1. We can create AssertionError with the keyword ‘assert’

var = 25

assert var < 0, ‘Variable must be less than 0’

2. It can be done as,

eggs = ‘hello’

beacon = ‘Hello’

assert eggs.lower() != beacon.lower(), ‘Variable eggs and beacon must not be equal’

3. We can use reaise\_assert() to raise assertion error everytime,

assert False, 'Always raise Assertion Error.'

4. There is two things that must be there before we start using logging

import logging

and set the basic configuration for the logging module

logging.basicConfig(

filename,

level,

format

)

5. The two things are.

Import logging

logging.basicConfig(

finelname=‘programLog.txt’,

level=logging.debug(),

format=' %(asctime)s - %(levelname)s - %(message)s'

)

This will write the debug in the ‘programLog.txt’ file.

6. There are six levels of logging

CRITICAL

ERROR

WARNING

INFO

DEBUG

NOTSET

7. To disable logging in the program, we can use

logging.disable = True

8. The main difference is when we use print(), will always print the value, but with logging we can choose what we print based on the level of the informaton, also with the logging package we can even disable printing while with the print statement we need to remove manually.

9. There are explained as under:

step in – this command and advances the app execution one statement at a time

step out – when we want to continue our debugging session but just want to run the current function all the way

step over – with the help of step over you skip the code or function or method that you are no interested at this time to run

10. The continue button will run the program mornally until the breakout point.