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

**Ans: assert int(spam) >= 0**

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()**

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

**Ans: assert False**

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 and critical.**

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

**Ans: logging.shutdown()**

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

Ans: **Logging hassevaral advantages over print() statement:**

* **It has separate levels based on severity and can be switched on or off.**
* **The messages can be stored in a separate file.**
* **Does not need an output console to display messages.**
* **In live production software, logged messages can be easily checked to find where the program got stuck or crashed.**

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

**Ans: Step over: Debugger executes the program statement by statement within the current execution context.**

**Step in: Debugger executes the program statement by statement.**

**Step out: If the debugger is within a nested scope, this action proceeds until the function exits the current execution context.**

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

**Ans: Debugger will only stop at user defined breakpoints.**

11. What is the concept of a breakpoint?

**Ans: A breakpoint is manually set by user at a point up to which the user thinks the program will run smoothly (ideally just before the point of defect) so that the program stops the continuous execution from start till the breakpoint and then from that point, each line of code can be individually checked for errors. This helps to avoid stepping over each line of code from start to the point of defect.**