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

**Answer: assert spam >= 0, 'The spam variable is Negative.'**

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).

**Answer: assert eggs.lower() != bacon.lower() or eggs.upper() != bacon.upper(), "The eggs and bacon variables are the same!"**

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

**Answer: assert False, 'This is an error !'**

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

**Answer:**

**import logging**

**logging.basicConfig( level=** **logging.DEBUG ,format='%(asctime)s %(levelname)s-%(message)s', datefmt='%Y-%m-%d %H:%M:%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?

**Answer:**

**import logging**

**logging.basicConfig(level=** **logging.DEBUG ,filename=** **programLog.txt',filemode='a',format='%(asctime)s %(levelname)s-%(message)s', datefmt='%Y-%m-%d %H:%M:%S')**

6. What are the five levels of logging?

**Answer: 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?

**Answer: logging.disable(logging.CRITICAL) will diable all loggin messages.**

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

**Answer: With the help of logging messages we can easily debug and identify the probable cause , it gives us all the flexibility, what needs to disabled , what need to be ignored and also we get additional information like timestamp with our log messages. Whereas print() simply prints the error message.**

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

**Answer: Step In button will move the debugger into a function call.**

**Step Over button will quickly execute the function call without stepping into it.**

**Step Out button 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 ?

**Answer: debugger will stop if any breakpoint is encountered.**

11. What is the concept of a breakpoint?

**Answer: breakpoint works as a pauser, it causes the debugger to pause when the program execution reaches that line.**