Code Academy

1 LYGIS

7 paskaita. Darbas su katalogais ir failais

2023

Aurimas Aleksandras Nausėdas

Šiandien išmoksite

01 Kas yra "os" modulis

02 Kaip kurti, nuskaityti ir redaguoti tekstinius failus

03 Kaip išsaugoti kintamuosius/objektus faile

```
import os
```

```
print(dir(os))
```

```
print(os.getcwd())

# C:\Users\Donoras\PycharmProjects\failai
```

```
os.chdir('C:\\Users\\Donoras\\Desktop')
print(os.getcwd())

# C:\Users\Donoras\Desktop
```

```
print(os.listdir())

# ['Demo katalogas', 'paveikslelis.jpg', 'tekstas.txt']
```

```
os.mkdir("Naujas katalogas")
```

```
os.makedirs("Naujas katalogas/Katalogas kataloge")

print(os.listdir())

# ['Demo katalogas', 'Naujas katalogas', 'paveikslelis.jpg', 'tekstas.txt']
```

OS modulis

os – operating system modulis

- dir(os) - kokias komandas turi modulis
- os.getcwd() - katalogas kuriame esame
- os.chdir() - pakečia katalogą kuriame esame
- os.listdir() - parodo kokie failai yra kataloge
- os.mkdir() - sukuria naują katalogą
- os.makedirs() - sukuria katalogų medį

```
print(os.stat("Demo Katalogas"))
```

```
print(os.stat("naujas_tekstas.txt"))

# os.stat_result(st_mode=33206,
# st_ino=11258999068714091, st_dev=987816996,
# st_nlink=1, st_uid=0, st_gid=0, st_size=279,
# st_atime=1553103727, st_mtime=1553072965,
# st_ctime=1553101362)
```

```
print(os.stat("naujas_tekstas.txt").st_size)

# 279 (baitais)
```

```
print(os.stat("naujas_tekstas.txt").st_mtime)

# 1553072965.1983721
```

- os.stat() - failo/katalogo informacija
- os.stat().st_size – failo dydis baitais
- os.stat().st_mtime – failo modifikavimo laikas

```
from datetime import datetime
data = os.stat("naujas_tekstas.txt").st_mtime
print(datetime.fromtimestamp(data))

# 2019-03-20 11:09:25.198372
```

Kaip sužinoti suprantamu formatu

```
with open("failas.txt", 'w') as failas:
    failas.write("Sveikas, pasauli!")
```

```
failas = open("failas.txt", 'w')
failas.write("Sveikas, pasauli!")
failas.close()
```

Tekstinių failų kūrimas ir nuskaitymas

Kaip sukurti tekstinį failą: (jei failo nėra, bus sukurtas naujas, jei yra - bus įrašoma jame), 2 būdai:

- su "with open(.....) as .... :"
- su rankinių failo uždarymu

```
with open("failas.txt", 'r') as failas:
    print(failas.read())

# Sveikas, pasauli!
```

Kaip nuskaityti tekstą iš failo

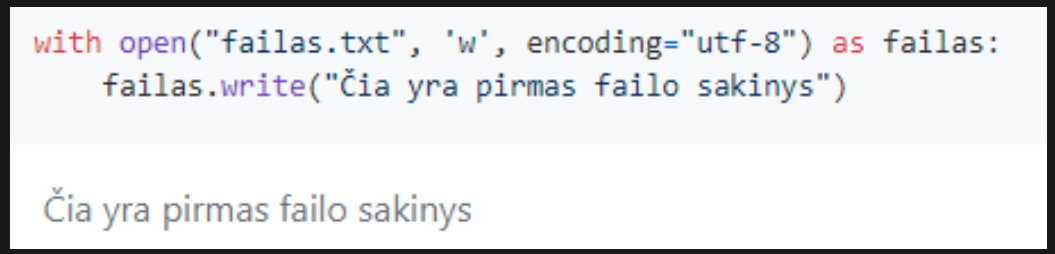
```
with open("failas.txt", 'r+') as failas:
    print(failas.read())
    failas.write("Labas rytas, pasauli!")

# Sveikas, pasauli!

with open("failas.txt", 'r') as failas:
    print(failas.read())

Sveikas, pasauli!Labas rytas, pasauli!
```

Kaip įrašyti ir nuskaityti failą vienu metu



Kaip į failą įrašyti lietuviškus rašmenis

```
with open("failas.txt", 'r', encoding="utf-8") as failas:
    print(failas.read())
```

```
Čia yra pirmas failo sakinys
```

Kaip nuskaityti failą lietuviškus rašmenis

```python
with open("failas.txt", 'a', encoding="utf-8") as failas:
    failas.write("Čia yra pirmas sakinys \n")

with open("failas.txt", 'a', encoding="utf-8") as failas:
    failas.write("Čia yra antras sakinys \n")
```

```
Čia yra pirmas sakinys

Čia yra antras sakinys
```

Kaip pridėti, o ne perrašyti failo eilutę

```python
with open("failas.txt", 'w', encoding="utf-8") as failas:
    failas.write("Test")
    failas.seek(0)
    failas.write("BE")
```

Kaip perrašyti tekstą norimoje vietoje

```
with open("failas.txt", 'r', encoding="utf-8") as failas:
    print(failas.readline())
    print(failas.readline())
    print(failas.readline())

# Čia yra pirmas sakinys
# Čia yra antras sakinys
# Čia yra trečias sakinys
```

```
with open("failas.txt", 'r', encoding="utf-8") as failas:
    print(failas.readlines())

# ['Čia yra pirmas sakinys \n', 'Čia yra antras sakinys \n', 'Čia
# yra trečias sakinys \n', 'Čia yra ketvirtas sakinys \n', 'Čia yra
# penktas sakinys \n', 'Čia yra šeštas sakinys \n', 'Čia yra septintas
# sakinys \n', 'Čia yra aštuntas sakinys \n', 'Čia yra devintas sakinys \
# n', 'Čia yra dešimtas sakinys \n']
```

Kaip nuskaityti failą po vieną eilutę

- readline()
- readlines()

```python
with open("failas.txt", 'r', encoding="utf-8") as failas:
    for eilute in failas:
        print(eilute)

# Čia yra pirmas sakinys

# Čia yra antras sakinys

# Čia yra trečias sakinys

# Čia yra ketvirtas sakinys
```

Iteravimas per failo eilutes

```
with open("failas.txt", 'r', encoding="utf-8") as failas:
    for eilute in failas:
        print(eilute, end="")

# Čia yra pirmas sakinys
# Čia yra antras sakinys
# Čia yra trečias sakinys
# Čia yra ketvirtas sakinys
# Čia yra penktas sakinys
# Čia yra šeštas sakinys
# Čia yra septintas sakinys
# ............
```

Iteravimas per failo eilutes be tarpų tarp jų

```python
with open("failas.txt", 'r', encoding="utf-8") as failas:
    print(failas.read(100))

# Čia yra pirmas sakinys
# Čia yra antras sakinys
# Čia yra trečias sakinys
# Čia yra ketvirtas sakinys

print(failas.read(100))

# Čia yra penktas sakinys
# Čia yra šeštas sakinys
# Čia yra septintas sakinys
# Čia yra aštuntas sakinys

print(failas.read(100))

# Čia yra devintas sakinys
# Čia yra dešimtas sakinys
```

Kaip nuskaityti ribotą kiekį duomenų

```
with open("failas.txt", 'r') as r_failas:
    with open("failo_kopija.txt") as w_failas:
        for r_eilute in r_failas:
            w_failas.write(r_eilute)
```

Darbas su dviem failais (teksto kopijavimas iš vieno į kitą)

```
with open("logo.png", 'rb') as r_failas:
    with open("logo_kopija.png", 'wb') as w_failas:
        for r_eilute in r_failas:
            w_failas.write(r_eilute)
```

Dvejetainių failų kopijavimas

```python
import pickle

a = 1024

with open("a.pkl", "wb") as pickle_out:
    pickle.dump(a, pickle_out)
```

```python
import pickle

with open("a.pkl", "rb") as pickle_in:
    naujas_a = pickle.load(pickle_in)

print(naujas_a)

# 1024
```

Kaip į failą išsaugoti kintamuosius/objektus

Objektų saugojimui naudojame "pickle" biblioteką:

- pickle.dump() - įrašymas
- pickle.load() - nuskaitymas

```python
import pickle

zodynas = {1:"Pirmas", 2:"Antras", 3:"Trečias"}

with open("zodynas.pkl", "wb") as pickle_out:
    pickle.dump(zodynas, pickle_out)
```

```python
import pickle

with open("zodynas.pkl", "rb") as pickle_in:
    naujas_zodynas = pickle.load(pickle_in)

print(naujas_zodynas)

# {1: 'Pirmas', 2: 'Antras', 3: 'Trečias'}
```

Masyvų saugojimas pickle faile

```python
a = 10
b = 7
c = 23

with open("abc.pkl", "wb") as pickle_out:
    pickle.dump(a, pickle_out)
    pickle.dump(b, pickle_out)
    pickle.dump(c, pickle_out)
```

```python
with open("abc.pkl", "rb") as pickle_in:
    nauja_a = pickle.load(pickle_in)
    nauja_b = pickle.load(pickle_in)
    nauja_c = pickle.load(pickle_in)

print(nauja_a)
print(nauja_b)
print(nauja_c)

# 10
# 7
# 23
```

```python
import pickle

with open("abc.pkl", "rb") as pickle_in:
    while True:
        try:
            print(pickle.load(pickle_in))
        except EOFError:
            break

# 10
# 7
# 23
```

Kelių kintamųjų saugojimas pickle faile

```python
import pickle

class Automobilis:
    def __init__(self, marke, modelis):
        self.marke = marke
        self.modelis = modelis

automobiliai = [Automobilis("Toyota", "Avensis"), Automobilis("Toyota", "Corolla"), Automobilis("Toyota", "Camry")]

with open("automobilis.pkl", "wb") as failas:
    pickle.dump(automobiliai, failas)

with open("automobilis.pkl", "rb") as failas:
    automobiliai = pickle.load(failas)
    for automobilis in automobiliai:
        print("Markė", automobilis.marke)
        print("Modelis", automobilis.modelis)
```

Objektų sąrašo saugojimas pickle faile

Užduotis nr. 1

Sukurti programą, kuri:

- Sukurtų failą „Tekstas.txt“ su pilnu tekstu "Zen of Python".
- Atspausdintų tekstą iš sukurto failo
- Paskutinėje sukurto failo eilutėje pridėtų šiandienos datą ir laiką
- Sunumeruotų teksto eilutes (kiekvienos pradžioje pridėtų skaičių).
- Sukurtame faile eilutę "Beautiful is better than ugly." pakeistų į "Gražu yra geriau nei bjauru."
- Atspausdintų visą failo tekstą atbulai
- Atspausdintų, kiek failo tekste yra žodžių, skaičių, didžiųjų ir mažųjų raidžių
- Nukopijuotų visą sukurto failą tekstą į naują failą, tik DIDŽIOSIOMIS RAIDĖMIS

Patarimai:

- Naudoti from datetime import datetime, datetime.today()
- Kintamajam priskirti sakinį, kuriuo bus operuojama, eilutėmis
- Kai kur galima panaudoti funkcijas iš praeitų pamok

Užduotis nr. 2

Sukurti programą, kuri:

- Leistų vartotojui įvesti norimą eilučių kiekį
- Įrašytų įvestą tekstą atskiromis eilutėmis į failą
- Leistų vartotojui įrašyti norimą kuriamo failo pavadinimą

Patarimai:

- Sukurti while ciklą, kuris užsibaigtų tik įvedus vartotojui tuščią eilutę (nuspaudus ENTER)

Užduotis nr. 3

Sukurti programą, kuri:

- Kompiuterio darbalaukyje (Desktop) sukurtų katalogą „Naujas Katalogas“
- Šiame kataloge sukurtų tekstinį failą, kuriame būtų šiandienos data ir laikas
- Atspausdintų šio tekstinio failo sukūrimo datą ir dydį baitais

Patarimai:

- Failo sukūrimo datą galima pasiekti per os.stat(„Failas.txt“).st_ctime

Užduotis nr. 4

Sukurti minibiudžeto programą, kuri:

- Leistų vartotojui įvesti pajamas arba išlaidas (su "-" ženklu)
- Pajamas ir išlaidas saugotų sąraše, o sąrašą pickle faile (uždarius programą, įvesti duomenys nedingtų)
- Atvaizduotų jau įvestas pajamas ir išlaidas
- Atvaizduotų įvestų pajamų ir išlaidų balansą (sudėtų visas pajamas ir išlaidas)

Patarimas:

- import pickle

Namų darbas

Užbaigti klasėje nepadarytas užduotis

Išspręsti paskaitos uždaviniai (įkelti ketvirtadienį)	https://github.com/aurimas13/Python-Beginner-Course/tree/main/Programs
DB browser for SQLite Duomenų bazės SQLite programa	https://sqlitebrowser.org/

Naudinga informacija