Wprowadzenie do języka Python - wykład 4

Funkcje

W definicji funkcji lub procedury występują: zawsze dokładnie jedno słowo kluczowe `def`, dowolnie wiele argumentów podanych w nawiasie i oddzielonych przecinkami, dokładnie jeden ogranicznik `:` kończący instrukcję `def`, słowo kluczowe `return`.

```
def jakasNazwa(arg1, arg2):
    """komentarz docstring"""
    instrukcja(e) do wykonania
    return jakasWartosc
```

Jeżeli funkcja ma być używana bez przekazywania jej argumentów, to nawias w nagłówku musi pozostać pusty. Wartość wynikowa funkcji jest deklarowana za pomocą instrukcji `return`. Instrukcja `return` powoduje natychmiastowe zakończenie wykonywania funkcji, wynikiem może być wszystko: nazwa zmiennej lub coś "bardziej skomplikowane". Instrukcja `return` bez argumentu jest równoważna `return None`.

Funkcja ma jeden argument i “zwraca”.

```
def kwadrat(a):
    return a ** 2


print(kwadrat(4))
```

```
## 16
```

Funkcja ma jeden argument i “nie zwraca”.

```
def kwadrat(a):
    print(a ** 2)


kwadrat(4)
```

```
## 16
```

Funkcja ma dwa argumenty i “zwraca”

```
def dodawanie(a, b):
    return a + b


print(dodawanie(5, 6))
```

```
## 11
```

Funkcja ma jeden argument i “zwraca dwie wartości”

```
def nazwa(a):
    return a, a + 1


print(nazwa(5))
```

```
## (5, 6)
```

Funkcja nie ma argumentu i “nie zwraca”

```
def nazwa():
    print("abc")


nazwa()
```

```
## abc
```

return kończy funkcję

```
def nazwa():
    return 3
    print("abc")


a = 4
print(a)
```

```
## 4
```

Standardowo funkcja wykonuje kopię lokalną argumentów (dla typów “niezmiennych”).

```
def funkcja(a):
    a = 5
    print("w środku:", a)


a = 10
print("Przed:", a)
funkcja(a)
print("Po:", a)
```

```
def printme(str):
    """Funkcja wyświetlająca string"""
    print(str)
    return


printme("abc")
```

```
## abc
```

```
print(printme.__doc__)
```

```
## Funkcja wyświetlająca string
```

Przekazywanie przez referencję

```
def changeme(lista):
    print("Przed zmianą: ", lista)
    lista[2] = 50
    print("Po zmianie: ", lista)
    return


mylist = [10, 20, 30]
changeme(mylist)
```

```
## Przed zmianą:  [10, 20, 30]
## Po zmianie:  [10, 20, 50]
```

```
print("Poza funkcją: ", mylist)
```

```
## Poza funkcją:  [10, 20, 50]
```

```
def changeme(lista):
    lista = [2, 3, 4]
    print("Wewnątrz funkcji: ", lista)
    return


lista = [10, 20, 30]
changeme(lista)
```

```
## Wewnątrz funkcji:  [2, 3, 4]
```

```
print("Poza funkcją: ", lista)
```

```
## Poza funkcją:  [10, 20, 30]
```

```
def changeme():
    global lista
    lista = [2, 3, 4]
    print("Wewnątrz funkcji: ", lista)
    return


changeme()
```

```
## Wewnątrz funkcji:  [2, 3, 4]
```

```
print("Poza funkcją: ", lista)
```

```
## Poza funkcją:  [2, 3, 4]
```

Obowiązkowy argument

```
def printme(str):
    print(str)
    return


printme()
```

```
## TypeError: printme() missing 1 required
positional argument: 'str'
```

Keyword argument

```
def kwadrat(a):
    return a*a


print(kwadrat(a=4))
```

```
## 16
```

Domyślny argument

```
def sumsub(a, b, c=0, d=0):
    return a - b + c - d


print(sumsub(12, 4))
```

```
## 8
```

```
print(sumsub(3, 4, 5, 7))
```

```
## -3
```

```
def srednia(first, *values):
    return (first + sum(values)) / (1 + len(values))


print(srednia(2, 3, 4, 6))
```

```
## 3.75
```

```
print(srednia(45))
```

```
## 45.0
```

```
def f(**kwargs):
    print(kwargs)


f()
```

```
## {}
```

```
f(pl="Polish", en="English")
```

```
## {'pl': 'Polish', 'en': 'English'}
```

Funkcje matematyczne

Link do dokumentacji https://docs.python.org/3/library/math.html

```
import math


a=0
b=math.sin(2*math.pi)
print(b)
```

```
## -2.4492935982947064e-16
```

```
print(math.isclose(a,b, rel_tol=1e-09, abs_tol=1e-09))
```

```
## True
```