Atrybuty

Atrybut - „nazwa przywiązana do obiektu”.
(nazwy nazywają obiekty, w szczególności atrybuty nazywają obiekty „przywiązane” do danego obiektu)

Składnia dostępu do atrybutu attribute obiektu obj: . (kropka):

obj.attribute

Przykład: moduły (obiekty) i ich atrybuty (np. funkcje, stałe):

```
import math

y = math.sin(math.pi / 2)
```

Przykład: liczby zespolone:

```
z = 2 + 1j      # liczba zespolona 2 + i
print(z.real) # czesc rzeczywista
print(z.imag) # czesc urojona
```

Metody

Metoda - atrybut, funkcja przywiązana do obiektu.

O takiej funkcji można myśleć, że jej ukrytym parametrem jest obiekt, do którego jest przywiązana.

Przykłady:

```
z1 = 2 + 1j              # liczba zespolona
z2 = z1.conjugate()      # sprzezenie z1, bez parametrow
# z2 == 2 - 1j
```

```
lst = [1, 2, 3]
lst.append(4)    # doklada 1 na koniec listy lst
# lst == [1, 2, 3, 4]
```

Operacje na listach

Atrybuty (w tym metody) obiektu są w znacznej części zdeterminowane przez typ tego obiektu.

Na wykładzie: niektóre metody list.

- append, extend, insert
- pop, remove, clear
- index, count
- sort, reverse
- copy

Oprócz tego, inne operacje na listach (nie przez metody):

- del, len
- min, max, sum
- sorted, reversed
- listy składane*
- (extended) slicing

* - materiały do Listy 5

Metody

Techniczna strona metod: typ obiektu też jest obiektem i posiada atrybuty - funkcje odpowiadające metodom obiektów tego typu.

Przykład: typ list ma funkcję append.

```
lst = [1, 2, 3]
list.append(lst, 0)  # dwa parametry: obiekt typu list,
                     # i co do niego dolozyc
# lst == [1, 2, 3, 0]
```

Gdy lst jest obiektem typu list, wtedy wywołanie metody

```
lst.append(x)
```

tłumaczy się na:

```
list.append(lst, x)
```