**1. Create an assert statement that throws an AssertionError if the variable spam is a negative integer.**

y = -5

assert y > 0

**---------------------------------------------------------------------------**

**AssertionError** Traceback (most recent call last)

Input **In [18]**, in <cell line: 3>**()**

1 y = -5

**----> 3** **assert** y > 0

**AssertionError**:

**2. Write an assert statement that triggers an AssertionError if the variables eggs and bacon contain strings that are the same as each other, even if their cases are different (that is, 'hello' and 'hello' are considered the same, and 'goodbye' and 'GOODbye' are also considered the same).**

**eggs = 'hello'**

**bacon = 'Hello'**

**assert eggs == bacon**

**---------------------------------------------------------------------------**

**AssertionError** Traceback (most recent call last)

Input **In [20]**, in <cell line: 4>**()**

1 eggs = 'hello'

2 bacon = 'Hello'

**----> 4** **assert** eggs == bacon

**AssertionError**:

**3. Create an assert statement that throws an AssertionError every time.**

assert False

**---------------------------------------------------------------------------**

**AssertionError** Traceback (most recent call last)

Input **In [25]**, in <cell line: 1>**()**

**----> 1** **assert** **False**

**AssertionError**:

**4. What are the two lines that must be present in your software in order to call logging.debug()?**

import logging

logging.basicConfig(filename='programLog.log', level=logging.DEBUG)

**5. What are the two lines that your program must have in order to have logging.debug() send a logging message to a file named programLog.txt?**

import logging

logging.basicConfig(filename='programLog.log', level=logging.DEBUG, format='%(asctime)s %(name)s %(levelname)s %(message)s' )

**6. What are the five levels of logging?**

Five levels of Logging

Debug

Info

Warning

Error

Critical

**7. What line of code would you add to your software to disable all logging messages?**

To disable all logging we can add

logging.disable()

**8.Why is using logging messages better than using print() to display the same message?**

Logging gives us more control over print(), we can disable logging message without removing the logging function calls, we can selectively disable lower level logging message. Logging provides a timestamp.

**9. What are the differences between the Step Over, Step In, and Step Out buttons in the debugger?**

Step Over - This will execute the function button will quickly execute the function call without stepping into it

Step in - This will move the debugger into a function call

Step Out – This will quickly execute the rest of the code until it steps out of the function it currently is in.

**10.After you click Continue, when will the debugger stop?**

After clicking continue, the debugger stops once the execution of current program is completed

**11. What is the concept of a breakpoint?**

In software development, a breakpoint is an intentional stopping or pausing place in a program, put in place for debugging purposes. It is also sometimes simply referred to as a pause.