Funkce

November 19, 2023

1 Funkce

Používání funkcí zvyšuje přehlednost kódu. Výhodou funkce je, že ji napíšete pouze jednou, ale spustit (volat) ji můžete vícekrát.

Syntaxe

def funkce():
    prikazy

Poznámky
- Klíčové slovo def, závorky pro definici funkce i pro volání nutné
- Funkce musí být definovaná před prvním voláním, typicky jsou všechny funkce uvedeny na začátku skriptu
- Dbejte na srozumitelné pojmenování funkcí, název funkce začíná typicky malým písmenem

[1]:
```
def helloWorld_v0():
    print("Hello!")
```

[3]:
```
helloWorld_v0()
helloWorld_v0()
```

```
Hello!
Hello!
```

1.1 Funkce s parametrem

Funkce mohou mít parametr, se kterým se volají.

Parametry používané uvnitř funkce (v naší ukázce níže parametry i a n) jsou dostupné pouze uvnitř funkce. Zvenku se k nim nemůžeme dostat. To je obvykle požadované chování.

[5]:
```
def helloWorld_v1(n):
    for i in range(n):
        print("Hello!")
```

[6]:
```
helloWorld_v1(3)
print("Dalsi volani")
helloWorld_v1(1)
```

```
Hello!
Hello!
Hello!
Dalsi volani
Hello!
```

Parametr funkce může být povinný (viz výše) či nepovinný (ukázka níže). Nepovinným parametrům přiřazujeme základní hodnotu.

[7]:
```
def helloWorld_v2(n=1, pozdrav="Hello"):
    for i in range(n):
        print(pozdrav)
```

Možnosti volání funkce s nepovinnými parametry:

[8]:
```
helloWorld_v2()
```

```
Hello
```

[9]:
```
helloWorld_v2(2)
```

```
Hello
Hello
```

[10]:
```
helloWorld_v2(2,'Ahoj :)')
```

```
Ahoj :)
Ahoj :)
```

[11]:
```
helloWorld_v2(n=4, pozdrav='Nazdárek!')
```

```
Nazdárek!
Nazdárek!
Nazdárek!
Nazdárek!
```

[12]:
```
helloWorld_v2(pozdrav='Čau')
```

```
Čau
```

Úkoly

Napište funkci, která
- spočítá, kolik je v seznamu sudých čísel
- vybere ze seznamu sudá čísla - dostane dva seřazené seznamy čísel a vypíše jejich průnik
- vypočte alternující ciferný součet (pro číslo 8643 bude výsledek $8 - 6 + 4 - 3 = 3$)

[14]:
```
def kolikSudych(seznam):
    pocet = 0
    for i in seznam:
        if i%2==0:
```

```
            pocet += 1
    print(pocet)
```

[15]:
```
s = [3, 6, 8, 11, -4, 6]
kolikSudych(s)
```

```
4
```

[9]:
```
zoznam2=[1,2,3,4,5,6,7,8,9,10,11,12]
zoznam = [7,9,12,20,21,64,5]

def prienik():
    zoznam3 = []
    for p in zoznam:
        for h in zoznam2:
            if p == h:
                zoznam3.append(p)
                break

    return zoznam3

print(prienik())
print(prienik())
```

```
[7, 9, 12, 5]
[7, 9, 12, 5]
```

[16]:
```
seznam=[1,2,3,4,5,6,7,8,9,10,11,12]
seznam2 = [5,7,9,12,20,21,64]

def prienik(zoznam, zoznam2):
    zoznam3 = []
    for p in zoznam:
        for h in zoznam2:
            if p == h:
                zoznam3.append(p)
                break

    return zoznam3

prienik(seznam, seznam2)
prienik(seznam, seznam2)
```

```
[5, 7, 9, 12]
[5, 7, 9, 12]
```

1.2 Funkce s výsledkem

Funkce může také vracet hodnotu (hodnoty). Typ proměnné, kterou funkce vrací není omezen. Může vracet např. celé číslo, řetezec, seznam.

Klíčové slovo pro vracení hodnoty je **return** - ukončí funkci a vrátí výsledek. Pokud je použit **return** bez výsledku, vrací se typ None.

[19]:
```
def soucet(cislo1, cislo2):
    return cislo1 + cislo2
    print('Vypis uvnitr funkce')
```

[22]:
```
suma = soucet(1, 100)
print(suma)
```

```
101
```

Pokud chceme vrátit více hodnot, můžeme vrátit jednotlivé hodnoty oddělené čárkou (typ tuple).

[23]:
```
def serad(cislo1, cislo2):
    if cislo1<cislo2:
        return cislo1, cislo2
    else:
        return cislo2, cislo1
```

[23]:
```
serad(2,2)
```

[23]:
```
(2, 2)
```

[25]:
```
a, b = serad(10,8)
print(a)
print(b)
```

```
8
10
```

[25]:
```
serad(8,10)
```

[25]:
```
(8, 10)
```

Úkoly

Napište funkci, která

- vrátí n-té Fibonacciho číslo (F (0) = 0, F (1) = 1, F (n) = F (n − 1) + F (n − 2)) - vrátí n-faktoriál (n! = n*(n-1)*(n-2)*….*2*1)
- dostane koeficienty kvadratické rovnice ax^2 + bx + c = 0 a vrátí seznam jejích kořenů (na množině reálných čísel)

[23]:
```
def fibonacci(n):
    if n == 0:
        return 0
```

```
    if n == 1:
        return 1
    fn_2 = 0
    fn_1 = 1
    for i in range(n-1):
        fn = fn_1 + fn_2
        fn_2 = fn_1
        fn_1 = fn
    return fn
```

[25]:
```
fibonacci(8)
```

[25]:
```
21
```

[27]:
```
def kvadraticka(a,b,c):
    D = b**2 - 4*a*c
    if D < 0:
        return("Rovnice nema v R reseni.")
    else:
        return((-b+D**0.5)/(2*a),(-b-D**0.5)/(2*a))

print(kvadraticka(1,-5,6))
```

```
(3.0, 2.0)
```

1.3 Globální proměnné

Funkce může načíst i proměnné vně funkce, ale pouze pokud do nich nic nepřiřazuje (nemění je).

Pokud potřebujeme, aby se při běhu funkce změnil parametr definovaný mimo funkci, použijeme klíčové slovo **global**.

UPOZORNĚNÍ S globálními proměnnými opatrně, ať si nepřepíšete proměnnou, kde to nečekáte....

Příklad: - Funkce načte globální proměnnou *jmeno*
- Proměnnou *pocetClenu* potřebujeme definovat jako globální

[10]:
```
jmeno = 'Novak'
pocetClenu = 12

def pridaniClena():
    print(jmeno)                                    # promennou jmeno funkce␣
 ↪zna
    print('Počet členů před změnou', pocetClenu)    # promennou pocetClenu␣
 ↪funkce zna
    pocetClenu += 1                                 # hodnotu promenne␣
 ↪pocetClenu funkce nemuze menit
    print('Počet členů pro změně', pocetClenu)
```

```
pridaniClena()
```

```
Novak
```

```
---------------------------------------------------------------------------
UnboundLocalError                         Traceback (most recent call last)
Cell In[10], line 10
      7     pocetClenu += 1                                    # hodnotu promenn␣
 ↪pocetClenu funkce nemuze menit
      8     print('Počet členů pro změně', pocetClenu)
---> 10 pridaniClena()

Cell In[10], line 6, in pridaniClena()
      4 def pridaniClena():
      5     print(jmeno)                                       # promennou jmeno␣
 ↪funkce zna
----> 6     print('Počet členů před změnou', pocetClenu)      # promennou␣
 ↪pocetClenu funkce zna
      7     pocetClenu += 1                                    # hodnotu promenn␣
 ↪pocetClenu funkce nemuze menit
      8     print('Počet členů pro změně', pocetClenu)

UnboundLocalError: cannot access local variable 'pocetClenu' where it is not␣
 ↪associated with a value
```

[11]:
```
jmeno = 'Novak'
pocetClenu = 12

def pridaniClena(jmeno):
    global pocetClenu                                  # promenna pocetClenu␣
 ↪jako globalni promenna
    print(jmeno)                                       # promennou jmeno funkce␣
 ↪zna
    print('Počet členů před změnou', pocetClenu)       # promennou pocetClenu␣
 ↪funkce zna
    pocetClenu += 1                                    # hodnotu promenne␣
 ↪pocetClenu funkce uz menit muze
    print('Počet členů pro změně', pocetClenu)

pridaniClena(jmeno)
print('Kontrola: Počet členů', pocetClenu)
```

```
Novak
Počet členů před změnou 12
Počet členů pro změně 13
```

```
Kontrola: Počet členů 13
```

Řešení bez globálních proměnných:

[32]:
```
jmeno = 'Novak'
pocetClenu = 12

def pridaniClena(name):
    print(name)
    return 1

pocetClenu += pridaniClena(jmeno)
print('Kontrola: Počet členů', pocetClenu)
```

```
Novak
Kontrola: Počet členů 13
```

1.4 Další

Vracení více hodnot

[14]:
```
def aritmetika(a, b):
    soucet = a + b
    rozdil = a - b
    nasobek = a*b
    return soucet, rozdil, nasobek

pocitani = aritmetika(8,10)
print(pocitani)
print(f'rozdil: {pocitani[1]}.')
```

```
(18, -2, 80)
rozdil: -2.
```

[17]:
```
#Ukazka: Prohozeni dvou hodnot
cislo1 = 10
cislo2 = 100

print('Pred prohozenim: ', cislo1, cislo2)
cislo1, cislo2 = cislo2, cislo1
print('Po prohozeni: ', cislo1, cislo2)
```

```
Pred prohozenim:  10 100
Po prohozeni:  100 10
```

2 Úkoly

Napište funkci, která
- vrátí počet sudých čísel v seznamu (parametr funkce bude seznam)
- vrátí seznam sudých čísel v seznamu (parametr funkce bude seznam, vrátí se nový seznam)
- zjistí, zda je dané číslo prvočíslo (vratí True-False)
- vypíše všechna prvočísla do zadaného čísla (včetně)

Další úkoly

Napište funkci, která
- vypočítá obsah obdélníka (parametry funkce budou strany obdélníka)
- rozmění částku na drobné (parametr funkce bude částka na rozměnění, výstup bude počet 50-ti korun, 20-ti korun, …, 1 korun)
- vypočte průměrnou známku ze seznamu známek (parametr funkce bude seznam známek)
- načte seznam čísel zadávaných po jednom uživatelem do seznamu. Načítání je ukončeno -1, ta už do seznamu nepatří (funkce nebude mít vstupní parametr, bude vracet seznam čísel)
- převede teplotu v Kelvinech nebo Fahrenheitech na teplotu ve stupních Celsia (funkce bude mít dva parametry - číselnou hodnotu teploty a údaj, jestli jde o teplotu v Kelvinech či Fahrenheitech)

[ ]: