A Short and Incomplete Introduction to Python

Part 6: Exception handling

Sigve Haug <sigve.haug@math.unibe.ch>,
Alexander Kashev <alexander.kashev@math.unibe.ch>
Science IT Support (ScITS), University of Bern

Based on a course by Riccardo Murri / Sergio Maffioletti
S3IT: Services and Support for Science IT, UZH

August 20–21, 2018

Exception handling

Exceptions

"Exceptions" is the name given in Python to error conditions.

You can write code that intercepts some error conditions and reacts appropriately.

See also: http://docs.python.org/library/exceptions.html

What does an exception look like?

```
>>> stream.write('foo')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IOError: File not open for writing
```

What does an exception look like?

```
>>> stream.write('foo')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IOError: File not open for writing
```

This is the exception *message*: it is supposed to be read by the (human) user.

What does an exception look like?

```
>>> stream.write('foo')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IOError: File not open for writing
```

This is the exception *class name*; it is used for catching exceptions (syntax in the next slide).

```
try:
  # code that might raise an exception
except SomeException:
  # handle some exception
except AnotherException as ex:
  # the actual Exception instance
  # is available as variable 'ex'
finally:
  # performed on exit in any case
```

The optional **`finally`** clause is executed on exit from the **`try`** or **`except`** block in *any* case.

Reference: http://docs.python.org/reference/compound_stmts.html#try

Common Exception types

ArithmeticError	Catch-all class for all class of number manipulation errors.
IOError	I/O error on open file.
IndexError	Position `i` out of bounds in a sequence access like `L[i]`
KeyError	Key `k` does not exist in a dictionary/mapping access like `D[k]`.
OSError	A system call failed.
TypeError	Argument of wrong type passed to function. For example: a `datetime` object passed to `int()` or `float()`.
ValueError	Argument has the right type but an invalid value. For example: convert a string to integer but string does not contain a number.

For more, see:
https://docs.python.org/3/library/exceptions.html

Raising exceptions in your code

Use the **`raise`** statement with an `Exception` instance:

```
if an_error_occurred:
  raise RuntimeError("Spider sense is tingling.")
```

Exercise 6.A: Try loading file `values2.txt` with the `load_data()` function from Exercise 4.A – what exception does Python raise?

Edit the `load_data()` function into a `load_data2()` that *ignores* any line that does not contain an integer number.

Bonus points if you can write `load_data2()` so that it has exactly the same output of `load_data()`, i.e. minimize the number of rejected input lines.

Exercise 6.B: Write a function `read_csv(p)` which reads a CSV (*Comma-sSparated Values*) file and returns a list of all rows in it. A *row* will be represented as a Python list of (string) items.