Εξαιρέσεις

Μιχάλης Δρακόπουλος

Μάϊος 2020

ΕΞΑΙΡΕΣΕΙΣ (EXCEPTIONS)

Είναι **σφάλματα** που **ανιχνεύονται** κατά την **εκτέλεση** ενός προγράμματος και τα οποία μπορούμε να τα **χειριστούμε**

Παραδείγματα εξαιρέσεων

[1]:
```
x = 5/0
```

```
---------------------------------------------------------------------------

ZeroDivisionError                         Traceback (most recent call last)

<ipython-input-1-fc2abf138dd5> in <module>()
----> 1 x = 5/0


ZeroDivisionError: division by zero
```

[2]:
```
L = [10, 20, 30]
L[4]
```

```
---------------------------------------------------------------------------

IndexError                                Traceback (most recent call last)

<ipython-input-2-fc93be0dbed4> in <module>()
      1 L = [10, 20, 30]
----> 2 L[4]


IndexError: list index out of range
```

[3]:
```
2 + 3*y
```

```
---------------------------------------------------------------------------

NameError                                 Traceback (most recent call last)

<ipython-input-3-ff3c7185a200> in <module>()
----> 1 2 + 3*y


NameError: name 'y' is not defined
```

[4]:
```
2 + '12'
```

```
---------------------------------------------------------------------------

TypeError                                 Traceback (most recent call last)

<ipython-input-4-18a3ae790887> in <module>()
----> 1 2 + '12'


TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

[5]:
```
int('42')
```

[5]: 42

[6]:
```
int('spam')
```

```
---------------------------------------------------------------------------

ValueError                                Traceback (most recent call last)

<ipython-input-6-33ef1fcca616> in <module>()
----> 1 int('spam')


ValueError: invalid literal for int() with base 10: 'spam'
```

Κάθε εξαίρεση έχει ένα **προκαθορισμένο μήνυμα** για το σφάλμα το οποίο την προκαλεί.

Χειρισμός εξαιρέσεων (exception handling)

Οταν συμβεί μια εξαίρεση και δεν τη χειριστούμε (unhandled exception), η εκτέλεση του προγράμματος τερματίζεται.

[7]:
```
passed = int(input('Number of students passed? '))
failed = int(input('Number of students failed? '))
ratio = passed/failed
print('Passed/Failed ratio is', ratio)
print('Number of students tested: ', passed + failed)
```

```
Number of students passed?  100
Number of students failed?  0


    ---------------------------------------------------------------------------

    ZeroDivisionError                         Traceback (most recent call last)

    <ipython-input-7-6e6d84e3d67e> in <module>()
          1 passed = int(input('Number of students passed? '))
          2 failed = int(input('Number of students failed? '))
    ----> 3 ratio = passed/failed
          4 print('Passed/Failed ratio is', ratio)
          5 print('Number of students tested: ', passed + failed)


    ZeroDivisionError: division by zero
```

Μπορούμε όμως να **χειριστούμε** μια εξαίρεση με την εντολή `try`

```
try:
    statements_to_test_for_exception
except ExceptionName:
    statements_to_execute_when_ExceptionName_is_raised
other_statements
```

[8]:
```
passed = int(input('Number of students passed? '))
failed = int(input('Number of students failed? '))
try:
    ratio = passed/failed
    print('Passed/Failed ratio is', ratio)
except ZeroDivisionError:
    print('No failures!!!')
print('Number of students tested: ', passed + failed)
```

```
Number of students passed?  100
Number of students failed?  0
```

```
No failures!!!
Number of students tested:  100
```

Παράδειγμα Είσοδος αριθμών

[9]:
```
val = int(input('Enter an integer: '))
print('The square of the number is: ', val**2)
```

```
Enter an integer:  spam
```

```
---------------------------------------------------------------------------

ValueError                                Traceback (most recent call last)

<ipython-input-9-9b3fc5ab501c> in <module>()
----> 1 val = int(input('Enter an integer: '))
      2 print('The square of the number is: ', val**2)


ValueError: invalid literal for int() with base 10: 'spam'
```

[10]:
```
while True:
    val = input('Enter an integer: ')
    try:
        val = int(val)
        print('The square of the number is: ', val**2)
        break
    except ValueError:
        print(val, 'is not an integer')
```

```
Enter an integer:  spam
spam is not an integer
Enter an integer:  12
The square of the number is:  144
```

Σε γενικότερη μορφή ως συνάρτηση εισόδου για διάφορους τύπους δεδομένων (**πολυμορφισμός**):

[11]:
```
def read_value(val_type, in_message, err_message):
    while True:
        val = input(in_message + ' ')
        try:
            return val_type(val)
        except ValueError:
            print(err_message)
```

[14]:
```
read_value(int, 'Enter an integer:', 'Not an integer')
```

```
Enter an integer:  abc

Not an integer

Enter an integer:  42
```

[14]: 42

Παραλλαγές της εντολής `try`

- Μπορούμε να πιάσουμε πολλές εξαιρέσεις με μία εντολή `try`

```
try:
    statements_to_test_for_exception
except ExceptionName1:
    statements_to_execute_when_ExceptionName1_is_raised
except ExceptionName2:
    statements_to_execute_when_ExceptionName2_is_raised
    ...
except ExceptionNameN:
    statements_to_execute_when_ExceptionName_is_raised
other_statements
```

- Υπάρχει η δυνατότητα να πιάσουμε όλες τις πιθανές εξαιρέσεις με έναν μόνο κλάδο `except` (**δεν συνιστάται** εκτός ειδικών περιπτώσεων)

```
try:
    statements_to_test_for_exception
except:
    statements_to_execute_when_any_Exception_is_raised
other_statements
```

- Η `try` με κλάδο `else`:

```
try:
    statements_to_test_for_exception
except ExceptionName:
    statements_to_execute_when_ExceptionName_is_raised
else:
    statements_to_execute_when_NO_exception_is_raised
other_statements
```

Ο κλάδος `else` πρέπει να βρίσκεται μετά από όλους τους χειριστές `except` της εντολής `try`

[15]:
```
passed = int(input('Number of students passed? '))
failed = int(input('Number of students failed? '))
try:
    ratio = passed/failed
except ZeroDivisionError:
    print('No failures!!!')
else:
    print('Passed/Failed ratio is', ratio)
```

```
print('Number of students tested: ', passed + failed)
```

```
Number of students passed?  100
Number of students failed?  0

No failures!!!
Number of students tested:  100
```

- Γενική μορφή της εντολής `try` / ο κλάδος `finally`

```
try:
    statements_to_test_for_exception
except ExceptionName1:
    statements_to_execute_when_ExceptionName1_is_raised
except ExceptionName2:
    statements_to_execute_when_ExceptionName2_is_raised
    ...
except ExceptionNameN:
    statements_to_execute_when_ExceptionName_is_raised
except:
    statements_to_execute_when_any_OTHER_Exception_is_raised
else:
    statements_to_execute_when_NO_exception_is_raised
finally:
    statements_to_execute_ALWAYS
other_statements
```

Ο κλάδος `finally`, αν υπάρχει, εκτελείται **πάντα** ανεξάρτητα αν προκλήθηκε ή όχι εξαίρεση.

Πρόκληση εξαίρεσης

Με την εντολή `raise`

[16]:
```
def get_month():
    month = int(input('Enter current month (1-12): '))
    if month < 1 or month > 12:
        raise ValueError('Invalid month value')
```

Κατά την εκτέλεση της παραπάνω συνάρτησης μπορεί να προκληθεί εξαίρεση `ValueError` σε 2 περιπτώσεις.

- είσοδος τύπου εκτός `int` (εμφανίζεται το προκαθορισμένο μήνυμα της εξαίρεσης)

[17]:
```
try:
    month = get_month()
except ValueError as msg:
    print(msg)
```

```
Enter current month (1-12):  2.0
```

```
invalid literal for int() with base 10: '2.0'
```

- είσοδος αριθμού εκτός ορίων (εμφανίζεται το μήνυμα που ορίσαμε εμείς)

[18]:
```
try:
    month = get_month()
except ValueError as msg:
    print(msg)
```

```
Enter current month (1-12):  15
Invalid month value
```

try vs if

Παράδειγμα Λίστα με τους λόγους των στοιχείων 2 λιστών

Με χειρισμό εξαιρέσεων

[19]:
```
def get_ratios(L1, L2):
    ratios = []
    for i in range(len(L1)):
        try:
            ratios.append(L1[i]/L2[i])
        except ZeroDivisionError:
            ratios.append(float('nan'))
        except:
            raise ValueError('get_ratios: Bad arguments')
    return ratios
```

[20]:
```
try:
    print(get_ratios([1, 2, 0, 3, 3], [10, 0, 30, 0, 1]))
    print(get_ratios([1, 2, 3], [10]))
except ValueError as msg:
    print(msg)
```

```
[0.1, nan, 0.0, nan, 3.0]
get_ratios: Bad arguments
```

Με if

[21]:
```
def get_ratios(L1, L2):
    ratios = []
    if len(L1) != len(L2):
        raise ValueError('get_ratios: Bad arguments')
    for i in range(len(L1)):
        e1 = L1[i]
        e2 = L2[i]
        if type(e1) not in (int, float) or type(e2) not in (int, float):
            raise ValueError('get_ratios: Bad arguments')
```

```
        if e2 == 0.0:
            ratios.append(float('nan'))
        else:
            ratios.append(L1[i]/L2[i])
    return ratios
```

[22]:
```
try:
    print(get_ratios([1, 2, 0, 3, 3], [10, 0, 30, 0, 1]))
    print(get_ratios(['abc', 1, 2], [12, 13, 14]))
except ValueError as msg:
    print(msg)
```

```
[0.1, nan, 0.0, nan, 3.0]
get_ratios: Bad arguments
```

Πώς μεταδίδονται οι εξαιρέσεις

- Αν μια συνάρτηση δεν χειρίζεται κάποια εξαίρεση που προκλήθηκε σε αυτήν, μεταφέρει τον χειρισμό της στη συνάρτηση που την κάλεσε. Αυτό συνεχίζεται μέχρις ότου κάποια από τις συναρτήσεις στην αλυσίδα κλήση χειριστεί την εξαίρεση, διαφορετικά το πρόγραμμα τερματίζει λόγω της εξαίρεσης.

[23]:
```
def A():
    print('>>>>>> In A')
    print(5/0)
    print('>>>>>> Leaving A')

def B():
    print('>>> In B')
    print('>>> Calling A')
    A()
    print('>>> Leaving B')

def C():
    print('In C')
    print('Calling B')
    B()
    print('Leaving C')
```

[24]:
```
C()
```

```
In C
Calling B
>>> In B
>>> Calling A
>>>>>> In A
```

```
ZeroDivisionError                         Traceback (most recent call last)

<ipython-input-24-eb6b12b99cce> in <module>()
----> 1 C()


<ipython-input-23-f4cdcf37f422> in C()
     13     print('In C')
     14     print('Calling B')
---> 15     B()
     16     print('Leaving C')


<ipython-input-23-f4cdcf37f422> in B()
      7     print('>>> In B')
      8     print('>>> Calling A')
----> 9     A()
     10     print('>>> Leaving B')
     11


<ipython-input-23-f4cdcf37f422> in A()
      1 def A():
      2     print('>>>>>> In A')
----> 3     print(5/0)
      4     print('>>>>>> Leaving A')
      5


ZeroDivisionError: division by zero
```

[25]:
```
def A():
    print('>>>>>> In A')
    try:
        print(5/0)
    except:
        print('****** Caught exception in A')
    print('>>>>>> Leaving A')

def B():
    print('>>> In B')
    print('>>> Calling A')
    A()
    print('>>> Leaving B')
```

```
def C():
    print('In C')
    print('Calling B')
    B()
    print('Leaving C')
```

[26]:
```
C()
```

```
In C
Calling B
>>> In B
>>> Calling A
>>>>>> In A
****** Caught exception in A
>>>>>> Leaving A
>>> Leaving B
Leaving C
```

[27]:
```
def A():
    print('>>>>>> In A')
    print(5/0)
    print('>>>>>> Leaving A')

def B():
    print('>>> In B')
    print('>>> Calling A')
    try:
        A()
    except:
        print('*** Caught exception in B')
    print('>>> Leaving B')

def C():
    print('In C')
    print('Calling B')
    B()
    print('Leaving C')
```

[28]:
```
C()
```

```
In C
Calling B
>>> In B
>>> Calling A
>>>>>> In A
*** Caught exception in B
>>> Leaving B
```

```
Leaving C
```

Η `try` ως εντολή ελέγχου

Παράδειγμα Αθροισμα ψηφίων μέσα σε συμβολοσειρά

[29]:
```
def sum_digits(s):
    res = 0
    for c in s:
        try:
            res += int(c)
        except:
            pass
    return res
```

[30]:
```
sum_digits('ab1XX2cd4pP')
```

[30]: 7