1. 1) raise statement: With the help of raise keyword we can create a user defined exceptions

2) assert statement: With the help of the assert keyword we can create an assertion error.

1. We uses try and except bock for printing out the exceptions we have raised by using raise statement. Example is:

a=1

try:

if a<5:

raise Exception('This is not under the condition')

print('Value if a is: ',a)

except Exception as e:

print(e)

This is not under the condition

1. Following are the two ways through whcih we can give context information to the exception object. Here object is e and we are providing the information to it

# 1)

a=1

try:

if a<5:

raise Exception('This is not under the condition')

print('Value if a is: ',a)

except Exception as e:

print(e)

This is not under the condition

# 2)

a=1

b=0

try:

div=a/b

except Exception as e:

print('Not applicable')

else:

print(div)

Not applicable

1. Name two ways that you can specify the error message text for exception objects.

# way 1)

a=1

b=0

try:

div=a/b

except Exception as e:

print(f'Error is: {e}')

else:

print(div)

Error is: division by zero

# way 2

a=1

b=0

try:

div=a/b

except ZeroDivisionError as e:

print(f'Error is: {e}')

Error is: division by zero

1. String based exceptions are the messages which we give by ourself so that when error occurs then instead of original exception error message, we get that message which we have written in the string form. But this has some drawback. Suppose an error occured and we wanted to know that which kind or type of error it is, there it failed because we have given the customised error message.

So even if wanted to change the mistake occured associated with that error, will be difficult to rectify.