



Bevezetés a Python programozási nyelvbe

Szathmáry László
Debreceni Egyetem
Informatikai Kar

1. Gyakorlat

- bevezető
- a sztring adattípus

(utolsó módosítás: 2019. szept. 6.)



A tantárgyról

A tantárgy neve: Bevezetés a Python programozási nyelvbe

A tantárgy kódja: IN[BCD]V381L

A tárgy BSc-s hallgatók számára lett meghirdetve.

Előfeltétel: Magas szintű programozási nyelvek 1

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

A gyakorlat ideje és helye:

- *kedd* 8.00, IK-205

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*
(<https://docs.python.org/3/download.html>, PDF-ben is, 150 oldal), 2019
- *Wesley J. Chun: Core Python Programming* (2nd Edition), 2006
- *Allen B. Downey: Think Python* (How to Think Like a Computer Scientist)
<http://www.greenteapress.com/thinkpython/>, O'Reilly, 2012
- *Doug Hellmann: The Python Standard Library by Example*
(Developer's Library), 2011
[online változat: **Python Module of the Week** (<https://pymotw.com/3/>)]
- *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)

Ajánlott irodalom (folyt.)

Python 3

- *Mark Pilgrim: Dive Into Python 3* (<http://www.diveintopython3.net/>), 2009
- *Mark Summerfield: Python 3*, Kiskapu Kft., 2009
- *Michael Driscoll: Python 101*, Leanpub, 2014 (kezdő)
- *Michael Driscoll: Python 201*, Leanpub, 2016 (középhaladó)

Expert

- *Luciano Ramalho: Fluent Python*, O'Reilly, 2015

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”)¹.
- 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.

¹ Lásd <https://pypi.org> (2019. 09. 06-án **195,008** csomag volt elérhető; 2019. 02. 06-án **167,357** 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, de a jelenlegi és jövőbeli fejlesztések a 3-as szériára koncentrálnak. Új projektet már Python 3-ban érdemes elkezdeni. A Python 2.7-hez 2020. január 1. után már nem jön ki újabb frissítés!
- A gyakorlaton a Python 3-as verzióját fogjuk használni. A jelenlegi legfrissebb verzió a Python 3.7-es. Mivel nagyon sok helyen még mindig a Python 2-t használják (2.7), ezért ki fogunk térni a legfontosabb eltérésekre. Ajánlott Python verzió: Python 3.6+.

Linkek

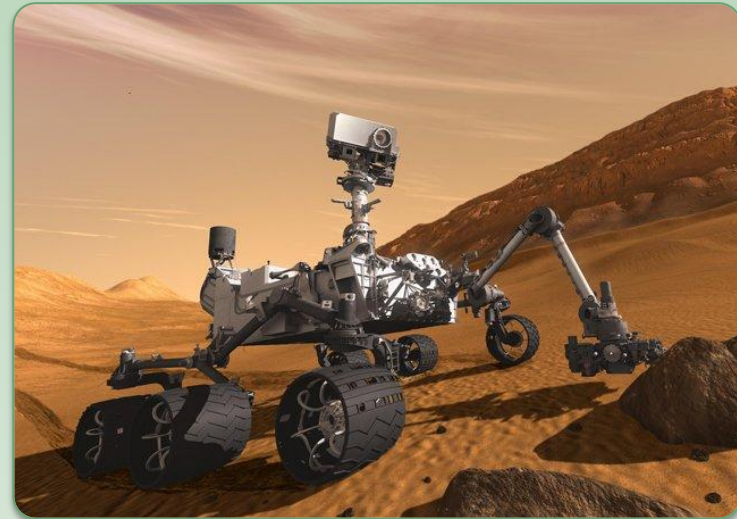
- Python HQ: <https://www.python.org/>
- Python dokumentáció: <https://docs.python.org/>
- A Python Standard Library: <https://docs.python.org/library/>
- Python FAQ: <https://docs.python.org/faq/general.html>
- PEP 8 -- Style Guide for Python Code:
<https://www.python.org/dev/peps/pep-0008/>
- <https://www.reddit.com/r/learnpython>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

Hol használják?

- Python sikertörténetek: <https://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?



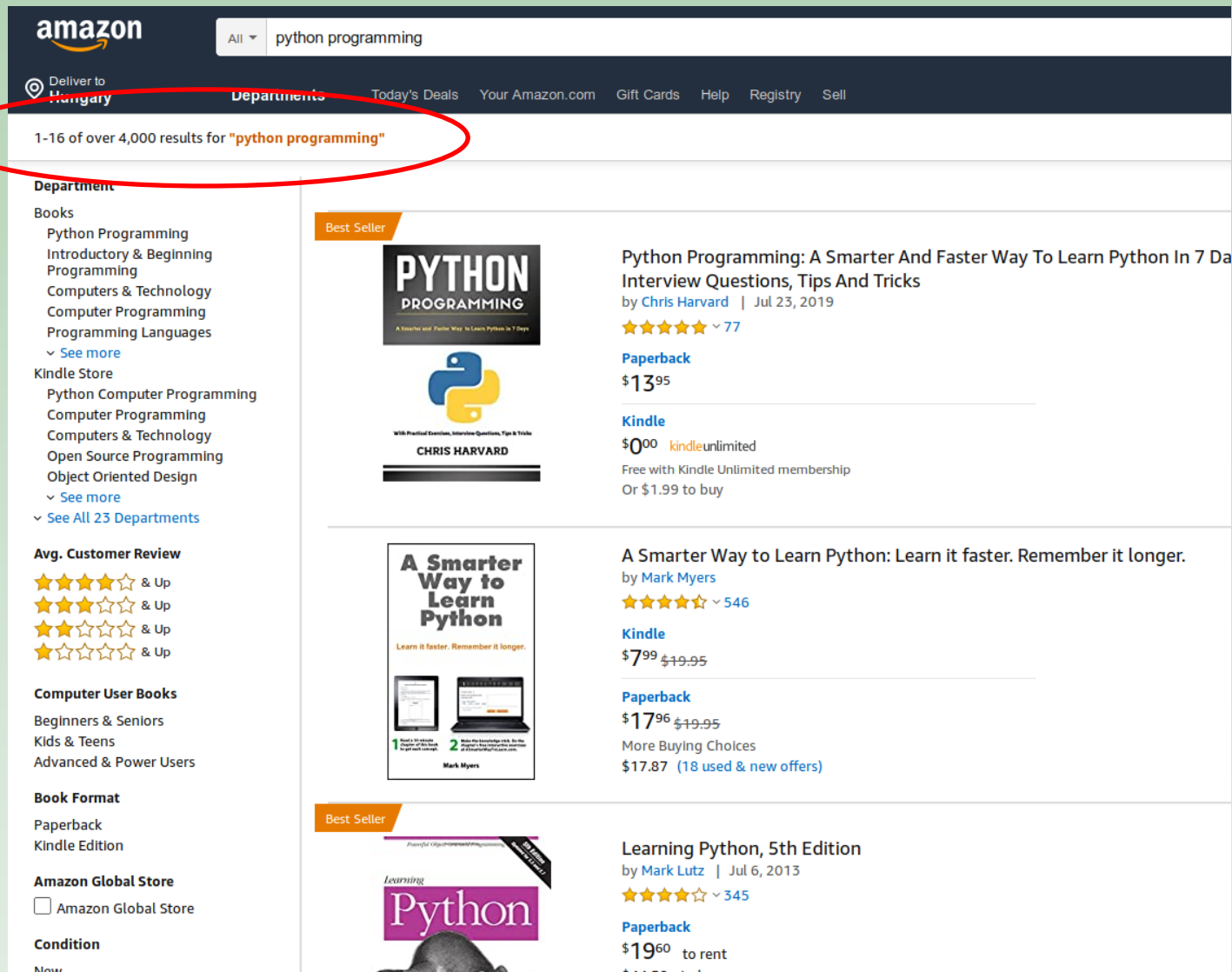
TIOBE index (<http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>)



Aug 2019	Aug 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.028%	-0.85%
2	2		C	15.154%	+0.19%
3	4	▲	Python	10.020%	+3.03%
4	3	▼	C++	6.057%	-1.41%
5	6	▲	C#	3.842%	+0.30%
6	5	▼	Visual Basic .NET	3.695%	-1.07%
7	8	▲	JavaScript	2.258%	-0.15%
8	7	▼	PHP	2.075%	-0.85%
9	14	▲▲	Objective-C	1.690%	+0.33%
10	9	▼	SQL	1.625%	-0.69%
11	15	▲▲	Ruby	1.316%	+0.13%
12	13	▲	MATLAB	1.274%	-0.09%
13	44	▲▲	Groovy	1.225%	+1.04%
14	12	▼	Delphi/Object Pascal	1.194%	-0.18%
15	10	▼	Assembly language	1.114%	-0.30%
16	19	▲	Visual Basic	1.025%	+0.10%
17	17		Go	0.973%	-0.02%
18	11	▼	Swift	0.890%	-0.49%
19	16	▼	Perl	0.860%	-0.31%
20	18	▼	R	0.822%	-0.14%

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

Szakirodalom



The screenshot shows the Amazon Hungary search results for "python programming". The search bar at the top displays "python programming" with a dropdown menu set to "All". Below the search bar, a red circle highlights the text "1-16 of over 4,000 results for 'python programming'". The left sidebar contains filters for Department (Books, Kindle Store), Avg. Customer Review (4 stars & up), Computer User Books (Beginners & Seniors, Kids & Teens, Advanced & Power Users), Book Format (Paperback, Kindle Edition), Amazon Global Store, and Condition (New). The main content area displays three book listings:

- Python Programming: A Smarter And Faster Way To Learn Python In 7 Days Interview Questions, Tips And Tricks** by Chris Harvard | Jul 23, 2019. 4.5 stars (77 reviews). Paperback: \$13.95. Kindle: \$0.00 (kindleunlimited) or \$1.99 to buy.
- A Smarter Way to Learn Python: Learn it faster. Remember it longer.** by Mark Myers. 4.5 stars (546 reviews). Kindle: \$7.99 (was \$19.95). Paperback: \$17.96 (was \$19.95). More Buying Choices: \$17.87 (18 used & new offers).
- Learning Python, 5th Edition** by Mark Lutz | Jul 6, 2013. 4.5 stars (345 reviews). Paperback: \$19.60 to rent or \$44.58 to buy.

(lekérdezés dátuma: 2019. szept. 6.)

Konferenciák

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



PyCon 2019

- előadások: <https://pyvideo.org/events/pycon-us-2019.html> (146 videó)
- <https://www.youtube.com/channel/UCxs2IIVXaEHHA4BtTiWZ2mQ/playlists>
- az előadások között tutorialok is vannak



PyCon 2018

- előadások: <https://pyvideo.org/events/pycon-us-2018.html> (146 videó)
- az előadások között tutorialok is vannak

A <https://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:

```
[11:57:44] ~ $ python3
Python 3.6.5 (default, Apr 1 2018, 05:46:30)
[GCC 7.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

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 nyugodtan kiírhatjuk.

```
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: <https://goo.gl/IELOYy>

```
>>> 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>

<https://goo.gl/uBQPYA>

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][5]...

- [1] <https://arato.inf.unideb.hu/szathmary.laszlo/pmwiki/index.php?n=Py.Feladatok>
- [2] <http://www.pythonchallenge.com/>
- [3] <http://projecteuler.net/> (főleg matematikai jellegű feladatok)
- [4] <https://www.hackerrank.com>
- [5] <https://adventofcode.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