Part 1:

Exercise 1.1:

Answer 1) To answer the question, I have to take all the possible events into account.

There are three possible events that can happen regarding the quotation marks.

1. Leave out the right quotation mark,
2. Leave out the left quotation mark,
3. Leave out both of the quotation mark.

For the first (1) possible event, we receive the following output:

>>> print(‘hello world!)  
 File “<stdin>”, line 1  
 print(‘hello world!)  
 ^

SyntaxError: EOL while scanning string literal

EOL stands for End of Line. The interpreter scans the code statements from left to right. As it scans the statement, the Python interpreter encounters the end of a line. All strings are supposed to be quoted (Downey, 2016). The first quotation mark on the right signifies the start of a string. So, the string needs to end. Without ending it without a quotation mark, the interpreter will throw an error in the standard output. This is what the error looks like.

For the second (2) possible event, we receive an invalid syntax.

>>> print(‘hello world!)  
 File “<stdin>”, line 1  
 print(hello world!’)  
 ^

It is because the interpreter is unable to parse a structure (Downey, 2016) of a statement it is not familiar with. There should be a quotation on the right of the string if there is a quotation at the end. So, it ultimately raises an error. The print statement will be

For the third (3) possible event, we receive an invalid syntax because the python interpreter is trying to parse an illegal syntax.

>>> print(hello world!)

File "<stdin>", line 1

print(hello world!)

^

SyntaxError: invalid syntax

The statement needs to have a proper structure for the python interpreter to interpret.

Answer 2)

If there is not negative sign before a number, the number is considered to be positive (‘+’) by default. If we use a ‘+’ operator, it will know that the operator is a positive number. By using a negative, we can change the positive to a negative number. As the interpreter scans from left to right, it computes as per the operators.

Input:  
>>>2++2  
Output:  
4  
Explanation: 2+(+2) = 4  
Input:  
 >>>2--2  
Output:   
4

Explanation: 2-(-2)

Input:  
>>>2+-2  
Output:  
0

Explanation: 2+(-2) = 0

Input:  
  
>>>2-+2  
Output:  
0  
Explanation: 2-(+2) = 0

Any numeric value preceded by ‘- ‘sign sets the numeric value to negative. And any ‘+’ sign changes a numeric value to positive (Hewgill et al., 2009)

Answer 3) Leading zeros are okay in Python 2.x. It means it is an octal number.

For example:

Input:  
>>>02  
Output:  
2

In python 3.x, leading zeros can be used followed by ‘b’, ‘x’, and ‘o’. Leading zeros are used to show a number of a different number system.

For example,

>>>0b10  
2

Answer 4)

Two values without any operator in between them is not a defined structure. It does not mean anything to the interpreter. It will raise a syntax error for that. For example:

>>> 2 4  
 File "<stdin>", line 1  
 2 4  
 ^  
SystaxError: invalid syntax

Part 2:

Experiment 1) I assigned a value to a variable and tried to print the output of the variable.

>>>i = 2  
>>>print(i)  
2

Here, if I did not assign any variable to i, the interpreter would give an error saying that the variable is not defined.  
  
>>>print(m)  
Traceback(most recent call last):  
 File “<stdin>”, line 1, in <module>  
NameError: name ‘m’ is not defined

Experiment 2)

>>>if a == 2:  
… print(‘2’)  
…   
…  
2

This is a conditional statement that prints the output based on a condition.

Experiment 3)

>>>a = 2  
>>>a == 2  
True  
  
The ‘=’ is an assigning operator. It assigns values to the operators. On the other hand, ‘==’ is a logic operator that checks a variable and returns Boolean values.

References:

1. Downey, A. (2016). *Think Python*. Sebastopol, CA: O'Reilly Media.
2. 1, U., & Hewgill, G. (2009, January 22). Why does 1+++2 = 3? Retrieved February 04, 2021, from https://stackoverflow.com/questions/470139/why-does-12-3