1. Create an assert statement that throws an AssertionError if the variable spam is a negative integer.  
Ans:   
spam = 10  
assert spam > 0, 'Negative integer is not allowed'  
print(spam)

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:  
eggs = 'hello'  
bacon = 'good bye'  
assert eggs.lower() != bacon.lower(), 'We are expecting eggs and bacon should have the same string value'  
print("eggs and bacon is having the same value")

3. Create an assert statement that throws an AssertionError every time.  
Ans:   
assert False, 'Always triggers an AssertionError'

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

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:   
logger = logging.getLogger()  
logger.disabled = True

8.Why is using logging messages better than using print() to display the same message?  
Logging can be disabled and enabled any time and also, log level (i.e. 1 to 5) can be set to sent specific type of logs messages. Whereas pirnt() is useful only for debugging code during development phase but it has to be removed from production ready code.

9. What are the differences between the Step Over, Step In, and Step Out buttons in the debugger?  
Ans: These are following buttons in the debugger -  
Step Over - When the next statement to execute reaches a method call, it get executed as a whole and stop.  
Step In - When the next statement to execute reaches a method call, it doesn’t get executed as a whole, but rather, the only first line of that method get executed and stop there.  
Step Out - Finish of executing the callee’s code and stop when execution returns to the caller.

10.After you click Continue, when will the debugger stop ?  
Ans: It will stop at the next break point when Continue is clicked while debugging code.

11. What is the concept of a breakpoint?  
Ans: Breakpoint stops the execution of code so that the lines below the breakpoint can be debugged.