**Python – Assignment 11**

|  |  |
| --- | --- |
| **S. No.** | **Question / Answer** |
| 1 | Create an assert statement that throws an AssertionError if the variable spam is a negative integer. |
|  | assert type(spam) is int and spam >= 0, ‘Spam is 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). |
|  | assert eggs.lower() != bacon.lower(), ‘eggs and bacon values are the same’ |
| 3 | Create an assert statement that throws an AssertionError every time. |
|  | assert False, ‘throwing an assertion error always’ |
| 4 | What are the two lines that must be present in your software in order to call logging.debug()? |
|  | 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? |
|  | import logging  logging.basicConfig(filename=’programLog.txt’, level=logging.DEBUG, format=’%(asctime)s - %(levelname)s - %(message)s’) |
| 6 | What are the five levels of logging? |
|  | DEBUG, INFO, WARNING, ERROR, CRITICAL |
| 7 | What line of code would you add to your software to disable all logging messages? |
|  | Logging.disable(logging.CRITICAL)  This disables all logs with levels CRITICAL and below. Thus all logs are disabled. |
| 8 | Why is using logging messages better than using print() to display the same message? |
|  | Logging messages is already a robust interface that provides levels and timestamps. Logging messages enable improved monitoring of the program execution, unlike plain print() statements. Logging messages also provide the facility to selectively disable log levels without removing log statements. |
| 9 | What are the differences between the Step Over, Step In, and Step Out buttons in the debugger? |
|  | Step Over: The debugger moves to the next line of code in sequence.  Step In: The debugger enters the function definition of the function call in the current line.  Step Out: The debugger steps out of the function definition to the next line of the function call line. |
| 10 | After you click Continue, when will the debugger stop ? |
|  | The debugger will stop at the next breakpoint or when the code is entirely executed if there are no further breakpoints. |
| 11 | What is the concept of a breakpoint? |
|  | A breakpoint is the line of code, where the normal flow of execution is halted, so that we can observe the variable in the memory. Breakpoint helps us debug the code flow. |