



# Bevezetés a Python programozási nyelvbe

Szathmáry László  
Debreceni Egyetem  
Informatikai Kar

## 1. Gyakorlat (2016. szept. 20.)

- bevezető
- a sztring adattípus

(utolsó módosítás: 2016. szept. 19.)



# A tantárgyról

**A tantárgy neve:** Bevezetés a Python programozási nyelvbe

**A tantárgy kódja:** IN[BCD]V381L

A tárgy BSc-s hallgatók számára lett meghirdetve (PTI, MI, GI).

**Előfeltétel:** Magas szintű programozási nyelvek 1

**A gyakorlat honlapja:** <https://arato.inf.unideb.hu/szathmary.laszlo>

**A gyakorlat ideje és helye:**

- *kedd* 14h-16h, IK-104

# Követelmények

A **gyakorlati jegy** megszerzésének egyik feltétele a **rendszeres részvétel** a gyakorlatokon. A félév során legfeljebb 3 hiányzás megengedett. Aki ezt túllépi, az automatikusan elégtelent kap. **Késés:** egy pár perces késést még tolerálok, de egy nagyobb késést már fél hiányzásnak fogok venni. Vagyis két nagyobb késés egy hiányzásnak lesz elszámolva.

A szorgalmi időszakban **2 zárthelyi dolgozat** lesz. Az első **papíron**, a második **számítógép** mellett. A két ZH alapján megajánlok egy jegyet. Nem egyértelmű jegy esetén (pl. 3/4, 4/5) az órai munka és a házi feladatok alapján kerekíték fel vagy le. Ha valaki a **házi feladatok** 80%-ánál kevesebbet old meg, akkor a két zárthelyi dolgozatra kapott jegyek átlagánál gyengébb jegyet fogok megajánlani.

Lesz **javítási lehetőség** is. Aki nincs megelégedve a megajánlott jeggyel, az a szorgalmi időszak utolsó hetében javíthat (vagy ronthat). Ekkor **legfeljebb egy jegyet** lehet javítani/rontani. A hallgatónak néhány konkrét programozási feladatot kell helyben megoldania, illetve ismernie kell a Python programozási nyelvvel kapcsolatos fogalmakat. Akinek mindkét zárthelyi dolgozata elégtelen lett, annak nincs lehetősége javításra.

# Ajánlott irodalom

- *Guido van Rossum*: **Python Tutorial**  
(<http://docs.python.org/tutorial/index.html>, [PDF](#)), 2016
- *Wesley J. Chun*: **Core Python Programming** (2nd Edition), 2006
- *Doug Hellmann*: **The Python Standard Library by Example**  
(Developer's Library), 2011  
[online változat: **Python Module of the Week** (<https://pymotw.com/3/>)]
- *Zed A. Shaw*: **Learn Python The Hard Way**  
(<http://learnpythonthehardway.org/>), 2011
- *Allen B. Downey*: **Think Python** (How to Think Like a Computer Scientist)  
<http://www.greenteapress.com/thinkpython/>, O'Reilly, 2012
- *Mark Summerfield*: **Rapid GUI Programming with Python and Qt**  
(1. fejezet), 2007

# Ajánlott irodalom (folyt.)

- *Gérard Swinnen: Tanuljunk meg programozni Python nyelven* (2005, Python 2.2)  
(online letölthető: <http://mek.oszk.hu/08400/08435/>)
- *Rashi Gupta: Mindentudó Python* (2003, Python 2.2)

## Python 3

- *Mark Pilgrim: Dive Into Python 3* (<http://www.diveintopython3.net/>), 2009
- *Mark Summerfield: Python 3*, Kiskapu Kft., 2009

# Bevezető



- A Python egy általános célú, nagyon magas szintű programozási nyelv.
- Fő tervezési szempont: olvashatóság.
- Interpreteres nyelv, a megírt program azonnal futtatható.
- Multiparadigmás (imperatív, objektumorientált, funkcionális).
- Az első változat 1991-ben jelent meg, nevét a Monty Python csoportról kapta.
- Tervezője Guido van Rossum holland kutató/programozó (1956-ban született). 2005-2012: Google; 2013 január óta: Dropbox.
- Mely nyelvek voltak rá hatással: ABC, ALGOL 68, C, C++, Dylan, Haskell, Icon, Java, Lisp, Modula-3, Perl.
- Mely nyelvekre volt hatással: Boo, Cobra, D, Falcon, Groovy, JavaScript, Ruby, Go.

# Bevezető

- Dinamikus típusokat és automatikus memóriakezelést használ.
- Platformfüggetlen (Unix/Linux, Windows, Mac OS, stb.)
- A Pythonnak igen kiterjedt és széles körű standard könyvtára van („batteries included”), amit még kiegészítenek az egyéb (mások által megírt) publikus modulok („3rd party modules”)<sup>1</sup>.
- Az interpreter és a standard könyvtár teljesen nyílt forrású.
- Könnyen tanulható, egyszerű a szintaxisa. A megírt kód könnyen olvasható.
- A programozói munkát hatékony magas szintű adatszerkezetek segítik. Egyszerűen, ugyanakkor nagyon hatásosan valósítja meg az objektumorientált programozást.

<sup>1</sup> Lásd <http://pypi.python.org> (2016.09.02-án **87,794** csomag volt elérhető; 2016.02.12-én **74,468** csomag volt elérhető).

# Bevezető

- Ideális nyelv szkriptek írásához, illetve gyors alkalmazásfejlesztéshez („rapid application development”).
- Gyors prototípusfejlesztést tesz lehetővé („rapid prototyping”).
- Hasonló programozási nyelvek: Perl, Ruby.
- Tökéletes választás kisebb (pl. 10-20 soros) szkriptekhez, de NEM CSAK erre jó! Nagy méretű, több ezer soros programokat is lehet benne írni úgy, hogy a program áttekinthető marad (modulok, csomagok).
- Két ág létezik: Python 2 és Python 3 (2008. dec.). A 2.7-es széria kiforrott, széles körben támogatott. A jelenlegi és jövőbeli fejlesztések a 3-as szériára koncentrálnak. Új projektet már Python 3-ban érdemes elkezdeni.
- A gyakorlaton a Python 3.5-ös verzióját fogjuk használni. Mivel nagyon sok helyen még mindig a Python 2-t használják (2.7), ezért ki fogunk térni a legfontosabb eltérésekre.



# Linkek

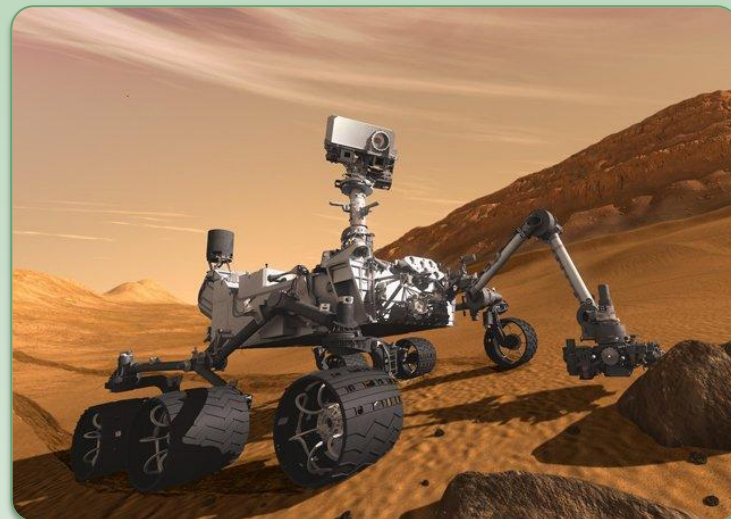
- Python HQ: <http://www.python.org/>
- Python dokumentáció: <http://docs.python.org/>
- A Python Standard Library: <http://docs.python.org/library/>
- Python FAQ: <http://docs.python.org/faq/general.html>
- PEP 8 -- Style Guide for Python Code:  
<http://www.python.org/dev/peps/pep-0008/>
- <http://www.reddit.com/r/learnpython>
- <http://www.reddit.com/r/python>
- <http://stackoverflow.com/questions/tagged/python>

# Hol használják?

- Python sikertörténetek: <http://www.python.org/about/success/>

- **Scientific**

- [Biology](#)
- [Bioinformatics](#)
- [Computational Chemistry](#)
- [Data Visualization](#)
- [Drug Discovery](#)
- [GIS and Mapping](#)
- [Scientific Programing](#)
- [Simulation](#)
- [Weather](#)



Mars Curiosity (2012. aug. 6.)

Szoftver: 2,5 millió C programsor.

A logfile-ok tesztelését Python szkriptekkel végezték.

- Google (C, C++ / Java / Python / Go)  
„Python where we can,  
C++ where we must”  
([link](#))

# Mennyire népszerű?

TIOBE index (<http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>)



Aug 2016	Aug 2015	Change	Programming Language	Ratings	Change
1	1		Java	19.010%	-0.26%
2	2		C	11.303%	-3.43%
3	3		C++	5.800%	-1.94%
4	4		C#	4.907%	+0.07%
5	5		<b>Python</b>	<b>4.404%</b>	<b>+0.34%</b>
6	7	⬆	PHP	3.173%	+0.44%
7	9	⬆	JavaScript	2.705%	+0.54%
8	8		Visual Basic .NET	2.518%	-0.19%
9	10	⬆	Perl	2.511%	+0.39%
10	12	⬆	Assembly language	2.364%	+0.60%
11	14	⬆	Delphi/Object Pascal	2.278%	+0.87%
12	13	⬆	Ruby	2.278%	+0.86%
13	11	⬇	Visual Basic	2.046%	+0.26%
14	17	⬆	Swift	1.983%	+0.80%
15	6	⬇	Objective-C	1.884%	-1.31%
16	37	⬆	Groovy	1.637%	+1.27%
17	20	⬆	R	1.605%	+0.60%
18	15	⬇	MATLAB	1.538%	+0.31%
19	19		PL/SQL	1.349%	+0.21%
20	95	⬆	Go	1.270%	+1.19%

Munkalehetőségek: <http://careers.stackoverflow.com/jobs?searchTerm=python>

# Szakirodalom

amazon Try Prime

All ▾ python programming

Shop by Department ▾ Your Amazon.com Today's Deals Gift Cards Sell Help

1-16 of 2,943 results for "python programming"

Show results for

**Books >**

- Python Programming
- Introductory & Beginning Programming
- Computers & Technology
- Reference
- Programming Languages
- + See more

**Kindle Store >**

- Python Computer Programming
- Computer Programming
- Computers & Technology
- + See more

+ See All 16 Departments

Refine by

**International Shipping**

☐ Ship to Hungary

**Amazon Prime**

☐ Prime

**Eligible for Free Shipping**

Free Shipping by Amazon

**Book Format**

- Audible Audio Edition
- Paperback
- Kindle Edition

**LOOK INSIDE!** Python **36% off**

New Advanced Python Book Bundle This three-book bundle provides advanced techniques in Python programming for web programming, data analysis, system

**Shop The Programmer's Bookshelf**  
Java | C++ | Python | Swift | Ruby | Visual Basic

Related Searches: [python](#), [python programming for the absolute beginner](#), [python programming in context](#).

**PYTHON PROGRAMMING: AN INTRODUCTION TO COMPUTER SCIENCE, 2ND EDITION**  
by John Zelle

Paperback  
\$14.36 to rent Prime  
\$19.99 to buy Prime  
Get it by **Tuesday, Feb 16**

More Buying Choices  
\$19.01 used & new (119 offers)

★★★★★ ▾ 126

Trade in yours for an Amazon C  
FREE Shipping on orders over

**Books:** See all 1,844 items

**Automate the Boring Stuff with Python: Practical Programming for Total Beginners**  
by Al Sweigart

Paperback  
\$24.65 \$29.95 Prime  
Get it by **Tuesday, Feb 16**

More Buying Choices  
\$16.20 used & new (68 offers)

★★★★★ ▾ 124

#1 Best Seller in Introductory

Trade in yours for an Amazon C  
FREE Shipping on orders over

**Books:** See all 1,844 items

# Konferenciák

A legnagyobb konferencia a PyCon US (<https://us.pycon.org/>).

## PyCon 2016



- előadások: <http://pyvideo.org/events/pycon-us-2016.html> (186 videó)
- az előadások között tutorialok is vannak

## PyCon 2015



- előadások: <http://pyvideo.org/events/pycon-us-2015.html> (134 videó)
- az előadások között tutorialok is vannak

A <http://pyvideo.org/> oldalon további konferenciák videóanyagai is fent vannak.

# Kvíz

Hogy hívják a nyelv alkotóját?

- ☐ Larry Wall
- ☐ Yukihiro Matsumoto
- ☐ Guido van Rossum
- ☐ Rasmus Lerdorf

**HF:** Nézzünk utána, hogy kik a többiek.

## Interpreter használata:

```
$ python3
Python 3.5.2 (default, Jul  5 2016, 12:43:10)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> "Hello, World!"
'Hello, World!'
>>>
```

## Szkript írása:

```
1 #!/usr/bin/env python3
2
3 print("Hello, World!")
```

## Ékezetek használata:

```
1 #!/usr/bin/env python3
2 # coding: utf-8
3
4 def main():
5     # megjegyzés ékezettel
6     print("Helló!")
```

Python 2-ben ékezetek esetén  
ezt ki **kellett** írni.

Python 3-ban ez az alapértelmezés,  
de inkább írjuk ki.

```
4 >>> a = 6
5 >>> a
6 6
7 >>> a = "hello"
8 >>> len(a)
9 5
10 >>> a
11 'hello'
12 >>> A
13 Traceback (most recent call last):
14   File "<stdin>", line 1, in <module>
15 NameError: name 'A' is not defined
16 >>> "hello " + "world"
17 'hello world'
18 >>> "hello " + 6
19 Traceback (most recent call last):
20   File "<stdin>", line 1, in <module>
21 TypeError: cannot concatenate 'str' and 'int' objects
22 >>> "hello " + str(6)
23 'hello 6'
```

változót nem kell külön deklarálni



def

```
1 #!/usr/bin/env python3
2
3
4 def main():
5     print("Hello, World!")
6
7
8 main()
```

kettőspont

indentálás

nincs pontosvessző

**Stílus:** egy függvény előtt és után hagyjunk ki 2 üres sort.

```
1 #!/usr/bin/env python3
2
3
4 def main():
5     print("Hello, World!")
6
7
8 if __name__ == "__main__":
9     main()
```

} Direkt módon futtatjuk  
vagy modulként?

Írassuk ki a parancssori argumentumokat:

```
3 import sys
4
5
6 def main():
7     print(sys.argv)
8
9
10 if __name__ == "__main__":
11     main()
```

A továbbiakban nem írom ki külön a  
#!/usr/bin/env python3  
sort...

**Majd:** adjunk meg egy nevet argumentumként (pl. `./hello.py Bob`),  
s üdvözljük az illetőt („Hello Bob!”).

```
3 import sys
4
5
6 def hello(name):
7     if name == 'Batman' or name == 'Robin':
8         print('Batman vagy Robin')
9     else:
10        NincsIlyenFuggveny()
11
12
13 def main():
14     hello(sys.argv[1])
15
16
17 if __name__ == "__main__":
18     main()
```

if után nem kell zárójel

Csak akkor derül ki a hiba, ha idekerül a vezérlés!

Ezért (is) lényegesek a unit tesztek komolyabb rendszerek esetén.  
Minden ágot le kell velük tesztelni.

## Egy általános minta Python 3 szkriptekhez

```
1 #!/usr/bin/env python3
2 # encoding: utf-8
3
4
5 def main():
6     print('Py3')
7
8 #####
9
10 if __name__ == "__main__":
11     main()
```

**Tipp:** mentjük el ezt a file-t `alap.py` néven, majd ha egy új Python programot akarunk írni, akkor egyszerűen csak készítsünk erről egy másolatot.

További minták: <http://goo.gl/9cxBEs>

```
>>> print "hello"
File "<input>", line 1
    print "hello"
          ^
SyntaxError: Missing parentheses in call to 'print'
>>>
>>> print("hello")
hello
>>>
>>> 7 / 2
3.5
>>>
>>> 7 // 2
3
>>>
```

← függvény

← matematikai osztás

← az egész osztás operátora  
(mindig ezt jelenti)

A Python 3  
leglényegesebb  
változásai



```
>>> print "hello"
hello
>>>
>>> 7 / 2
3
>>>
>>> 7 // 2
3
>>>
```

← utasítás

← egész osztás, mint C-ben

← az egész osztás operátora  
(mindig ezt jelenti)

# Sztringek



```
4 >>> s = "Hello"
5 >>> s
6 'Hello'
7 >>> s = 'Hello'
8 >>> s
9 'Hello'
10 >>> s = "isn't"
11 >>> s
12 "isn't"
13 >>> s = 'he said: "go home"'
14 >>> s
15 'he said: "go home"'
16 >>> s = "he said: \"go home\""
17 >>> s
18 'he said: "go home"'
19 >>> s = 'batman'
20 >>> len(s)
21 6
22 >>> s[0]
23 'b'
24 >>> s[0] = 'B'
25 Traceback (most recent call last):
26   File "<stdin>", line 1, in <module>
27   TypeError: 'str' object does not support item assignment
28 >>> s
29 'batman'
30 >>> s + '!'
31 'batman!'
32 >>> s = 'Joker'
33 >>> s.lower()
34 'joker'
35 >>> s.upper()
36 'JOKER'
37 >>> s.find('k')
38 2
39 >>> s.find('a')
40 -1
41 >>> s[20]
42 Traceback (most recent call last):
43   File "<stdin>", line 1, in <module>
44   IndexError: string index out of range
```

## Sztring metódusok:

<https://docs.python.org/3/library/stdtypes.html#string-methods>

strings are *immutable* objects  
(read-only)

**HF:** válasszunk ki egy sztring metódust s írjunk egy kis szkriptet ami bemutatja a használatát.

# Néhány gyakori sztring metódus

`s.lower()`, `s.upper()`

a sztring kisbetűs, nagybetűs verziójával tér vissza

`s.strip()`

a whitespace karaktereket levágja a sztring elejéről és végéről

`s.isalpha()` / `s.isdigit()` / `s.isspace()`...

megnézi, hogy a sztring vmennyi karaktere az adott karakterosztályba tartozik-e

`s.startswith('other')`, `s.endswith('other')`

megnézi, hogy a sztring a másik sztringgel kezdődik-e / végződik-e

`s.find('other')`

A sztringben szerepel-e a másik sztring (nem reguláris kifejezésként adjuk meg).

Ha igen, akkor az első előfordulás első karakterének indexével tér vissza.

Ha nem, akkor -1 a visszatérési érték.

`s.replace('old', 'new')`

a sztringben az 'old' vmennyi előfordulását 'new'-ra cseréli

`s.split('delim')`

A sztringet az adott szeparátor mentén részsstringek listájára bontja. A szeparátor nem reguláris kifejezés. Példa: `'aaa,bbb,ccc'.split(',')` -> `['aaa', 'bbb', 'ccc']`. Ha csak `s.split()` -et írunk, akkor a whitespace karakterek mentén bontja fel a sztringet.

`s.join(list)`

A `split()` ellentéte. Egy lista elemeit kapcsolja össze egy adott szeparátorral (ez lesz az s sztring). Példa: `'---'.join(['aaa', 'bbb', 'ccc'])` -> `aaa---bbb---ccc`.

# A Python egy újabb eszköz



A Pythonra úgy tekintünk, mint egy újabb eszközre az eszköztárunkban.  
Az adott feladathoz a legmegfelelőbb eszközt használjuk.



# Keleti bölcsesség

*"I hear and I forget. I see and I remember. I do and I understand."*

Confucius



Vagyis: gyakorolni [1], gyakorolni [2] és gyakorolni [3][4]...

[1] <https://arato.inf.unideb.hu/szathmary.laszlo/pmwiki/index.php?n=Py.Feladatok>

[2] <http://www.pythonchallenge.com/>

[3] <http://projecteuler.net/>

[4] <https://www.hackerrank.com>

Még egy tipp: a Prog. 2-n kapott feladatokat gyakorlásképpen oldják meg Pythonban is. Fordítva is lehet: az itt kapott feladatokat oldják meg Java/C# nyelven is.

# Tippek és trükkök rovat

## Ajánlott rövidítések:

```
# ~/.bashrc
alias p2='python2'
alias p3='python3'
alias p='python3'

alias ..='cd ..'
alias ...='cd ..; cd ..'
# lehet folytatni...
```

Ne gépeljünk feleslegesen...

# Feladatok

1. [20121001a] sztring metódus

# Python 2 jegyzet



Egy általános minta Python 2 szkriptekhez

```
1 #!/usr/bin/env python2
2 # coding: utf-8
3
4 from __future__ import (absolute_import, division,
5                          |   |   |   |   |   print_function, unicode_literals)
6
7
8 def main():
9     print('Py2→3')
10
11 #####
12
13 if __name__ == "__main__":
14     main()
```

Ez a 4 dolog alapértelmezett a Python 3-ban.  
Ezen speciális import segítségével Python 2.7-ben is be tudjuk őket kapcsolni.

Ha Python 2-ben akarunk programozni, akkor így kezdünk egy új projektet. A speciális `__future__` import hatására úgy tudunk Python 2-ben dolgozni, hogy az már nagyon hasonló a Python 3-hoz. Mire jó? Ha később át akarjuk írni Python 3-ra, akkor már sokkal könnyebb dolgunk lesz.

*division*: osztás, pl. `7 / 2 == 3.5` (mint Python 3-ban)

*print\_function*: a `print` függvény lesz, pl. `print("hello")`