CSE-101, LAB 1: GETTING STARTED

Name: _____ Roll Number: _____

The purpose of this assignment is to get you started with Python immediately. You will have hands on experience with Python expressions.
In addition to this handout, you need to download a file from Backpack's resource page.

1. Getting Started
For today's lab you will work with one file. Rename the downloaded file to `lab1_01.py`
Create a new directory on Desktop and name it as lab01 and copy the file in that directory.

Contents of lab_01.py:

```
print('Hello World!')
#print('Welcome to CSE101!')
```

This handout has several empty boxes that where you have to write answer for a given question. When you are finished with the lab, you should show your written answers to your lab instructor. The instructor may ask you a few questions to make sure you understand the material.

There are two ways to run Python. One is to execute "scripts", or files that contain Python commands. The other is to use Python interactively, typing in just one command at a time.

The file `lab1_01.py` is a script. To run a script, you type **python**, followed by the name of the script, into the command shell. Type

```
python lab1_01.py
```

and hit Return. Does anything happen?
Your response:
"Hello World!" gets printed on the terminal (without the quotes). This means the python script 'lab_01.py' gets executed.

Now open up the file lab1_01.py with some editor.
Delete the # character and save the file. Once again, run the script by typing "python lab1_01.py". What you do see this time ?
Your response:
The following gets printed on the terminal:
Hello World!
Welcome to CSE101!

To run Python interactively, type the word python by itself and hit Return. You will see several lines of text followed by the symbol >>>. This is the Python prompt. The purpose of the >>> is to let you know that you are currently running Python, and that you are no longer working with files and folders.

At the Python prompt, type
>>> 2+5
and hit Return (do not type the >>>). What happens?
Your response: 7 is printed on the terminal and the prompt (>>>) appears again.

2. Python Expressions
The following pages have a list of expressions. For each expression, first compute the expression in your head, without using Python. Write down what you think the value is in the second column. If you have no idea, write "?".
Next, use Python to compute the same expression. Write down Python's result in the third column. If the two values are different, you should try to figure out why Python gave the answer that it did. Come up with a reasonable explanation and put it in the final column. In this assignment, you are not graded on being correct so make your best guess at what is happening. For this assignment you can take help of lab instructor.

NOTE: The solution is given for Python 2.x and Python 3.x both. Both the answers are accepted for this lab exercise.

2.1. int and float Expressions

Expression	Calculated Value		Reason for Calculated Value
	Python 2.7	**Python 3.5**	
2 * 4	8	8	Multiplication Operator
3 ** 2	9	9	Exponentiation Operator
5 + 7 * 5	40	40	Operator precedence of ***** is greater than **+**. 5 + 7 * 5 → 5 + 35 → 40
(5 + 2) * 6	42	42	Evaluate brackets first. (5 + 2) * 6 → 7 * 6 → 42
-5 - -4 - -4	3	3	Unary plus and minus has greater precedence than addition and subtraction. -5 - -4 - -4 → -5 +4 +4 → 3

2 ** 3 ** 0	2	2	Exponent operator has right-to-left associativity. 2 ** 3 ** 0 → 2 ** 1 → 2
(2 ** 3) ** 0	1	1	Parentheses have greatest precedence. (2 ** 3) ** 0 → 8 ** 0 → 1
6 / 3	2	2.0	Python2: / is floor division ie., (½ = 0) Python3: / is true division ie., real quotient (float) is returned (½ = 0.5)
7 / 4	1	1.75	Same as above
7.0 / 4	1.75	1.75	Both operands get up casted to float
7 / 4.0	1.75	1.75	Both operands get up casted to float
16.0 * 0.5	8.0	8.0	Multiplication operation on floats returns float
16.0 ** 0.5	4.0	4.0	Exponent operation on floats returns float
16 % 2	0	0	Modulus operator. Remainder of 16 / 2 is 0.
17 % 2	1	1	Modulus operator. Remainder of 17 / 2 is 1.
6.2 % 4	2.2	2.2	Modulus operator. In python, a % b = a - (b * floor (a/b))

2.2. Comparisons and bool Expressions.

Expressions	Calculated Value (**Python 2.x/3.x**)	Reason for Calculated Value
4 < 5	True	4 is less than 5 hence expr evaluated to True
4 < 5 and 5 < 4	False	4 < 5 and 5 < 4 → (4 < 5) and (5 < 4) → True and False → False
True	True	True evaluates to True
true	NameError	‘true’ is not a defined keyword in python hence its value cannot be printed.

Expressions	Calculated Value (**Python 2.x/3.x**)	Reason for Calculated Value
True and False	False	and evaluates to False if any of its operands is False
True and True	True	and evaluates to True if all of its operands are True
True or False	True	or evaluates to True if any of its operands is True
not False	True	not inverts False to True
not not True	True	not not True → not (not True) → not False → True
not (False and True)	True	not (False and True) → not False → True
True or (False and True)	True	True or (False and True) → True or False → True
(6/0 == 1) and False	ZeroDivisionError	Evaluation of 6/0 causes a ZeroDivisionError
False and (6/0 == 1)	False	The second argument of a boolean expression is not evaluated if the first argument suffices to determine the value of the expression. False and 'Anything' → False, so 6/0 is never evaluated and therefore doesn't cause a ZeroDivisionError

Why does the last expression in the table above "work" but the one above it doesn't ?

Your response:

The second argument of a boolean expression is not evaluated if the first argument suffices to determine the value of the expression.

False and 'Anything' → False, so 6/0 is never evaluated and therefore doesn't cause a ZeroDivisionError

Whereas in the second last expression 6/0 == 1 is the first argument, evaluation of which causes a ZeroDivisionError

2.3. Types and Casting

Expression	Calculated Value (**Python 2.x/3.x**)	Reason for Calculated Value
float(6)	6.0	6 is converted to its corresponding floating point value which is 6.0
int(6)	6	int cast to int doesn't do anything
int(7.3)	7	7.3 is truncated to give 7 ie., only the integral part of the float is retained
float(int(7.3))	7.0	float(int(7.3)) → float(7) → 7.0
int(-7.3)	-7	-7.3 is truncated to give -7 ie., only the integral part of the float is retained
int(-7.7)	-7	-7.7 is truncated to give -7 ie., only the integral part of the float is retained
float(9) / 4	2.25	float(9) / 4 → 9.0 / 4 → 9.0 / 4.0 → 2.25
float(9 / 4)	2.0	float(9 / 4) → float(2) → 2.0