**Python\_Assingment\_11**

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

Ans – Assert spam>=10, The spam variable is less then 10.

**Ques – 2 Write an assert statement that triggers an assertionError if the variable eggs and bacon contain strings that are the same as each other, even if their case are different ( that is, “hello” and “hello” are considered the same, and ‘goodbye’ and ‘GOODbye’ are also consider the same)?**

Ans – assert eggs.lower() ! = bacon.lower, ‘ The eggs and bacon variable are same! Or assert eggs.upper()! = bacon.upper(), , The eggs and bacon variable are the same.

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

Ans – assert False, ‘This assertion always triggers’.

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

Ans – import logging logging.basicconfig(level=logging.DEBUG, formate=’%(asctime )s-%(levename)s-%(message)s,)

**Ques – 5 What are the two lines that your program must have in order to have logging.debug() send a logging message to a file name programlog.txt?**

Ans – import logging

loggingmbasicComfig(filename) = ‘programLog.txt’,

level = logging.DEBUG,

format = ‘%(asctime)s-%(levelname)s-%(message)s’

**Ques – 6 What are the five level of logging?**

Ans – DEBUG, INFO, WARNING, ERROR, and CRITICAL

**Ques – 7 What line of code would you add to your software to disable all logging software message?**

Ans – logging.disable(logging.CRITICAL)

**Ques – 8 Why is using logging message better then using the print() to display tha same message?**

Ans – You can disable logging message without removing the logging function call. You can selectively disable lower-level logging message. You create logging message.Logging messages provides a timestamp.

**Ques – 9 What are the different between the step over, step in, and step Out button in the debugger?**

Ans – The button will move the debugger into a function call. The Over button will quick execute the function call without stepping into it. The Out button will quick execute the rest of the code until it steps out of the function it currently is in.

**Ques – 10 After you click continue, when will the debugger stop?**

Ans – After you click go, the debugger will stop when it has reached the end of the program or a lone with a breakpoint.

**Ques – 11 What is the concept of a breakpoint?**

Ans – A breakpoint is a setting in a line of code that causes the debugger to pause when the program execution reaches the line.