# Exceptions and Exception Handling

## Exercises

### Week 9

Prior to attempting these exercises ensure you have read the lecture notes and/or viewed the video, and followed the practical. You may wish to use the Python interpreter in interactive mode to help work out the solutions to some of the questions.

Download and store this document within your own filespace, so the contents can be edited. You will be able to refer to it during the test in Week 6.

Enter your answers directly into the highlighted boxes.

For more information about the module delivery, assessment and feedback please refer to the module within the MyBeckett portal.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## ©2020 Mark Dixon / Tony Jenkins

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What is the purpose of a try block in the Python language?

*Answer:*

The purpose of a try block is to run code which may raise errors which can then be handled by the except block gracefully.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What **type** of *exception* would be *raised* during evaluation of the following expression?

value = total / 0

*Answer:*

ZeroDivisionError

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What **type** of *exception* would be *raised* during evaluation of the following expression?

num = int("xyz")

*Answer:*

ValueError

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What happens if an *exception* occurs when a program is running, but is not handled by a try...except block within the code?

*Answer:*

It will exit/ crash the program and return a traceback message to the cause of the exception.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Should *syntax errors* be handled using a try...except block or should the code be changed to remove the error?

*Answer:*

Syntax Errors should be handled by changing the code to remove the error and not by try except.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Is it possible to associate more than one except block with a single try statement?

*Answer:*

Yes more than one except block can be used in a single try to properly assess each exception separately.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

When an else statement follows a try...except block, under what circumstances is the code contained within the else block executed?

*Answer:*

The else block after a try…except block only executed if none of the except blocks are executed.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

When a finally statement follows a try...except block, under what circumstances is the code contained within the finally block executed?

*Answer:*

The code within a finally block is always executed.

Does a finally statement always need to be present?

*Answer:*

No, a finally statement need not always be present. It is only needed in case of some cleanup that is absolutely needed. It runs even if the try block contains a return statement.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What **type** of *exceptions* could be *raised* during execution of the following program?

pos = int(input("Enter a number between 1 and 6"))

hex\_vals = [ "A", "B", "C", "D", "E", "F" ]

letter = hex\_vals[pos-1]

*Answer:*

ValueError

IndexError

Write a small program that surrounds the above statements with a try...except block to handle the possible exceptions. Print a message reporting the problem encountered.

*Answer:*

try:

pos = int(input("Enter a number between 1 and 6"))

hex\_vals = [ "A", "B", "C", "D", "E", "F" ]

letter = hex\_vals[pos-1]

except ValueError:

print("The value entered is not a number")

except IndexError:

print("The value entered is too large for hex vals")

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

What statement is used in the Python language to *cause* an exception to occur?

*Answer:*

raise

Give a simple example -

*Answer:*

raise ValueError("The given value is not a number")

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Write an example try...except statement that displays a message accessed from the *exception object* within the except block.

*Answer:*

try:

int(input("Enter a letter to cause an exception"))

except Exception as e:

print("Error",e)

Is a *traceback* message the best way of reporting run-time errors to an end-user?

*Answer:*

No a traceback error is not the best way to report run-time errors to end-users as it is cryptic and might end up causing code to be leaked if it were not meant to be.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## **Exercises are complete**

Save this logbook with your answers. Then ask your tutor to check your responses to each question.