

Óbudai Egyetem
Kandó Kálmán Villamosmérnöki Kar
Python
input, output

Dr. Schuster György

2017. november 13.

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

Csak egy sima soremelés.

```
>>> print
```

```
>>>
```

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

Egyetlen elem kiírása.

```
>>> a=5  
>>> print a
```

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

Egyetlen elem kiírása.

```
>>> a=5
>>> print a
5
>>>
```

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

```
>>> a="Hello"
```

Ami lehet szöveg is.

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

Ami lehet szöveg is.

```
>>> a="Hello"  
>>> print a  
Hello  
>>>
```

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

```
>>> a="Hello"  
>>> b=" world"
```

Ha több elemet kell kiírni.

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

```
>>> a="Hello"  
>>> b=" world"  
>>> print a,b  
Hello world  
>>>
```

Ha több elemet kell kiírni.

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

```
print,
```

Ha nem akarunk sort emelni `print` után.

print, mint kifejezés

A `print` kifejezés segítségével az `stdout`-ra tudunk írni egyszerűen

`print,`

Ha nem akarunk sort emelni `print` után.
De ez nem megy python shell-ben.

A `print`, mint függvény

A `print` létezik függvényként is.

A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

Ezt akarjuk kiírni:

A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

Ezt akarjuk kiírni:

```
Sum=3 a=2 b=1
```

A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

Ezt akarjuk kiírni:

```
Sum=3 a=2 b=1
```

```
Adjunk értékeket!
```

```
  ⋮  
a=2  
b=1  
c=a+b  
  ⋮
```


A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

Ezt akarjuk kiírni:

```
Sum=3 a=2 b=1
```

Adjunk értékeket!

```
  :  
a=2  
b=1  
c=a+b  
  :
```

Normál szöveg összefűzéssel:

```
print ("Sum=", c, "a=", a, "b=", b)
```

A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

Ezt akarjuk kiírni:

```
Sum=3 a=2 b=1
```

Adjunk értékeket!

```
  :  
a=2  
b=1  
c=a+b  
  :
```

Normál szöveg összefűzéssel:

```
print ("Sum=", c, "a=", a, "b=", b)
```

Karakterlánc összefűzéssel:

```
print ("Sum="+str(c)+"a="+str(a)+"b="+str(b))
```

A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

Ezt akarjuk kiírni:

```
Sum=3 a=2 b=1
```

Adjunk értékeket!

```
  :  
a=2  
b=1  
c=a+b  
  :
```

Normál szöveg összefűzéssel:

```
print ("Sum=", c, "a=", a, "b=", b)
```

Karakterlánc összefűzéssel:

```
print ("Sum="+str(c)+"a="+str(a)+"b="+str(b))
```

Formátum sztring és tuple használatával.

A `print`, mint függvény

A `print` létezik függvényként is.
Ekkor formátumozott kiírást is tud.

Ezt akarjuk kiírni:

```
Sum=3 a=2 b=1
```

Adjunk értékeket!

```
  :  
a=2  
b=1  
c=a+b  
  :
```

Normál szöveg összefűzéssel:

```
print ("Sum=", c, "a=", a, "b=", b)
```

Karakterlánc összefűzéssel:

```
print ("Sum="+str(c)+"a="+str(a)+"b="+str(b) )
```

Formátum sztring és tuple használatával.

Pozicionálással és a `format` használatával.

Formátumozott kiiratás

A `print` függvény alkalmas c-szerű formátumozott kiíratásra is.

Formátumozott kiiratás

A `print` függvény alkalmas c-szerű formátumozott kiíratásra is.

Példa:

Formátumozott kiiratás

A `print` függvény alkalmas c-szerű formátumozott kiíratásra is.

Példa:

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```


A printelő kifejezés.

Formátumozott kiiratás

A `print` függvény alkalmas c-szerű formátumozott kiíratásra is.

Példa:

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```


Formátum sztring

A printelő kifejezés.
A formátum sztring.

Formátumozott kiiratás

A `print` függvény alkalmas c-szerű formátumozott kiíratásra is.

Példa:

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```

Adat kupac

A printelő kifejezés.
A formátum sztring.
Az adatok kupaca (tuple).

Formátumozott kiiratás

A `print` függvény alkalmas c-szerű formátumozott kiíratásra is.

Példa:

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```

↑
Sztring moduló operátor

A printelő kifejezés.
A formátum sztring.
Az adatok kupaca (tuple).
A sztring moduló operátor.

Formátum sztring

A formátum sztring tartalmazhat formátum specifikátort és más elemeket.

Formátum sztring

A formátum sztring tartalmazhat formátum specifikátort és más elemeket.


A formátum specifikátor helyére íródik be a megfelelő adat az adat kupacból.

Formátum sztring

A formátum sztring tartalmazhat formátum specifikátort és más elemeket.

A formátum specifikátor helyére íródik be a megfelelő adat az adat kupacból.

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```



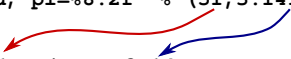
Formátum sztring

A formátum sztring tartalmazhat formátum specifikátort és más elemeket.

A formátum specifikátor helyére íródik be a megfelelő adat az adat kupacból.

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```

a= 51, pi= 3.14



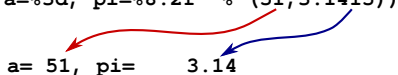
Formátum sztring

A formátum sztring tartalmazhat formátum specifikátort és más elemeket.

A formátum specifikátor helyére íródik be a megfelelő adat az adat kupacból.

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```

```
a= 51, pi=      3.14
```

A diagram with two arrows. A red arrow starts from the number '51' in the tuple '(51, 3.1415)' and points to the '%3d' format specifier in the string 'a=%3d, pi=%8.2f'. A blue arrow starts from the float '3.1415' in the tuple and points to the '%8.2f' format specifier in the string.

Azok az elemek, amelyek nem értelmezhetőek formátum specifikátorként, azok kiíródnak.

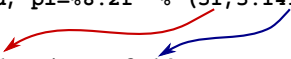
Formátum sztring

A formátum sztring tartalmazhat formátum specifikátort és más elemeket.

A formátum specifikátor helyére íródik be a megfelelő adat az adat kupacból.

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```

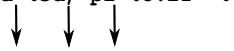
a= 51, pi= 3.14



Azok az elemek, amelyek nem értelmezhetőek formátum specifikátorként, azok kiíródnak.

```
print("a=%3d, pi=%8.2f" % (51, 3.1415))
```

a= 51, pi= 3.14



Formátum specifikátor

A formátum specifikátor szerkezete:

Formátum specifikátor

A formátum specifikátor szerkezete:

```
% [flags] [width] [.prec] type
```

Formátum specifikátor

A formátum specifikátor szerkezete:

```
% [flags] [width] [.prec] type
```

A **%** és a **type** kötelező.

Formátum specifikátor

A formátum specifikátor szerkezete:

```
% [flags] [width] [.prec] type
```

A % és a `type` kötelező.

A többi elem, ami `[]` jelek között van nem kötelező.

Formátum specifikátor

A formátum specifikátor szerkezete:

```
% [flags] [width] [.prec] type
```

A % és a **type** kötelező.

A többi elem, ami [] jelek között van nem kötelező.

De jól jön.

Formátum specifikátor `t`

d, i előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

előjeltelen egész.

Formátum specifikátor `type`

`d, i` előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print("%d" % (5))  
5
```

előjeltelen egész.

Formátum specifikátor `type`

`d, i` előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print ("%d" % (5))  
5
```

```
>>>print ("%d" % (3.1415))  
3
```

előjeltelen egész.

Formátum specifikátor `type`

`d`, `i` előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print("%d" % (5))  
5
```

```
>>>print("%d" % (3.1415))  
3
```

`u`, `x`, `X`, `o` előjeltelen egész.

Formátum specifikátor `t`

`d`, `i` előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print("%d" % (5))  
5
```

```
>>>print("%d" % (3.1415))  
3
```

`u`, `x`, `X`, `o` előjeltelen egész. (De kiírja az előjelet is.)

Formátum specifikátor `type`

`d`, `i` előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print ("%d" % (5))  
5
```

```
>>>print ("%d" % (3.1415))  
3
```

`u`, `x`, `X`, `o` előjeltelen egész. (De kiírja az előjelet is.)

```
>>>print ("%u" % (53))  
53
```

Formátum specifikátor `type`

`d`, `i` előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print ("%d" % (5))  
5
```

```
>>>print ("%d" % (3.1415))  
3
```

`u`, `x`, `X`, `o` előjeltelen egész. (De kiírja az előjelet is.)

```
>>>print ("%u" % (53))  
53
```

```
>>>print ("%x" % (53))  
35
```

Formátum specifikátor `type`

d, i előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print ("%d" % (5))  
5
```

```
>>>print ("%d" % (3.1415))  
3
```

u, x, X, o előjeltelen egész. (De kiírja az előjelet is.)

```
>>>print ("%u" % (53))  
53
```

```
>>>print ("%x" % (53))  
35
```

x esetén a 9-nél nagyobb számjegyeket kisbetűkkel,

Formátum specifikátor `type`

d, i előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print ("%d" % (5))  
5
```

```
>>>print ("%d" % (3.1415))  
3
```

u, x, X, o előjeltelen egész. (De kiírja az előjelet is.)

```
>>>print ("%u" % (53))  
53
```

```
>>>print ("%x" % (53))  
35
```

x esetén a 9-nél nagyobb számjegyeket kisbetűkkel,

X esetén nagybetűkkel írja ki.

Formátum specifikátor `type`

d, i előjeles egész. Ha a kupacban megfelelő helyen tört szám van, akkor annak egészrészét írja ki.

```
>>>print ("%d" % (5))  
5
```

```
>>>print ("%d" % (3.1415))  
3
```

u, x, X, o előjeltelen egész. (De kiírja az előjelet is.)

```
>>>print ("%u" % (53))  
53
```

```
>>>print ("%x" % (53))  
35
```

```
>>>print ("%o" % (53))  
65
```

x esetén a 9-nél nagyobb számjegyeket kisbetűkkel,

x esetén nagybetűkkel írja ki.

Formátum specifikátor `t` `T`

`f`, `F` lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

Formátum specifikátor `type`

`f`, `F` lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

```
>>>print ("%f" % (3.14))  
3.140000
```

Formátum specifikátor `t`

`f`, `F` lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

```
>>>print ("%f" % (3.14))  
3.140000
```

```
>>>print ("%f" % (3))  
3.000000
```

Formátum specifikátor `type`

f, F lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

```
>>>print ("%f" % (3.14))  
3.140000
```

```
>>>print ("%f" % (3))  
3.000000
```

e, E normálformájú lebegőpontos kiíratás.

Formátum specifikátor `type`

f, F lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

```
>>>print ("%f" % (3.14))  
3.140000
```

```
>>>print ("%f" % (3))  
3.000000
```

e, E normálformájú lebegőpontos kiíratás.
e esetén a normál alak elválasztó **e**,

Formátum specifikátor `type`

f, F lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

```
>>>print ("%f" % (3.14))  
3.140000
```

```
>>>print ("%f" % (3))  
3.000000
```

e, E normálformájú lebegőpontos kiíratás.
e esetén a normál alak elválasztó **e**,
E esetén a normál alak elválasztó **E**.

Formátum specifikátor `type`

f, F lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

```
>>>print ("%f" % (3.14))  
3.140000
```

```
>>>print ("%f" % (3))  
3.000000
```

e, E normálformájú lebegőpontos kiíratás.
e esetén a normál alak elválasztó **e**,
E esetén a normál alak elválasztó **E**.

```
>>>print ("%e" % (3.14))  
3.140000e00
```

Formátum specifikátor `type`

f, F lebegőpontos kiíratás. Ha a kiírandó szám tizedesjegyeinek száma kisebb, mint az alappontosság, akkor kiíratást nullákkal egészíti ki. Ez igaz egész számokra is.

```
>>>print ("%f" % (3.14))  
3.140000
```

```
>>>print ("%f" % (3))  
3.000000
```

e, E normálformájú lebegőpontos kiíratás.
e esetén a normál alak elválasztó **e**,
E esetén a normál alak elválasztó **E**.

```
>>>print ("%e" % (3.14))  
3.140000e00
```

```
>>>print ("%E" % (3.14))  
3.140000E00
```

Formátum specifikátor `t` `e`

`g`, `G` a rövidebb lebegőpontos kiiratás. Annyit ír ki amennyit kell és a rövidebb módot választja. A kisbetű - nagybetű, mint `e`, `E`-nél.

Formátum specifikátor `t` `e`

`g`, `G` a rövidebb lebegőpontos kiiratás. Annyit ír ki amennyit kell és a rövidebb módot választja. A kisbetű - nagybetű, mint `e`, `E`-nél.

Rövid kiiratás:

Formátum specifikátor `type`

`g`, `G` a rövidebb lebegőpontos kiiratás. Annyit ír ki amennyit kell és a rövidebb módot választja. A kisbetű - nagybetű, mint `e`, `E`-nél.

Rövid kiiratás:

```
>>>print ("%g" % (3.14))  
3.14
```

Formátum specifikátor `t` `e`

`g`, `G` a rövidebb lebegőpontos kiiratás. Annyit ír ki amennyit kell és a rövidebb módot választja. A kisbetű - nagybetű, mint `e`, `E`-nél.

Rövid kiiratás:

```
>>>print ("%g" % (3.14))  
3.14
```

Rövidebb kiiratás:

Formátum specifikátor `type`

`g`, `G` a rövidebb lebegőpontos kiiratás. Annyit ír ki amennyit kell és a rövidebb módot választja. A kisbetű - nagybetű, mint `e`, `E`-nél.

Rövid kiiratás:

```
>>>print ("%g" % (3.14))  
3.14
```

Rövidebb kiiratás:

```
>>>print ("%g" % (0.0002))  
0.0002
```

Formátum specifikátor `type`

`g`, `G` a rövidebb lebegőpontos kiiratás. Annyit ír ki amennyit kell és a rövidebb módot választja. A kisbetű - nagybetű, mint `e`, `E`-nél.

Rövid kiiratás:

```
>>>print ("%g" % (3.14))  
3.14
```

Rövidebb kiiratás:

```
>>>print ("%g" % (0.0002))  
0.0002
```

```
>>>print ("%g" % (0.00002))  
2e-05
```

Formátum specifikátor `t`

- c karakter kiírása.

Formátum specifikátor `t`

- `c` karakter kiíratása.

```
>>>print ("%c" % (65))
```

```
A
```

Formátum specifikátor `type`

- c** karakter kiírása.

```
>>>print ("%c" % (65))
```

```
A
```

- s** sztring kiírása, illetőleg minden olyan objektumé, amelyet az `str()` függvénnyel konvertáltak.

Formátum specifikátor `type`

- c** karakter kiíratása.

```
>>>print ("%c" % (65))  
A
```

- s** sztring kiíratása, illetőleg minden olyan objektumé, amelyet az `str()` függvénnyel konvertáltak.

```
>>>a="Humpty Dumpty"  
>>>print ("%s" % (a))  
Humpty Dumpty
```

Formátum specifikátor `type`

- c** karakter kiírása.

```
>>>print ("%c" % (65))  
A
```

- s** sztring kiírása, illetőleg minden olyan objektumé, amelyet az `str()` függvénnyel konvertáltak.

```
>>>a="Humpty Dumpty"  
>>>print ("%s" % (a))  
Humpty Dumpty
```

- s** sztring kiírása minden olyan objektumra, amelyet a `repr()` függvénnyel konvertáltak.

Formátum specifikátor `type`

- c** karakter kiíratása.

```
>>>print ("%c" % (65))  
A
```

- s** sztring kiíratása, illetőleg minden olyan objektumé, amelyet az `str()` függvénnyel konvertáltak.

```
>>>a="Humpty Dumpty"  
>>>print ("%s" % (a))  
Humpty Dumpty
```

- s** sztring kiíratása minden olyan objektumra, amelyet a `repr()` függvénnyel konvertáltak.

```
>>>print ("%s" % repr("Humpty Dumpty"))  
'Humpty Dumpty'
```

Formátum specifikátor `width`

A kiiratás minimális méretét határozza meg.

Formátum specifikátor `width`

A kiiratás minimális méretét határozza meg.

Ha a kiiratás mérete rövidebb, mint a `width` paraméter, akkor a kiiratást a hely jobb oldalára ütközteti. A fennmaradó helyeket szóközökkel tölti fel.

Formátum specifikátor `width`

A kiiratás minimális méretét határozza meg.

Ha a kiiratás mérete rövidebb, mint a `width` paraméter, akkor a kiiratást a hely jobb oldalára ütközteti. A fennmaradó helyeket szóközzel tölti fel.

```
>>>print ("%3d" % (5))  
__ 5
```

A `_` egy szóköz.

Formátum specifikátor `width`

A kiiratás minimális méretét határozza meg.

Ha a kiiratás mérete rövidebb, mint a `width` paraméter, akkor a kiiratást a hely jobb oldalára ütközteti. A fennmaradó helyeket szóközzel tölti fel.

```
>>>print ("%3d" % (5))  
_ _ 5
```

A `_` egy szóköz.

Ha a kiiratás mérete hosszabb, mint a `width` paraméter, akkor ezt felülírja.

Formátum specifikátor `width`

A kiiratás minimális méretét határozza meg.

Ha a kiiratás mérete rövidebb, mint a `width` paraméter, akkor a kiiratást a hely jobb oldalára ütközteti. A fennmaradó helyeket szóközzel tölti fel.

```
>>>print ("%3d" % (5))  
_ _ 5
```

A `_` egy szóköz.

Ha a kiiratás mérete hosszabb, mint a `width` paraméter, akkor ezt felülírja.

```
>>>print ("%3d" % (5000))  
5000
```


Formátum specifikátor `width`

A kiiratás minimális méretét határozza meg.

Ha a kiiratás mérete rövidebb, mint a `width` paraméter, akkor a kiiratást a hely jobb oldalára ütközteti. A fennmaradó helyeket szóközzel tölti fel.

```
>>>print ("%3d" % (5))
  5
```

A egy szóköz.

Ha a kiiratás mérete hosszabb, mint a `width` paraméter, akkor ezt felülírja.

```
>>>print ("%3d" % (5000))
5000
```

```
>>>print ("%3f" % (5))
5.000000
```

Formátum specifikátor `width`

A kiiratás minimális méretét határozza meg.

Ha a kiiratás mérete rövidebb, mint a `width` paraméter, akkor a kiiratást a hely jobb oldalára ütközteti. A fennmaradó helyeket szóközzel tölti fel.

```
>>>print ("%3d" % (5))  
_ _ 5
```

A `_` egy szóköz.

Ha a kiiratás mérete hosszabb, mint a `width` paraméter, akkor ezt felülírja.

```
>>>print ("%3d" % (5000))  
5000
```

```
>>>print ("%3f" % (5))  
5.000000
```

Sajnos!

Formátum specifikátor `%flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használatos. A kiíratást balra ütközteti.

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használatos. A kiíratást balra ütközteti.

```
>>>print ("% -3d" % (5))  
5_ _
```

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használatos. A kiíratást balra ütközteti.

```
>>>print ("% -3d" % (5))  
5_ _
```

+ a pozitív számok elé kiírja a `+` jelet.

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használatos. A kiíratást balra ütközteti.

```
>>>print ("% -3d" % (5))  
5_ _
```

+ a pozitív számok elé kiírja a `+` jelet.

```
>>>print ("% +d" % (-5))  
-5
```

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használatos. A kiíratást balra ütközteti.

```
>>>print ("% -3d" % (5))  
5_ _
```

+ a pozitív számok elé kiírja a `+` jelet.

```
>>>print ("% +d" % (-5))  
-5
```

```
>>>print ("% +d" % (5))  
+5
```

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használatos. A kiíratást balra ütközteti.

```
>>>print ("% -3d" % (5))  
5_ _
```

+ a pozitív számok elé kiírja a `+` jelet.

```
>>>print ("% +d" % (-5))  
-5
```

```
>>>print ("% +d" % (5))  
+5
```

_ a pozitív számok elé kiír egy szóközt.

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használatos. A kiíratást balra ütközteti.

```
>>>print ("% -3d" % (5))  
5_ _
```

+ a pozitív számok elé kiírja a `+` jelet.

```
>>>print ("% +d" % (-5))  
-5
```

```
>>>print ("% +d" % (5))  
+5
```

_ a pozitív számok elé kiír egy szóközt.

```
>>>print ("% _d" % (-5))  
-5
```

Formátum specifikátor `flags`

speciális kiíratást ír elő. Hexadecimális esetben kiírja a `0x`-et, vagy `0X`-et, oktális esetben a vezető `0`-t.

```
>>>print ("%#x" % (53))  
0x35
```

```
>>>print ("%#o" % (53))  
065
```

- a `width` paraméterrel együtt használható. A kiíratást balra ütközteti.

```
>>>print ("% -3d" % (5))  
5_ _
```

+ a pozitív számok elé kiírja a `+` jelet.

```
>>>print ("% +d" % (-5))  
-5
```

```
>>>print ("% +d" % (5))  
+5
```

_ a pozitív számok elé kiír egy szóközt.

```
>>>print ("% _d" % (-5))  
-5
```

```
>>>print ("% _d" % (5))  
_5
```

Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Lebegőpontos számok esetén a tizedesjegyek számát adja meg.

Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Lebegőpontos számok esetén a tizedesjegyek számát adja meg.

```
>>>print("%.3f" % (3.1415926535))  
3.142
```


Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Lebegőpontos számok esetén a tizedesjegyek számát adja meg.

```
>>>print("%.3f" % (3.1415926535))  
3.142
```

És még kerekít is.

Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Lebegőpontos számok esetén a tizedesjegyek számát adja meg.

```
>>>print("%.3f" % (3.1415926535))  
3.142
```

És még kerekít is.

Egészek esetén a minimálisan kiírandó számjegyek számát adja meg.

Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Lebegőpontos számok esetén a tizedesjegyek számát adja meg.

```
>>>print("%.3f" % (3.1415926535))  
3.142
```

És még kerekít is.

Egészek esetén a minimálisan kiírandó számjegyek számát adja meg.

```
>>>print("%.5d" % (5))  
00005
```

Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Lebegőpontos számok esetén a tizedesjegyek számát adja meg.

```
>>>print("%.3f" % (3.1415926535))  
3.142
```

És még kerekít is.

Egészek esetén a minimálisan kiírandó számjegyek számát adja meg.

```
>>>print("%.5d" % (5))  
00005
```

Sztringek esetén a maximálisan kiíradó karakterek számát adja meg.

Formátum specifikátor `.prec`

A `.prec` paraméter jelentése típusfüggő.

Lebegőpontos számok esetén a tizedesjegyek számát adja meg.

```
>>>print("%.3f" % (3.1415926535))  
3.142
```

És még kerekít is.

Egészek esetén a minimálisan kiírandó számjegyek számát adja meg.

```
>>>print("%.5d" % (5))  
00005
```

Sztringek esetén a maximálisan kiíradó karakterek számát adja meg.

```
>>>a="Humpty Dumpty"  
>>>print("%.4s" % (a))  
Hump
```

format függvény

A **format** hasonló formátumozott kiíratást tesz lehetővé, mint a formátum sztring, illetve a formátum specifikátorok által vezérelt kiíratás.

format függvény

A `format` hasonló formátumozott kiíratást tesz lehetővé, mint a formátum sztring, illetve a formátum specifikátorok által vezérelt kiíratás.

Itt is van formátum sztring, de a formátum specifikátorok aszociációs tömbökben vannak.

format függvény

A `format` hasonló formátumozott kiíratást tesz lehetővé, mint a formátum sztring, illetve a formátum specifikátorok által vezérelt kiíratás.

Itt is van formátum sztring, de a formátum specifikátorok aszociációs tömbökben vannak.

```
print ("a={ } b={ } c={ }".format ('a', 'b', 'c'))
```

The diagram shows the code `print ("a={ } b={ } c={ }".format ('a', 'b', 'c'))` with several annotations:

- Three purple arrows point to the curly braces in the format string, labeled "formátum hash-ek".
- A red bracket under the format string is labeled "formátum sztring".
- A blue arrow points from the text "pont operátor" to the period character in the code.
- A green bracket under the `format ('a', 'b', 'c')` part is labeled "format függvény paraméterekkel".

format függvény

A **format** hasonló formátumozott kiíratást tesz lehetővé, mint a formátum sztring, illetve a formátum specifikátorok által vezérelt kiíratás.

Itt is van formátum sztring, de a formátum specifikátorok aszociációs tömbökben vannak.

```
print ("a={ } b={ } c={ }".format ('a', 'b', 'c'))
```

The diagram shows the code `print ("a={ } b={ } c={ }".format ('a', 'b', 'c'))` with several annotations:

- Three purple arrows point to the curly braces in the format string, labeled "formátum hash-ek".
- A red bracket under the format string is labeled "formátum sztring".
- A blue arrow points from the text "pont operátor" to the period character in the code.
- A green bracket under the `format ('a', 'b', 'c')` part is labeled "format függvény paraméterekkel".

Amit nem tud formátumként értelmezni, azt kiírja.

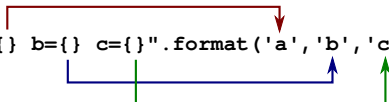
format függvény

A formátum specifikátorban meghatározható a kiírás helye.

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
print("a={ } b={ } c={ }".format('a', 'b', 'c'))
```



format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
>>> print("a={} b={} c={}".format('a', 'b', 'c'))  
a=a b=b c=c
```

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
>>> print("a={} b={} c={}".format('a', 'b', 'c'))  
a=a b=b c=c
```

Nem adtunk meg pozíció értéket.

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
print("a={1} b={2} c={0}".format('a', 'b', 'c'))
```

The diagram illustrates the mapping of arguments to format specifiers in the code `print("a={1} b={2} c={0}".format('a', 'b', 'c'))`. A blue arrow points from the argument `'a'` to the format specifier `{1}`. A green arrow points from the argument `'b'` to the format specifier `{2}`. A red arrow points from the argument `'c'` to the format specifier `{0}`.

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
>>> print ("a={1} b={2} c={0}").format ('a', 'b', 'c')  
a=b b=c c=a
```

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
>>> print ("a={1} b={2} c={0}").format ('a', 'b', 'c'))  
a=b b=c c=a
```

Pozíció értékek működése.

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
print("a={b:} b={c:} c={a:}".format(a='a',b='b',c='c'))
```

The diagram illustrates the argument substitution in the format function. A green arrow points from the argument 'a' to the format specifier '{b:}'. A blue arrow points from the argument 'b' to the format specifier '{c:}'. A red arrow points from the argument 'c' to the format specifier '{a:}'.

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
>>> print("a={b:} b={c:} c={a:}").format(a='a', b='b', c='c')  
a=b b=c c=a
```

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
>>> print("a={b:} b={c:} c={a:}").format(a='a', b='b', c='c')  
a=b b=c c=a
```

Név szerinti pozicionálás.

format függvény

A formátum specifikátorban meghatározható a kiírás helye.

```
>>> print("a={b:} b={c:} c={a:}".format(a='a', b='b', c='c'))  
a=b b=c c=a
```

Név szerinti pozicionálás.

Ennél azonban sokkal többet tud.

format függvény

Formátumozás a hash segítségével.

format függvény

Formátumozás a hash segítségével.
Hasonlóan, mint a klasszikus formátumnál.

format függvény

Formátumozás a hash segítségével.

Hasonlóan, mint a klasszikus formátumnál.

Típusok:

d, i	előjeles egész	f, e, E, g, G	lebegőpontos,
u	előjeltelen egész, decimális,	s	sztring,
x, X	előjeltelen egész, hexa,	c	karakter
o	előjeltelen egész, oktális	r	repr konvertált objektum

format függvény

Formátumozás a hash segítségével.
Hasonlóan, mint a klasszikus formátumnál.

Típusok:

d, i	előjeles egész	f, e, E, g, G	lebegőpontos,
u	előjeltelen egész, decimális,	s	sztring,
x, X	előjeltelen egész, hexa,	c	karakter
o	előjeltelen egész, oktális	r	repr konvertált objektum

Példa 1.a:

```
>>> print("a={0:d}".format(5000000))  
a=5000000
```


format függvény

Formátumozás a hash segítségével.
Hasonlóan, mint a klasszikus formátumnál.

Típusok:

d, i	előjeles egész	f, e, E, g, G	lebegőpontos,
u	előjeltelen egész, decimális,	s	sztring,
x, X	előjeltelen egész, hexa,	c	karakter
o	előjeltelen egész, oktális	r	repr konvertált objektum

Példa 1.b:

```
>>> print ("a={:d}".format(5000000))  
a=5000000
```

format függvény

Formátumozás a hash segítségével.
Hasonlóan, mint a klasszikus formátumnál.

Típusok:

d, i	előjeles egész	f, e, E, g, G	lebegőpontos,
u	előjeltelen egész, decimális,	s	sztring,
x, X	előjeltelen egész, hexa,	c	karakter
o	előjeltelen egész, oktális	r	repr konvertált objektum

Példa 2:

```
>>> print("a={:f}".format(5000000))  
a=5000000.000000
```

format függvény

Formátumozás a hash segítségével.
Hasonlóan, mint a klasszikus formátumnál.

Típusok:

d, i	előjeles egész	f, e, E, g, G	lebegőpontos,
u	előjeltelen egész, decimális,	s	sztring,
x, X	előjeltelen egész, hexa,	c	karakter
o	előjeltelen egész, oktális	r	repr konvertált objektum

Példa 3:

```
>>> print ("a={:g}".format(5000000))  
a=5e+6
```

format függvény

Formátumozás a hash segítségével.
Hasonlóan, mint a klasszikus formátumnál.

Típusok:

d, i	előjeles egész	f, e, E, g, G	lebegőpontos,
u	előjeltelen egész, decimális,	s	sztring,
x, X	előjeltelen egész, hexa,	c	karakter
o	előjeltelen egész, oktális	r	repr konvertált objektum

Példa 4:

```
>>> print ("a={:c}".format(65))  
a=A
```

format függvény

Formátumozás a hash segítségével.
Hasonlóan, mint a klasszikus formátumnál.

Típusok:

d, i	előjeles egész	f, e, E, g, G	lebegőpontos,
u	előjeltelen egész, decimális,	s	sztring,
x, X	előjeltelen egész, hexa,	c	karakter
o	előjeltelen egész, oktális	r	repr konvertált objektum

Példa 5:

```
>>> print("a={:s}".format("Hello world"))  
a=Hello world
```

format függvény további paraméterek

width paraméter.

format függvény további paraméterek

width paraméter. Mint a klasszikus formánál.

format függvény további paraméterek

width paraméter. Mint a klasszikus formánál.

Példa 1:

```
>>> print ("a={:3d}".format(5))  
a=  5
```


format függvény további paraméterek

width paraméter. Mint a klasszikus formánál.

Példa 2:

```
>>> print ("a={:3d}".format(5000))  
a=5000
```

stb.

format függvény további paraméterek

width paraméter. Mint a klasszikus formánál.

Példa 2:

```
>>> print ("a={:3d}".format(5000))  
a=5000
```

stb.

.prec paraméter.

format függvény további paraméterek

width paraméter. Mint a klasszikus formánál.

Példa 2:

```
>>> print ("a={:3d}".format(5000))  
a=5000
```

stb.

.prec paraméter. Mint a klasszikus formánál, **egészeknél nem megy.**

format függvény további paraméterek

width paraméter. Mint a klasszikus formánál.

Példa 2:

```
>>> print ("a={:3d}".format(5000))  
a=5000
```

stb.

.prec paraméter. Mint a klasszikus formánál, **egészekenél nem megy.**

Példa 1:

```
>>> print ("a={:.2f}".format(3.1415926535))  
a=3.14
```

format függvény további paraméterek

width paraméter. Mint a klasszikus formánál.

Példa 2:

```
>>> print ("a={:3d}".format(5000))  
a=5000
```

stb.

.prec paraméter. Mint a klasszikus formánál, **egészekenél nem megy.**

Példa 2:

```
>>> print ("a={:.5s}".format("Hello world"))  
a=Hello
```

format függvény további paraméterek

flag-ek.

format függvény további paraméterek

flag-ek. A `%1ag` paraméterek a pozícionáló paraméterek kivételével működnek.

format függvény további paraméterek

flag-ek. A `flag` paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 1:

```
>>> print ("a={:+d}".format (5) )  
a=+5
```


format függvény további paraméterek

flag-ek. A `flag` paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 2:

```
>>> print ("a={:_d}".format(5))  
a=_5
```

format függvény további paraméterek

flag-ek. A `flag` paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 3:

```
>>> print ("a={: #x}".format (5))  
a=0x5
```

stb.

format függvény további paraméterek

flag-ek. A `flag` paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 4:

```
>>> print("a={:05d}".format(5))
```

```
a=00005
```

Ez egy kicsit más.

format függvény további paraméterek

flag-ek. A `flag` paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 4:

```
>>> print ("a={:05d}".format(5))  
a=00005
```

Ez egy kicsit más.

Pozícionálás.

format függvény további paraméterek

flag-ek. A **flag** paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 4:

```
>>> print("a={:05d}".format(5))  
a=00005
```

Ez egy kicsit más.

Pozícionálás.

Példa 1: (alapállapot sztringnél)

```
>>> print("a={:10s} world".format("Hello"))  
a=Hello      world
```

sztringnél balra ütköztet.

format függvény további paraméterek

flag-ek. A **flag** paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 4:

```
>>> print("a={:05d}".format(5))  
a=00005
```

Ez egy kicsit más.

Pozícionálás.

Példa 2: balra ütköztet, (ami alapállapot sztringnél)

```
>>> print("a={:<s} world".format("Hello"))  
a=Hello_ _ _ _ _ world
```

format függvény további paraméterek

flag-ek. A **flag** paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 4:

```
>>> print("a={:05d}".format(5))  
a=00005
```

Ez egy kicsit más.

Pozícionálás.

Példa 3: jobbra ütköztet,

```
>>> print("a={:>s} world".format("Hello"))  
a=_____Hello world
```

format függvény további paraméterek

flag-ek. A **flag** paraméterek a pozícionáló paraméterek kivételével működnek.

Példa 4:

```
>>> print("a={:05d}".format(5))  
a=00005
```

Ez egy kicsit más.

Pozícionálás.

Példa 3: középre helyez,

```
>>> print("a={:^s} world".format("Hello"))  
a=  Hello  world
```


Python 3 bővítések

A python3 néhány új lehetőséget biztosít:

Python 3 bővítések

A python3 néhány új lehetőséget biztosít:

sep függvény.

Python 3 bővítések

A python3 néhány új lehetőséget biztosít:

`sep` függvény.

Példa:

```
>>> print('a', 'b', 'c', sep="#")  
a#b#c
```

Python 3 bővítések

A python3 néhány új lehetőséget biztosít:

`sep` függvény.

Példa:

```
>>> print('a', 'b', 'c', sep="#")  
a#b#c
```

Példa:

```
>>> print('a', 'b', 'c', "#")  
a b c#>>>
```

Nem volt soremelés.

Sztring pozícionálás

Készítsünk egy sztringet!

Sztring pozícionálás

Készítsünk egy sztringet!

```
>>> s="Python"
```

Sztring pozícionálás

Készítsünk egy sztringet!

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

Sztring pozícionálás

Készítsünk egy sztringet!

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10)
```

```
'Python_     '
```

Sztring pozícionálás

Készítsünk egy sztringet!

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10)
```

```
'Python    '
```

Nyomtassuk jobbra ütköztetetten!

Sztring pozícionálás

Készítsünk egy sztringet!

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10)
```

```
'Python____'
```

Nyomtassuk jobbra ütköztetetten!

```
>>> s=s.rjust(10)
```

```
'____Python'
```

Sztring pozícionálás

Készítsünk egy sztringet!

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10)
```

```
'Python____'
```

Nyomtassuk jobbra ütköztetetten!

```
>>> s=s.rjust(10)
```

```
'____Python'
```

Nyomtassuk középre ütköztetetten!

Sztring pozícionálás

Készítsünk egy sztringet!

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10)
'Python____'
```

Nyomtassuk jobbra ütköztetetten!

```
>>> s=s.rjust(10)
'____Python'
```

Nyomtassuk középre ütköztetetten!

```
>>> s=s.center(10)
'__Python__'
```

Sztring pozícionálás

Kitöltő karakter is megadható.

Sztring pozícionálás

Kitöltő karakter is megadható.

```
>>> s="Python"
```

Sztring pozícionálás

Kitöltő karakter is megadható.

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

Sztring pozícionálás

Kitöltő karakter is megadható.

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10, "*")
```

```
'Python****'
```

Sztring pozícionálás

Kitöltő karakter is megadható.

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10, "*")
```

```
'Python****'
```

Nyomtassuk jobbra ütköztetetten!

Sztring pozícionálás

Kitöltő karakter is megadható.

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10, "*")  
'Python****'
```

Nyomtassuk jobbra ütköztetetten!

```
>>> s=s.rjust(10, "*")  
'****Python'
```

Sztring pozícionálás

Kitöltő karakter is megadható.

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10, "*")  
'Python****'
```

Nyomtassuk jobbra ütköztetetten!

```
>>> s=s.rjust(10, "*")  
'****Python'
```

Nyomtassuk középre ütköztetetten!

Sztring pozícionálás

Kitöltő karakter is megadható.

```
>>> s="Python"
```

Nyomtassuk balra ütköztetetten!

```
>>> s=s.ljust(10, "*")  
'Python****'
```

Nyomtassuk jobbra ütköztetetten!

```
>>> s=s.rjust(10, "*")  
'****Python'
```

Nyomtassuk középre ütköztetetten!

```
>>> s=s.center(10, "*")  
'**Python**'
```


Standard input

A `python2..` és a `python3..` eltér egymástól.

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```

Ezt kiírja a képernyőre

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

```
Enter the value:
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

```
Enter the value:H
```


Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

```
Enter the value:He
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

```
Enter the value:Hel
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")  
Enter the value:Hell
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

```
Enter the value:Hello
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

```
Enter the value:Hello
```

```
>>> print(a)
```

Standard input

A `python2..` és a `python3..` eltér egymástól.

A `python2..` a `raw_input` függvényt használja.

A `python3..` az `input` függvényt használja.

A két függvény használata lényegében nem különbözik.

```
a=input("Enter the value:")
```



Ezt kiírja a képernyőre

Ebbe a változóba olvas be

```
>>> a=input("Enter the value:")
```

```
Enter the value:Hello
```

```
>>> print(a)
```

```
Hello
```

Standard input

A probléma az, hogy a beolvasott érték sztring.

Standard input

A probléma az, hogy a beolvasott érték sztring.

Konvertálni kell!

Standard input

A probléma az, hogy a beolvasott érték sztring.

Konvertálni kell!

Egészé:

```
>>> a=input("Enter the value:")  
5  
>>> b=int(a)  
>>> c=b/2  
>>> print(c)  
2
```

Standard input

A probléma az, hogy a beolvasott érték sztring.

Konvertálni kell!

Egészszé:

```
>>> a=input("Enter the value:")
5
>>> b=int(a)
>>> c=b/2
>>> print(c)
2
```

Lebegőpontoság:

```
>>> a=input("Enter the value:")
5
>>> b=float(a)
>>> c=b/2
>>> print(c)
2.5
```

Standard input

A probléma az, hogy a beolvasott érték sztring.

Konvertálni kell!

Egészszé:

```
>>> a=input("Enter the value:")
5
>>> b=int(a)
>>> c=b/2
>>> print(c)
2
```

Lebegőpontosá:

```
>>> a=input("Enter the value:")
5
>>> b=float(a)
>>> c=b/2
>>> print(c)
2.5
```

Ettől kezdve használhatjuk kedvünkre.

