as_string)`, e.g. `int_number = int("3.3")`).
- Calculates and displays the area of a cone
- Includes comments explaining:
    i. acceptable inputs (don't forget units).
    ii. the steps in your script.

Save your script as:

`Py1_Task5_`*`teamnumber`*`.py`

Task 5 Files:

1) `Py1_Task5_`*`teamnumber`*`.py`

Individual Tasks

Guidelines for Tasks 6–7:
These two tasks are individual assignments. You may seek help from classmates, the instructional team or others but the work you submit should be your own. If you collaborate with others and use information developed together or by someone else, ALWAYS document and reference that material.

Task 6 (of 7) [Individual]

Learning Objectives: Perform arithmetic operations in Python (i.e., addition, subtraction, multiplication, division, and exponentiation), while keeping in mind order of operations; Create valid identifiers, accounting for relevant Python rules (i.e. keywords) and code standard; Implement the use of expressions in Python to assign values to variables; Recognize and explain the differences between various data types in Python; Convert between data types (e.g. strings to integers, or floats to integers) in Python based on program needs; Identify which resources are available to you to aid in learning Python; Import modules in Python (i.e. math module); Employ the Spyder IDE to write, edit, and save Python code; Output data from a script to the screen in Python.

Task:
As you further explore programming, you will find that there are a number of concepts and features hidden within each language that lend greatly to the usability of the language in a niche field or when looking to solve a specific problem in a specific way.

Python contains a number of hidden elements within its built-in modules that apply to unique situations or tasks. In this activity, you will be asked to delve deeper into the world of Python modules. In order to learn about and apply these modules, you will need to do some research on your own. Being able to use your resources to help you find answers to difficult problems will be an important skill for you to master, not only for this class, but also going forward as an engineer.

Suggested modules to research: `random`, `fractions`

Suggested built-in functions to research: `round()`, `int()`, `float()`, `input()`

Hint: When you want to find out what a function does, type `help(function_name)`, e.g. `help(int)` or `help(round)` into the Python Console (or search the internet).

Suggested: Investigate the difference between the following and how it affects the calling of functions:

```
from (module) import (function)
import module
```

You are spending your summer consulting at a start-up company that creates on-line educational tools to assist elementary school students in practicing math. You are working on a tool that demonstrates to students the equivalence between doing calculations with decimal numbers and doing the same calculations with fractions. The tool should randomly generate three numbers between zero and 100, add the rounded numbers together, and then convert these rounded numbers to fractions and repeat the addition using the fractions. The random numbers should be printed with two decimal's precision to the

screen. The fractions should be printed to the screen with a denominator smaller or equal to 100. An example output is shown below.

Write a Python script (filename: `Py1_Task6_username.py`) that will follow the procedure outlined above. Utilizing `random.seed(int(input("Enter the seed: ")))` before generating the random numbers initially is required in your program for it to be graded properly. When generating a random number in Python, a pseudo-random number generator is used which generates random numbers based on a seed. If a seed is set, the sequence of random numbers it generates is determined and the sequence of random numbers will be the same again if you set the seed to the same value.

Example Output:

```
Enter the seed: 143
First Random Number : 90.6371
Second Random Number: 85.14
Third Random Number: 4.5817
Sum from decimals   : 90.64 + 85.14 + 4.58 = 180.36
Sum from fractions  : 2266/25 + 4257/50 + 229/50 = 4509/25
```

Before you start programming in Python, create a flow chart representing the algorithm and save it in a PDF file that will be named `Py1_Ind_username.pdf`.

Note that your program will be graded based upon the output and the process that you used to reach this output. Your program should not produce any errors when running or points will be deducted. Do not use the `input()` function except to set the seed for the pseudo-random number generator.

Task 6 Files:

1) `Py1_Task6_username.py`
2) `Py1_Ind_username.pdf`

Task 7 (of 7) [Individual]

Learning Objectives: Create and execute simple scripts comprised of basic Python concepts; Output data from a script to the screen in Python; Apply course code standard in development of Python scripts; Modularize and comment code in Python for readability and reusability.

Task:
You have just started your first co-op assignment at Mho-ROM Electronic Components, Inc., a large manufacturer of semiconductors and integrated circuits. Over the years, Mho-ROM engineers have written a number of computer programs in languages that are no longer officially supported by the company. With many of those engineers retired or soon to retire, the company is concerned that it will no longer have the in-house expertise it needs to maintain these programs, leaving the company vulnerable should problems arise. Your supervisor has given you a first assignment to update several of these programs to use a modern programming language such as Python.

To get started, you decide to tackle a widely used algorithm which helps to model inductance in a circuit. Since you have already completed ECE 201, you know that an inductor is an electrical component comprised of insulated wire wound into a coil around a magnetic core. You remember that inductors store energy in a magnetic field when electric current flows through the wire. You are aware that you will need to accommodate two types of inductance networks: series and parallel. Additionally, you know that they are computed differently. Looking back at your ECE 201 notes, you find the following equations for inductors in series and parallel:

$$L_{equiv} = L_1 + L_2 + \ldots + L_n \qquad \text{(Series Inductance)}$$

$$\frac{1}{L_{equiv}} = \frac{1}{L_1} + \frac{1}{L_2} + \ldots + \frac{1}{L_n} \qquad \text{(Parallel Inductance)}$$

For this assignment, you will need to write a Python script (filename: `Py1_Task7_username.py`) to calculate the equivalent inductance of both the series and parallel configurations of two inductors. The program should output the results for both series AND parallel calculations. The output should be formatted in a table modeled similar to the one seen below, with all values rounded to the nearest tenth. You have been given a test case by your supervisor to illustrate the computation which you can use to test if your program outputs correct values.

The following tables are test cases only and should not be included in your program.

Type of Inductance	First Inductance	Second Inductance	Equivalent Inductance
Parallel	10.0 µH	20.0 µH	6.6 µH
Series	10.0 µH	20.0 µH	30.0 µH

Next, you must demonstrate the correctness of your program by applying it to the test case where the first inductance is an input from the user and the second inductance is $\sqrt{3.14\pi}$ µH. (Hint: When finding the value of $\sqrt{3.14\pi}$, be as exact as possible.). Do not use the input() function except to get the first inductance from the user. To better display your logic, create a flowchart and save it in your previously created PDF file (`Py1_Ind_username.pdf`)

Example: Assuming the following values: $L_1 = 5$ µH and $L_2 = \sqrt{3.14}\pi$ µH
The outputs of your Python program should be formatted like the following (inputs in bold):

```
Input the first inductance value [µH]: 5
Type       First      Second     Equivalent Inductance
Series     5.0 µH      5.6 µH      10.6 µH
Parallel   5.0 µH      5.6 µH      2.6 µH
```

Hint: `fstrings` can be used to obtain formatted strings and the built-in function `len()` gives you the length of a string.

Task 7 Files:

1) Py1_Task7_*username*.py
2) Py1_Ind_*username*.pdf

Submit files:

1. Py1_Team_*teamnumber*.pdf
2. Py1_Task1_*teamnumber*.py
3. Py1_Task5_*teamnumber*.py
4. Py1_Task6_*username*.py
5. Py1_Task7_*username*.py
6. Py1_Ind_*username*.pdf