Funkcje w Pythonie

Funkcje to części programu wielokrotnego użytku. Pozwalają nam nadać nazwę blokowi wyrażeń, a następnie uruchamiać ten blok używając tej nazwy gdziekolwiek w programie, dowolną ilość razy.

Funkcje są prawdopodobnie *najważniejszą* częścią każdego poważnego programu (w każdym języku programowania).

Funkcje definiuje się używając słowa `def`. Po nim następuje nazwa *identyfikująca* funkcję, następnie para nawiasów, które mogą zawierać kilka nazw zmiennych, a na końcu dwukropek. Poniżej zaczyna się blok wyrażeń, które są częścią tej funkcji.

Python stosuje pojęcie funkcji by grupować fragmenty kodu, a następie wywoływać je z dowolnych miejsc w programie. Aby utworzyć nazwaną funkcję z własnym kodem, stosuje się słowo **def,** które jest skrótem słowa „definiuj”.

Przyglądając się kodowi:

```
def hello():
    print "Hello World!"
    return
```

zauważamy jedynie 2 nowe słowa: **def** i **return**.

`def` – służy właśnie do definiowania funkcji w Pythonie.

`return` – służy do zakończenia wykonywania kodu zawartego w ciele funkcji i pozwala na zwrot dowolnej wartości przy czym tylko 1 wartość można zwrócić jako rezultat wykonania funkcji.

W pytonie, jeśli żadna wartość nie jest zwrócona to domyślnie przyjmowana jest wartość None.

Przykład 1.

```
def powiedzAhoj():
    print 'Ahoj, przygodo!' # Blok należący do funkcji.
# Koniec funkcji.

powiedzAhoj() # Wywołanie funkcji.
powiedzAhoj() # Ponowne wywołanie funkcji.
```

Rezultat:

```
Ahoj, przygodo!
Ahoj, przygodo!
```

Przykład 2. Napiszemy funkcję o nazwie „wypiszMax”, która szuka w dwóch podanych parametrach szuka wartości maksymalnej i ją wypisuje.

```
def wypiszMax(a, b):
    if a > b:
        print a, 'to maksimum'
```

```
    elif a == b:
        print a, 'jest równe', b
    else:
        print b, 'to maksimum'
```

```
1 Sposób wywołania:
wypiszMax(3, 4) # Wartości są wprowadzone bezpośrednio.

2 Sposób wywołania:

x = 5
y = 7

wypiszMax(x, y) # Zmienne stają się argumentami.
```

Rezultat:

```
4 to maksimum
7 to maksimum
```

Przykład 3.

```
def powiedz(wiadomosc, ile = 1):
    print wiadomosc * ile

powiedz('Ahoj')
powiedz('Przygodo!', 5)
```

Rezultat:

```
Ahoj
Przygodo!Przygodo!Przygodo!Przygodo!Przygodo!
```

Wyrażenie return

Wyrażenia `return` używamy do wyjścia z funkcji. Możemy opcjonalnie zwrócić w tym momencie jakąś wartość.
Przykład:

```
def maximum(x, y):
    if x > y:
        return x
    else:
        return y

print maximum(2, 3)
```

Rezultat:

```
3
```

1. #funkcja dodaje dwie liczby do siebie i zwraca ich sumę

```
def suma(a,b):
    return a+b
x=input("podaj 1 liczbe: ")
y=input("podaj 2 liczbe: ")
print "suma liczb wynosi = ",suma(x,y)
```

Efekt:

```
IDLE 2.6.5
>>> ================================ RESTART =====
>>>
podaj 1 liczbe: 1
podaj 2 liczbe: 2
suma liczb wynosi =  3
>>>
```

Wartości przekazywane do funkcji noszą nazwę parametrów. W momencie wywoływania funkcji parametry te mogą być referencjami do danych lub wartościami statycznymi, na przykład liczbami albo ciągami znaków. Niezależnie od sytuacji dane te znajdują się w lokalnym zasięgu funkcji bez potrzeby dostępu do zasięgu globalnego. Powyższa funkcja **suma** posiada dwa parametry: dwie liczby, które zostaną zsumowane. Parametry określa się w nawiasach okrągłych tuż za nazwą funkcji. Wartości przekazanych argumentów dostępne będą we wnętrzu funkcji pod nazwami wskazanymi w nawiasach: **a** i **b.** W momencie wywoływania funkcji przekazuje się do niej argumenty, których wartości mają zostać wystawione do parametrów dostępnych w ciele funkcji. Dokładnie ta sama zasada obowiązuje dla funkcji **suma**, jak i dla wszystkich innych funkcji.

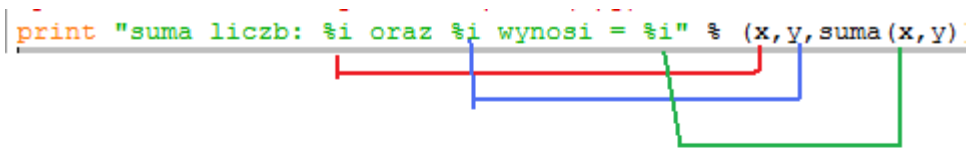
1. Niech funkcja wyświetla rezultat w formie:

```
>>> ================================ RESTART =====
>>>
podaj 1 liczbe: 1
podaj 2 liczbe: 2
suma liczb: 1 oraz 2 wynosi = 3
>>>
```

W tym celu należy dopracować linię z wyświetlaniem wyników i wywołaniem funkcji `suma(a,b)`. Kod powinien wyglądać następująco:

Funkcja dodaje dwie liczby do siebie i zwraca ich sumę:

```
def suma(a,b):
    return a+b
x=input("podaj 1 liczbe: ")
y=input("podaj 2 liczbe: ")
#print "suma liczb wynosi = ",suma(x,y)
print "suma liczb: %i oraz %i wynosi = %i" % (x,y,suma(x,y))
```



%i — podany argument będzie wyświetlony w formiacie: liczby całkowitej (i)

Napiszemy program, a więc funkcję i program ją wywołujący, gdzie zadaniem funkcji o nazwie „parzysta” będzie sprawdzenie, czy liczba wprowadzona przez użytkownika jest parzysta czy nie i zwrócenie jako rezultat tekstu „parzysta” bądź „nieparzysta” w zależności od tego jaka dana liczba faktycznie jest.

```
def parzysta(a):
    if ((a%2)==0):
        return "parzysta"
    else:
        return "nieparzysta"
x=input("podaj 1 liczbe: ")
print "podana liczba jest ",parzysta(x)
```

Rezultat będzie taki:

```
podaj 1 liczbe: 4
podana liczba jest  parzysta
podaj 1 liczbe: 5
podana liczba jest  nieparzysta
```

Teraz napiszemy funkcję o nazwie „*kobieta*”, której przekażemy jako argument znak wprowadzony przez użytkownika z klawiatury. Znak powinien odpowiadać płci użytkownika. Otrzymuje on od programu pytanie czy jest kobietą i jeśli jest powinien odpowiedzieć znakiem „K” bądź „K”. Zadaniem funkcji o nazwie „kobieta” ma być właśnie sprawdzenie czy znak będący odpowiedzią użytkownika a więc i argumentem wywołania funkcji kobieta(a) jest równy znakowi „K” albo „k”. Jeśli tak to funkcji zwróci rezultat „t”. W przeciwnym przypadku zwróci „n”. A więc jeśli ktoś odpowie na pytanie znakiem np. „x” to i tak funkcja zwróci rezultat „n” odpowiadający przypadkowi, że to nie jest kobieta.
Jak widzimy funkcja jest wywołana w instrukcji warunkowej IF..ELSE.
Jeśli funkcja zwróci jako rezultat „t” to na ekranie pojawi się komunikat „Jesteś kobieta”. W przeciwnym przypadku pojawi się komunikat „Jesteś mężczyzną”.

```
def kobieta(a):
    if ((a=='K') or (a=='k')):
        return "t"
    else:
        return "n"
x=raw_input("Czy jestes kobieta ? [K/M]: ")
if kobieta(x)=="t":
    print "Jestes kobieta"
else:
    print "Jestes mezczyzna"
```

Przypomnienie materiału dotyczącego tekstów (łańcuchów znaków, list i innych typów danych w pytonie do przechowywania znaków.

```
Gdybyśmy chcieli przypisać do łańcucha znaków (nazwiemy go „s”) tekst = „Ala ma kota i 12
psów” zrobimy to tak:
>>> s = "Ala ma kota i 12 psów"
>>> s[0]          # zwróci znak, który jest przechowywany w łańcuchu s na 1 indeksie
(lancuchy są indeksowane od 0)
 'A'
>>> s[3] = ' '
>>> s[11:18]      # zwróci wycinek z łańcucha zaczynający się na indeksie 11 a kończący
na 18, a więc zwróci tak naprawdę podłańcuch znaków od 12 do 19 czyli „”
' 12 psów'
>>> s[:4]        # podłańcuch od indeksu 0 do 4
'Ala m'
>>> s[:5]+s[5:]   # podłańcuch znaków od indeksu 0 do 5 i od 5 do końca łańcucha s, a
więc:
'Ala ma kota i 12 psów'
```

Metody w klasie String

```
>>> s = "Ala ma kota i 12 psów"
>>> len ( s )
21
>>> s = s.replace ( 'kota', 'rybę' )
>>> test
'Ala ma rybę i 12 psów'
>>> s.count ( 'a' )
3
>>> s.find ( 'm' )
4
>>> s [4 ]
'm'
>>> s.split()
['Ala', 'ma', 'kota', 'i', '12', 'psów']
>>> s.split ( 'i' )
[Ala ma kota ', ' 12 psów']
>>> s.upper()
'ALA MA KOTA I 12 PSÓW'
>>> s.lower()
'ala ma kota i 12 psów'
>>> s.lower().capitalize()
'This is just a short string.'
>>> 'UPPER'.isupper()
True
>>> 'UpPEr'.isupper()
False
>>> 'lower'.islower()
True
>>> 'Lower'.islower()
False
```

Ćwiczenia do wykonania:

1. Napisz funkcję, która sprawdza, czy wprowadzone imię jest imieniem żeńskim czy męskim i zwraca jako rezultat odpowiedni komunikat: „Jesteś mężczyzną” albo „Jesteś kobietą”.
2. Napisz funkcję, która pyta o dwie liczby i mówi czy są one równe a jeśli nie to mówi jaka jest różnica między nimi.
3. Napisz funkcje: `liczba_pierwiastkow` i `oblicz_pierwiastki`, z których pierwsza będzie dla podanych parametrów równania kwadratowego ustalać ile jest rozwiązań tego równania, zaś druga obliczać te rozwiązania.
4. Napisz funkcję, która oblicza przychód po odliczeniu podatku od dochodu. Dochód i wielkość podatku (w procentach) podaje użytkownik (program musi go o to zapytać).
5. Napisz program, który dla podanego łańcucha znaków zamieni podaną pozycję na jakiś znak (też podany przez użytkownika). Funkcja, która dokona zamiany znaku z podanej pozycji musi sprawdzić, czy w ogóle pozycja wskazana jest prawidłowo (nie każemy np. Zamienic 6 znaku, podczas gdy łańcuch ma tylko 5 znaków). Funkcja powinna przyjmować 3 parametry (łańcuch do analizy, pozycję, którą należy podmienić, i znak który ma służyć do podmiany). Niech program wyświetla łańcuch przed i po zamianie.