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

a=5

assert a > 5

print('The spam variable must be a negative integer.')

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 = 'good bye'

assert eggs.lower() != bacon.lower(), 'eggs/bacon should not be the same!

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

assert False, 'This assertion always triggers.'

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

The below are the two lines that must be present in order to call logging.debug():

import logging

logging.basicConfig(level=logging.DEBUG, format=' %(asctime)s - %(levelname)s - %(message)s')

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?

The below are the lines that must be present in order to send logging.debug() send a logging message to a file named programLog.txt:

import logging

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

6. What are the five levels of logging?

The five levels of logging are DEBUG, INFO, WARNING, ERROR, and CRITICAL.

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

The line of code that would disable all logging messages is :

logging.disable(logging.CRITICAL)

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

It gives the ability to control and also configure it according to our requirements. It also helps define what information should be included in the logs.

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

Step over- In Step over it will debug the given line and result will be returned directly.

Step In- In Step In If the line does not contain a function it behaves the same as “step over” but if it does the debugger will enter the called function and continue line-by-line debugging there.

Step out- In Step out it returns the debugger to the line where the current function is being called.

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

We need to click on Go, the debugger will stop when it has reached the end of the program with a breakpoint.

11. What is the concept of a breakpoint?

The breakpoint is used to pause the debugging process when the program execution reaches at a defined endpoint.