APL!TT

Python!

Lekcja 1 – podstawy języka, typy danych

Plan

- Python – z czym jeść?
- Python jako język OO
- Python 2 czy 3
- Typy danych: liczby – int, float, bool (!)
- Typy danych: sekwencje – stringi, krotki, listy
- Typy danych: słowniki (tablice asocjacyjne)

Python – z czym jeść

- Dokumentacja (RTFM)
- Interpreter Python – przerabia napisany program na kod maszynowy.
- pip – manager pakietów/modułów Python
- Implementacje na każde podniebienie: CPython, IronPython, Jython, PyPy

Python jako język OO

- OO = *object-oriented*
- *Obiekt = pola + metody*
- *Wszystko w Pythonie jest obiektem. Wszystko.*

- **Cechy typowe dla OO:**
  - Dziedziczenie
  - Polimorfizm
  - Abstrakcja
  - Hermetyzacja

  W skrócie: str = ‘ ‘ to ZŁY pomysł

Python 2 czy 3

- Python 3 !
- Python 2 jest nierozwijany – brak nowych rozwiązań, bugfixy do 2020 (PEP 373)
- Python 2 wciąż popularny w produkcji – stare systemy, stare repozytoria, dużo bibliotek

- Python 2

```
>>> print "Hello World!"
Hello World!
>>> type(input('gib num: '))
gib num: 12
<type 'int'>
>>> 1/2
0
>>> 1//2
0
```

- Python 3

```
>>> print("Hello World!")
Hello World!
>>> type(input('gib num: '))
gib num: 12
<class 'str'>
>>> 1/2
0.5
>>> 1//2
0
```

https://wiki.python.org/moin/Python3.0

http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html

Typy danych: liczby – int, float, bool (!)

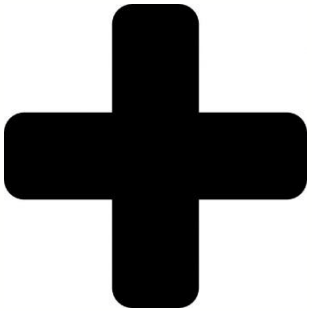
- int – liczby całkowite
- float – liczby zmiennoprzecinkowe (ułamki!)
- bool – typ logiczny True/False

+ dodawanie
- odejmowanie
* mnożenie
** potęgowanie
/ dzielenie
// dzielenie z podłogą (serio)
% modulo

Typy danych: liczby – int, float, bool (!)

+ dodawanie

```
>>> 1 + 1
2
>>> 0.5 + 0.5
1.0
>>> True + True
2
```



Typy danych: liczby – int, float, bool (!)

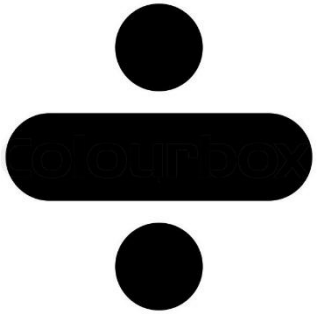
- odejmowanie

```
>>> 2 - 1
1
>>> 1 - 0.5
0.5
>>> False - True
-1
```

Typy danych: liczby – int, float, bool (!)

```
// dzielenie z podłogą

>>> 1 // 2
0
>>> 6 // 3
2
>>> 1.5 // 3
0.0
>>> 2.5 // 3
0.0
```



Typy danych: liczby – int, float, bool (!)

% modulo

```
>>> 2 % 4
2
>>> 4 % 2
0
>>> 2 % 0.5
0.0
>>> 2 % 0.4
0.3999999999999999
```

Typy danych: sekwencje – stringi, krotki, listy

- stringi – ciągi znaków
- listy – uporządkowany zestaw danych (zmienne)
- krotki – uporządkowany zestaw danych (niezmienne)

Listy mogą być edytowane po utworzeniu, krotki nie.

Typy danych: sekwencje – stringi, krotki, listy

- String

```
>>> "hello"+"world"
'helloworld'
>>> "hello"*3
'hellohellohello'
>>> "hello"[0]
'h'
>>> "hello"[-1]
'o'
>>> "hello"[1:4]
'ell'
>>> len("hello")
5
```

- List

To samo, plus:

```
del list[index]
list.append(obj)
list.count(obj)
list.extend(seq)
list.index(obj)
list.insert(index, obj)
list.pop(index) <- usuwa i zwraca
list.remove(obj)
list.reverse()
list.sort([func])
```

Typy danych: sekwencje – stringi, krotki, listy

Krotki – prawie jak listy, ale po utworzeniu nie mogą być zmienione – stąd brak metod

Dostęp do danych analogicznie do listy i stringów.

Typowe operacje:

cmp(tuple1, tuple2)

len(tuple)

max(tuple)

min(tuple)

tuple(seq)

Typy danych: słowniki (aka tablice asocjacyjne)

- Dane w postaci klucz: wartość
- Klucze niekoniecznie w kolejności alfabetycznej, bo haszowanie
- Klucze muszą być niezmienne (immutable), bo haszowanie

Typy danych: słowniki (aka tablice asocjacyjne)

```
>>> dict = {'key':'value', 'key1':'value1'}
>>> dict['key']
'value‘
>>> del dict['key']
>>> dict.clear()
```

+inne, nudne funkcje

Python 2 czy 3 cd.

- **Python 2**

```
>>> dict = {}
>>> dict.iteritems()
<dictionary-itemiterator object at 0x027D9300>
>>> dict.iterkeys()
<dictionary-keyiterator object at 0x02961090>
>>> dict.itervalues()
<dictionary-valueiterator object at 0x027D9300>
```

- **Python 3**

```
>>> dict = {}
>>> dict.items()
dict_items(list)
>>> dict.keys()
dict_keys(list)
>>> dict.values()
dict_values(list)
```

Metody z wersji 3 zostały przeniesione do 2, więc śmiało można używać ;)
Uwaga: w starszych wersjach może nie działać!

https://wiki.python.org/moin/Python3.0

http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html

Bonus: mutable vs immutable

- mutable

list
dict

- immutable

int
float
tuple

A string?

https://wiki.python.org/moin/Python3.0
http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html

Dziękuję za uwagę!