



**Code
Academy**



1 LYGIS

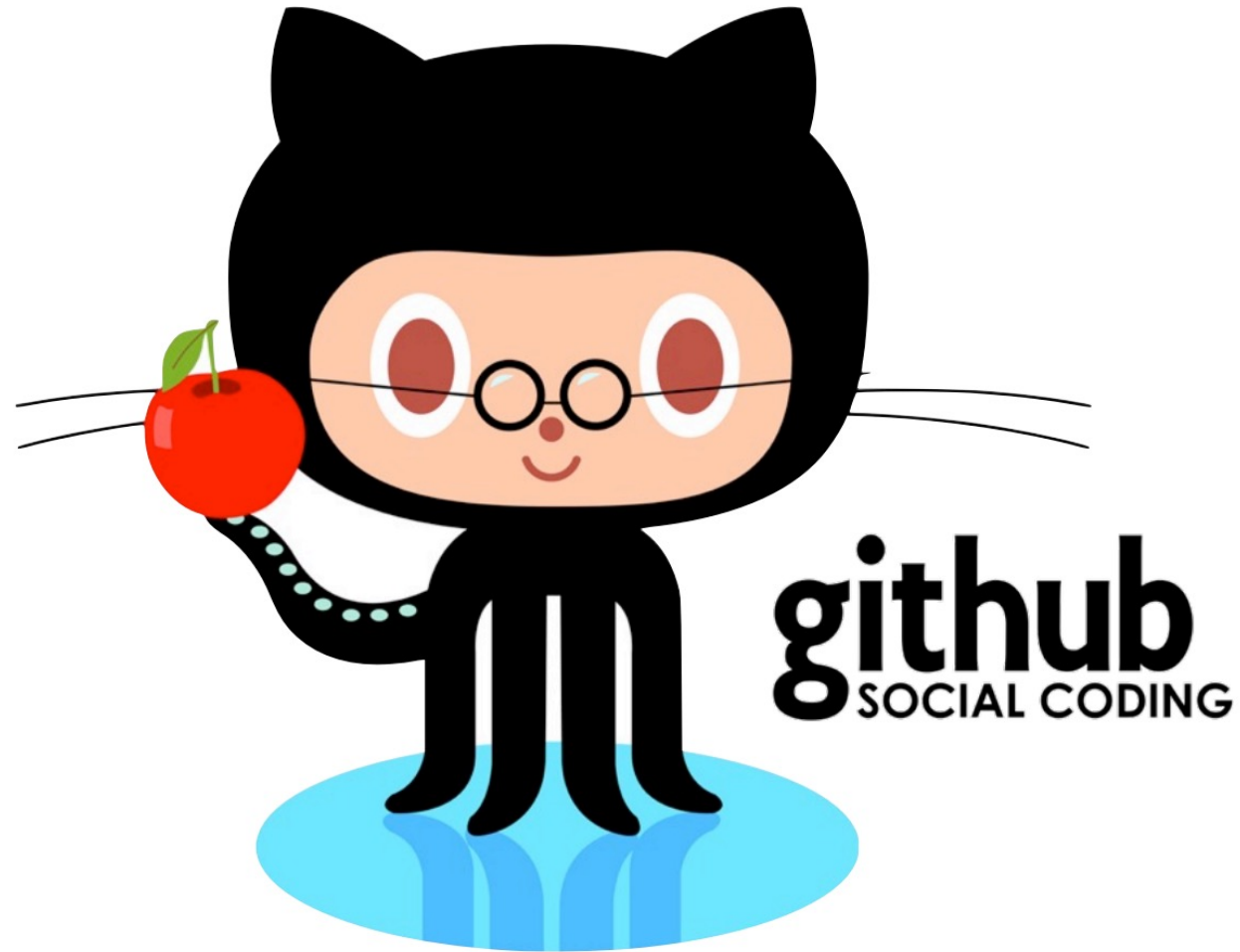
4 paskaita. GitHub ir Funkcijos

2023

Aurimas Aleksandras Nausėdas



**Code
Academy**



Python pradedančiųjų kurso pasakitos ir šiek tiek daugiau medžiagos - <https://github.com/aurimas13/Python-Beginner-Course>



Šiandien išmoksite

01

Kas yra funkcijos

02

Kas yra lambda funkcijos

03

Atlikti veiksmus naudojant funkcijas

04

Iš funkcijos grąžinti norimą reikšmę



```
def pasisveikinti(vardas):  
    print(f"Sveikas, {vardas}")
```

```
pasisveikinti("Tomas")  
pasisveikinti("Jonas")  
pasisveikinti("")
```

```
# Sveikas, Tomas  
# Sveikas, Jonas  
# Sveikas,
```

```
def kvadratas(skaicius):  
    kvadratu = skaicius ** 2  
    print(kvadratu)
```

```
kvadratas(2)  
# 4
```

Funkcijos su argumentais



```
def kvadratu(skaicius):  
    rezultatas = skaicius ** 2  
    print(rezultatas)  
  
kvadratu(3)  
# 9  
  
daugyba = kvadratu(3) * 2  
print(daugyba)  
# TypeError: unsupported operand type(s) for *: 'NoneType' and 'int'  
  
print(kvadratu(3))  
None
```

Funkcijos be return trūkumas



```
def kvadratu(skaicius):  
    rezultatas = skaicius ** 2  
    return rezultatas  
  
daugyba = kvadratu(3) * 2  
print(daugyba)  
# 18
```

**Funkcijos su grąžinama
reikšme (return)**



```
def skaiciu_suma(skaicius1, skaicius2, skaicius3):  
    suma = skaicius1 + skaicius2  
    daugyba = suma * skaicius3  
    return daugyba  
  
print(skaiciu_suma(2, 5, 20))  
# 140
```

Funkcijos su keliais argumentais



```
def skaiciu_suma(skaicius1, skaicius2, skaicius3=1):  
    rezultatas = (skaicius1 + skaicius2) * skaicius3  
    return rezultatas
```

```
print(skaiciu_suma(2, 5, 4))  
# 28
```

```
print(skaiciu_suma(2, 5))  
# 7
```

Funkcijas su nebūtinais argumentais 1



```
def skaiciu_suma(skaicius1=10, skaicius2=10, skaicius3=1):  
    rezultatas = (skaicius1 + skaicius2) * skaicius3  
    return rezultatas
```

```
print(skaiciu_suma())  
print(skaiciu_suma(2))  
print(skaiciu_suma(2, 5))  
print(skaiciu_suma(2, 5, 4))
```

```
# 20  
# 12  
# 7  
# 28
```

Funkcijas su nebūtinais argumentais 2



```
def skaiciu_suma(skaicius1=10, skaicius2=10, skaicius3=1):  
    rezultatas = (skaicius1 + skaicius2) * skaicius3  
    return rezultatas  
  
print(skaiciu_suma(skaicius3=3))  
print(skaiciu_suma(skaicius1=20, skaicius3=3))  
  
# 60  
# 90
```

Kaip priskirti konkretų argumentą (-us)



```
def daug_kvadratu(*args):  
    for skaicius in args:  
        print(skaicius ** 2)  
  
daug_kvadratu(4, 5, 7, 8, 9, 10)  
  
# 16  
# 25  
# 49  
# 64  
# 81  
# 100
```

Funkcijos su neribotais argumentais



```
def spausdinti_reiksmes(**kwargs):
    for raktas, reiksme in kwargs.items():
        print(raktas, reiksme)

spausdinti_reiksmes(vardas="Tomas", lytis="Vyras", amzius=29, daiktai=["Telefonas", "Ausinės", "Krepšys"])

# vardas Tomas
# pavarde Rutkauskas
# lytis Vyras
# amzius 29
# daiktai ['Telefonas', 'Ausinės', 'Krepšys']
```

Funkcijos su neribotais argumentais 2



```
def spausdinti_reiksmes(vardas, pavarde, **kwargs):
    print(f"Vardas: {vardas}, Pavardė: {pavarde}")
    for raktas, reiksme in kwargs.items():
        print(raktas, reiksme)

spausdinti_reiksmes("Tomas", "Rutkauskas", lytis="Vyras", amzius=29, daiktai=["Telefonas", "Ausinės", "Krepšys"])

# Vardas: Tomas, Pavardė: Rutkauskas
# lytis Vyras
# amzius 29
# daiktai ['Telefonas', 'Ausinės', 'Krepšys']
```

Funkcijos su įprastais ir neribotais argumentais



```
def spausdinti_reiksmes(skaicius1, skaicius2, *args):  
    print("Skaičių suma: ", skaicius1 + skaicius2)  
    for vienas in args:  
        print(vienas)
```

```
spausdinti_reiksmes(5, 2, "Labas", 5.26)
```

```
# Skaičių suma: 7  
# Labas  
# 5.26
```

Funkcijos su įprastais ir neribotais argumentais 2



```
globalus = 10

def funkcija():
    lokalus = 12
    suma = globalus + lokalus
    print(suma)

kita_suma = globalus + lokalus
print(kita_suma)
# NameError: name 'lokalus' is not defined

funkcija()
# 22
```

Globalūs ir lokālūs kintamieji



```
def funkcija(parametras1, parametras2):  
    ...  
  
    :param parametras1:  
    :param parametras2:  
    :return:  
    ...  
  
    return
```

```
def funkcija(parametras1, parametras2):  
    ...  
    Ši funkcija visiškai nieko nedaro  
    :param parametras1: Nereikalingas parametras  
    :param parametras2: Dar vienas nereikalingas  
    parametras  
    :return: Nieko negražina  
    ...  
  
    return
```

Funkcijos komentavimas (Doststring)


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

```
lambda a: a ** 2
```

```
kvadratu = lambda a: a ** 2  
  
print(kvadratu(2))
```

```
sarasas = [2, 5, 4, 65, 78, 99, 38]  
  
sarasas2 = map(lambda a: a ** 2, sarasas)  
  
for skaicius in sarasas2:  
    print(skaicius)
```



Anoniminės (Lambda) funkcijos

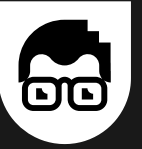
Tai supaprastinta funkcija, paprastai naudojama tik kartą, visas kodas telpa vienoje eilutėje.



```
daugyba_is_saves = [lambda i=skaicius: i*i for skaicius in range(1, 6)]  
for vienas in daugyba_is_saves:  
    print(vienas())
```

```
keliameji = [lambda i=metai: i for metai in range(1900, 2101) if  
              (metai % 400 == 0) or (metai % 100 != 0 and metai % 4 == 0)]  
for vienas in keliameji:  
    print(vienas())
```

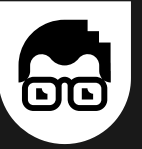
Anoniminės (Lambda) funkcijos 2



Užduotis nr. 1

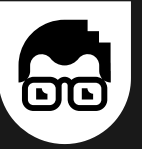
Sukurkite ir išsibandykite funkcijas, kurios:

1. Gražinti trijų paduotų skaičių sumą.
2. Gražintų paduoto sąrašo iš skaičių, sumą.
3. Atspausdintų didžiausią iš kelių paduotų skaičių (panaudojant *args).
4. Gražintų paduotą stringą atbulai.
5. Atspausdintų, kiek paduotame stringe yra žodžių, didžiųjų ir mažųjų raidžių, skaičių.
6. Gražintų sąrašą tik su unikaliais paduoto sąrašo elementais.
7. Gražintų, ar paduotas skaičius yra pirminis.
8. Išrikiuotų paduoto stringo žodžius nuo paskutinio iki pirmojo
9. Gražina, ar paduoti metai yra keliamieji, ar ne.
10. Atspausdina, kiek nuo paduotos sukakties praėjo metų, mėnesių, dienų, valandų, minučių, sekundžių.



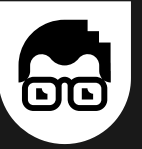
Užduotis nr. 2

1. Sukurti funkciją, kuri patikrintų, ar paduotas Lietuvos piliečio asmens kodas yra validus.
2. Padaryti, kad programa sugeneruotų teisingą asmens kodą (panaudojus anksčiau sukurtą funkciją) pagal įvestą lytį, gimimo datą ir eilės numerį).



Užduotis nr. 3

1. Sukurti funkciją, kuri grąžintų True reikšmę, jei įvesto skaičiaus pirma skaitmenų pusė yra lygi antrajai, priešingu atveju grąžintų False.
2. Parašyti funkciją, kuri grąžintų, kiekvieno elemento gretimą skaičių. Pvz:
Input: 5678
Output: 5 – 46, 6 – 57, 7 – 68, 8 - 79



Namų darbas

Užbaigti klasėje nepadarytas užduotis



Išspręsti paskaitos uždaviniai (įkelti pirmadienį)

<https://github.com/aurimas13/Python-Beginner-Course/tree/main/Programs>

Asmens kodas

Info apie asmens kodo sudarymą

https://lt.wikipedia.org/wiki/Asmens_kodas

Codingbat

Užduotys Python praktikai

codingbat.com

**Naudinga
informacija**