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

Ans: assert spam >= 0, "spam cannot 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).**

Ans: assert eggs.lower() != bacon.lower(), "eggs and bacon should not be the same"

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

Ans: assert False, "This assertion will always trigger"

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

Ans: import logging

logging.basicConfig(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?**

Ans: import logging

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

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

Ans: DEBUG

INFO

WARNING

ERROR

CRITICAL

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

Ans: logging.disable(logging.CRITICAL)

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

Ans: Logging messages can be easily controlled and configured to show or hide messages based on severity levels. Logging messages can be directed to different outputs (console, files, etc.). Logging provides consistent formatting and timestamping of messages. Logging can be enabled or disabled globally, which is useful for debugging and production environments.

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

Ans: Step Over: Executes the current line of code and advances to the next line in the current function. If the current line contains a function call, it will not step into the function but will execute it as a single step.

Step In: If the current line contains a function call, this will step into the function and start debugging within that function.

Step Out: This will continue execution until the current function returns and control is back to the caller. It's useful for stepping out of a function's debugging context.

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

Ans: After you click Continue, the debugger will stop when it encounters the next breakpoint (if any) or when the program execution completes.

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

Ans: A breakpoint is a designated point in the source code where the debugger will pause the program's execution so that you can inspect variables, step through the code, and diagnose issues. It allows you to interactively examine the program's state and behavior at a specific location in the code, aiding in the process of debugging and identifying problems in your software.