Βασικοί τύποι, Είσοδος/Έξοδος

Μιχάλης Δρακόπουλος

Μάρτιος 2021

Από το MATLAB στην Python

Μεταβλητές

Για αποθήκευση τιμών.

Επιτρεπτοί χαρακτήρες για ονόματα μεταβλητών: γράμματα πεζά και κεφαλαία, τα ψηφία 0-9, και ο χαρακτήρας _ (κάτω παύλα). Ο πρώτος χαρακτήρας δεν μπορεί να είναι κάποιο από τα ψηφία 0-9!

Η εκχώρηση τιμών σε μεταβλητές γίνεται με τον τελεστή =.

```
variable = value
```

Κάποιοι βασικοί τύποι δεδομένων

> **ΣΗΜΕΙΩΣΗ** Η Python διαθέτει τη συνάρτηση `type()` που επιστρέφει τον τύπο του ορίσματος της.

Αριθμοί

Ο τύπος `int` για ακεραίους και ο τύπος `float` για πραγματικούς αριθμούς (διπλής ακρίβειας), με τους συνήθεις τελεστές `+`, `-`, `*`, `/` και `**` (ύψωση σε δύναμη).

[1]:
```
1 + 1
```

[1]: 2

[2]:
```
2.5 * 2e2 + 12.2
```

[2]: 512.2

[3]:
```
answer = 42
PI = 3.14159
y = 2.021e+3
R = 10.0
```

[4]:
```
type(answer), type(PI), type(y), type(R)
```

[4]: (int, float, float, float)

Δεδομένα `int` και `float` συνδυάζονται χωρίς περιορισμούς σε αριθμητικές εκφράσεις και το αποτέλεσμα αντιστοιχεί πάντα στον σωστό από πλευράς ακρίβειας τύπο.

[5]:
```
x = 1 + 2
type(x)
```

[5]:
```
int
```

[6]:
```
10/3
```

[6]:
```
3.3333333333333335
```

[7]:
```
A = 2*R + 5
A, type(A)
```

[7]:
```
(25.0, float)
```

Η Python έχει επίσης τους τελεστές `//` για ακέραια (floor) διαίρεση, και `%` για υπόλοιπο διαίρεσης.

[8]:
```
year = 1821
last = 1821 % 10
left = year // 10
year, last, left
```

[8]:
```
(1821, 1, 182)
```

[9]:
```
# Convert total minutes to hours and minutes
total_minutes = 734
hours = total_minutes // 60
minutes = total_minutes % 60
hours, minutes
```

[9]:
```
(12, 14)
```

Χρησιμοποιούνται και για `float`:

[10]:
```
10.6 // 2.5, 10.6 % 2.5
```

[10]:
```
(4.0, 0.5999999999999996)
```

Ο τύπος `int` της Python υποστηρίζει πολύ μεγάλους ακέραιους αριθμούς (arbitrary precision). Το μέγεθος τους περιορίζεται μόνο από το μέγεθος της μνήμης του υπολογιστή.

[11]:
```
2**1000
```

[11]:
```
107150860718626732094842504906000181056140481170553360744375038837035105112493612249319837881569585812759467291755314682518714528569231404359845775746985748039345677748242309854210746050623711418779541821530464749835819412673987675591655439460770629145711964776865421676604298316526243868372056680693 76
```

Μερικές χρήσιμες συναρτήσεις της Python

- Οι ενσωματωμένες συναρτήσεις `abs`, `max`, `min`

[12]:
```
abs(-10), min(-5, 2, -1, 0, 100), max(-5, 2, 100)
```

[12]:
```
(10, -5, 100)
```

- Πολλές μαθηματικές συναρτήσεις. Βρίσκονται στο **module** (= είδος βιβλιοθήκης προς το παρόν) `math`. Μερικές από αυτές είναι:

  `sqrt`, `floor`, `ceil`, `cos`, `sin`, `tan`, `exp`, `log`, `log2`, `log10` και οι σταθερές: `pi`, `e`, `inf`

Για να τις χρησιμοποίησουμε πρέπει να εισάγουμε το module `math`

[13]:
```
import math
```

Τώρα τα παραπάνω είναι διαθέσιμα με το *πρόθεμα* `math`

[14]:
```
math.sqrt(2), math.pi, math.cos(-math.pi), math.ceil(1.2), math.log10(1000)
```

[14]:
```
(1.4142135623730951, 3.141592653589793, -1.0, 2, 3.0)
```

Συμβολοσειρές

Αντιστοιχούν στον τύπο `str` της Python.

Περικλείονται είτε σε μονά (`'...'`) είτε σε διπλά (`"..."`) εισαγωγικά, χωρίς διαφορά ανάμεσα στις 2 αναπαραστάσεις.

[15]:
```
language = 'Python'
creator = "Guido van Rossum"
movie = "Monty Python's Life of Brian"
empty = ''              # empty string
creator, movie
```

[15]:
```
('Guido van Rossum', "Monty Python's Life of Brian")
```

[16]:
```
type(creator)
```

[16]:
```
str
```

Μπορεί να εκτείνονται σε παραπάνω από μια γραμμές:
1. Ένας τρόπος είναι να χρησιμοποιήσουμε *τριπλά* εισαγωγικά `'''...'''` ή `"""..."""` 2. Ένας άλλος τρόπος είναι να χρησιμοποιήσουμε τον *ειδικό χαρακτήρα* `\n`

[17]:
```
message1 = '''Hello world!'''
message1
```

[17]:
```
'Hello world!'
```

[18]:
```
message2 = 'Multiline
message'
```

```
  File "<ipython-input-18-3d4382be8d91>", line 1
    message2 = 'Multiline
                         ^
SyntaxError: EOL while scanning string literal
```

[19]:
```
message2 = '''Hello
world
of
Python'''
```

[20]:
```
message3 = "Goodbye\nworld\nof\nMATLAB"
```

Η εσωτερική αναπαράσταση είναι ίδια και για τις 2 περιπτώσεις:

[21]:
```
message3
```

[21]:
```
'Goodbye\nworld\nof\nMATLAB'
```

[22]:
```
message2
```

[22]:
```
'Hello\nworld\nof\nPython'
```

Ερμηνεύονται όμως σωστά στην *έξοδο*:

[23]:
```
print(message2)
```

```
Hello
world
of
Python
```

[24]:
```
print(message3)
```

```
Goodbye
world
of
MATLAB
```

Τελεστές για πράξεις συμβολοσειρών

1. Συνένωση με τον τελεστή +

[25]:
```
'Monty' + "Python"
```

[25]:
```
'MontyPython'
```

[26]:
```
creator + ' is the creator of ' + language
```

[26]: 'Guido van Rossum is the creator of Python'

ΠΡΟΣΟΧΗ: και οι 2 όροι πρέπει να είναι συμβολοσειρές!

[27]:
```
'Computer Science ' + 2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-27-219d31ddd997> in <module>
----> 1 'Computer Science ' + 2

TypeError: can only concatenate str (not "int") to str
```

2. Επανάληψη με τον τελεστή *

[28]:
```
'AA' * 10
```

[28]: 'AAAAAAAAAAAAAAAAAAAA'

[29]:
```
3*language + '!'
```

[29]: 'PythonPythonPython!'

[30]:
```
3*(language + '!')
```

[30]: 'Python!Python!Python!'

Ο ένας όρος πρέπει να είναι τύπου `int`.

Λογικά δεδομένα

Αντιστοιχούν στον τύπο `bool` της Python. Ο τύπος `bool` είναι στην ουσία μια **εξειδίκευση** του τύπου `int`.

Εχουν μόνο 2 δυνατές τιμές: `True` (αντιστοιχεί στο 1) και `False` (αντιστοιχεί στο 0)

[31]:
```
type(True)
```

[31]: bool

Ερμηνεύονται και ως `int`:

[32]:
```
True + 10
```

[32]: 11

Σχηματίζονται με 1. Συγκρίσεις δεδομένων μέσω των **σχεσιακών** τελεστών >, >=, <, <=, ==, !=

[33]:
```
2 > 1, answer != 42, language == 'Python'
```

[33]: (True, False, True)

2. Λογικές πράξεις and (σύζευξη), or (διάζευξη), not (άρνηση).

[34]:
```
answer == 42 and language != 'MATLAB'
```

[34]: True

[35]:
```
x = 2
P = x >= 0 or x < 0
P, type(P)
```

[35]: (True, bool)

[36]:
```
not x > 0
```

[36]: False

Στην Python, πράξεις όπως οι τριπλές ανισότητες έχουν νόημα!

[37]:
```
1 <= x < 3
```

[37]: True

Ισοδυναμεί με

[38]:
```
1 <= x and x < 3
```

[38]: True

Μετατροπή τύπων

Τα ονόματα των τύπων είναι ταυτόχρονα και *ειδικές συναρτήσεις* που μετατρέπουν το όρισμα τους στον αντίστοιχο τύπο, εφόσον αυτό είναι εφικτό.

[39]:
```
float(10), int(math.pi), float("2.71"), int('1999')
```

[39]: (10.0, 3, 2.71, 1999)

[40]:
```
int('12A')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-40-26827b6c5f16> in <module>
----> 1 int('12A')

ValueError: invalid literal for int() with base 10: '12A'
```

[41]:
```
volume = 2
lesson = 'Computer Science ' + str(volume)
```

[42]:
```
lesson
```

[42]: 'Computer Science 2'

[43]:
```
type(lesson), type(volume)
```

[43]: (str, int)

Παρατήρηση Για τον τύπο `bool` κάθε τιμή $\neq 0$ θεωρείται `True`, ενώ οι "μηδενικές" τιμές είναι `False`

[44]:
```
bool(12), bool(3.14), bool('Python')
```

[44]: (True, True, True)

[45]:
```
bool(0.0), bool('')
```

[45]: (False, False)

Εξοδος και είσοδος

Εξοδος με τη συνάρτηση `print()`

`print(arg1, arg2, ..., argN)`

Εμφανίζονται οι **τιμές** των ορισμάτων `arg1`, `arg2`, ..., `argN`. Εσωτερικά, οι τιμές των ορισμάτων μετατρέπονται σε **συμβολοσειρές** προκειμένου να εκτυπωθούν.

[46]:
```
print(lesson)
print(math.e)
print(2*math.pi*10)
```

```
Computer Science 2
2.718281828459045
62.83185307179586
```

[47]:
```
lesson
```

[47]: 'Computer Science 2'

Οι τιμές πολλαπλών ορισμάτων εμφανίζονται με *κενό* μεταξύ τους. Δυνατότητα επιλογής διαχωριστικού με την παράμετρο `sep` (τύπου `str`)

[48]:
```
print(lesson, 'is boring', 1<0)
```

```
Computer Science 2 is boring False
```

[49]:
```
print(lesson, 'is boring', 1<0, sep = '-------')
```

```
Computer Science 2-------is boring-------False
```

Κάθε `print` εμφανίζει τις τιμές των ορισμάτων της και στη συνέχεια *αλλάζει γραμμή*. Η συμπεριφορά αυτή μπορεί να αλλάξει με την παράμετρο `end` (τύπου `str`)

[50]:
```
print('Circle with radius R =', 5)
print('has perimeter', 2*math.pi*5)
```

```
Circle with radius R = 5
has perimeter 31.41592653589793
```

[51]:
```
print('Circle with radius R =', 5, end = ' ----> ')
print('has perimeter', 2*math.pi*5)
```

```
Circle with radius R = 5 ----> has perimeter 31.41592653589793
```

[52]:
```
print('Circle with radius R =', 5, end = '\n ****** ')
print('has perimeter', 2*math.pi*5)
```

```
Circle with radius R = 5
 ****** has perimeter 31.41592653589793
```

Μορφοποίηση εξόδου `format()` Ενας τρόπος ελέγχου του εύρους εκτύπωσης `w` και του πλήθους των δεκαδικών ψηφίων `d` και του τρόπου εμφάνισης ενός πραγματικού `x` είναι με τη συνάρτηση `format()` που επιστρέφει *συμβολοσειρά* `s`.

```
s = format(x, 'w.df')
```

Η δεύτερη παράμετρος είναι συμβολοσειρά και ονομάζεται *τελεστής προσαρμογής* (συντακτικά όπως τους αντίστοιχους τελεστές προσαρμογής της `fprintf()` στο MATLAB).

[53]:
```
format(math.pi, '.3f')
```

[53]: '3.142'

[54]:
```
format(10/3, '15.2f')
```

[54]: '   3.33'

Είσοδος με τη συνάρτηση `input()`

```
variable = input('message')
```

Η `input()` επιστρέφει πάντα **συμβολοσειρά**

[55]:
```
name = input('What is your name? ')
print('Hi', name)
```

```
What is your name?  Guido
```

```
Hi Guido
```

[56]:
```
R = input('Circle radius? ')
```

```
Circle radius?  12
```

[57]:
```
type(R)
```

[57]: str

[58]:
```
area = math.pi*R**2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-58-81ea4f5dd01a> in <module>
----> 1 area = math.pi*R**2

TypeError: unsupported operand type(s) for ** or pow(): 'str' and 'int'
```

που πρέπει να *μετατραπεί* στον κατάλληλο τύπο:

[59]:
```
R = float(R)
type(R)
```

[59]: float

[60]:
```
r = float(input('new radius? '))
type(r)
```

```
new radius?  10
```

[60]: float