1. raise statement - This is used to raise an inbuilt exception explicitly.

assert statement - when condition in assert statement is true, program is executed normally. When condition is false, program gives an AssertionError.

2. When exceptions are raised, then the exception is handled by the same class mentioned in except statement or its base classes. "Exception" is the base class(Superclass) for all other exception classes.

3. By using logging.exception() method, we can log log messages with level ERROR on the logger. I will store the information like type of error, line number, file name and path.

By using raise keyword. It will also display Traceback of the recent exception raised.

4. We can print the text of an exception, by using the object of an Exception class. We can access different attributes like messgae, args, traceback from that object.

By using logging.error() method

try:

a = 'abc'

b = 2

c = a+b

except TypeError as e:

print(e)

can only concatenate str (not "int") to str

try:

file = open('one.txt','r')

except Exception as e:

logging.error(e)

ERROR:root:[Errno 2] No such file or directory: 'one.txt'

5. In Python versions 1.5 and earlier, Exceptions were strings and not classes. Now all exceptions are classes, derived from the superclass BaseException. With classes it is more efficiently handled as classes have many useful features.