1. assert spam >= 0, 'spam should be a non-negative integer'

2. assert eggs.lower() != bacon.lower(), 'eggs and bacon should not be the same strings, even if their cases are different'

3. assert False, 'this assertion always triggers an AssertionError'

4. The following two lines must be present in your software in order to call logging.debug():

import logging

logging.basicConfig(level=logging.DEBUG)

5. The following two lines should be added to your program in order to have logging.debug() send a logging message to a file named programLog.txt:

import logging

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

6. The five levels of logging, in increasing order of severity, are: DEBUG, INFO, WARNING, ERROR, and CRITICAL.

7. To disable all logging messages, you can add the following line of code to your software:

logging.disable(logging.CRITICAL)

8. Using logging messages is better than using print() because it allows for more fine-grained control over the level of detail in the messages, and it can easily be configured to output messages to different destinations (e.g. console, file, email, etc.) based on the severity of the message.

9. The Step Over button allows you to execute the current line of code and move to the next line, without stepping into any function calls. The Step In button allows you to step into the first function call on the current line of code. The Step Out button allows you to execute the remaining lines of code in the current function and return to the calling function.

10. After you click Continue, the debugger will stop at the next breakpoint, or at the end of the program if there are no more breakpoints.

11. A breakpoint is a point in the code where the debugger will stop and allow you to inspect the current state of the program. You can set breakpoints manually in your code using the debugger's interface, or you can use conditional breakpoints to stop the program only when certain conditions are met.