LD4. Interneto paslaugų scenarijai

Aprašymas

Darbui naudosime scenarijų kalbą Python. Trumpas Python sintaksės aprašymas:

Kai kurios operacijos su kintamaisiais:

```
> a, b = 5, 3 # a, b: (5, 3)
> b - a - 2
> 17 / a # dalyba
> b == a
> a >= b
> b < a < 10
> b != a
> not b == a
> 10 % b
> b**3 # kėlimas laipsniu
```

Sąlygos sakinys if:

```
if x > 1000:
   print ("x is big")
elif x > 100:
   print ("x is average")
elif x > 10:
   print ("x is small")
else:   # visais likusiais atvejais
   print ("x is not interesting")
```

Eilutės:

```
> print (‘Labas‘)  # galima naudoti ir dvigubas kabutes
> print ("x yra" + str(x))
> print ("x yra“, x)
> print ("x yra {}".format(x))
> print (‘y:’ + y)
> print  # tuščia eilutė
> linkejimas = "Good Luck"
> vardas = "Petras"
> sakinys = linkejimas +' '+ vardas + "!"
> print (sakinys)
Good Luck Petras!
> linkejimas.upper()  # tekstą paversti didžiosiom raidėm
> 'obuolys'.replace('o', 'a')  # keičiame 'o'  į  'a'
> len('labas rytas') # teksto ilgis
> duom_mas = duom_txt.split(' ')
# skaidoma pagal tarpą
> duom_txt.strip() # tarpų nuėmimas
> eilutes = duom_txt.strip().split('\n')
> x.endswith(".png")
# eilutė baigiasi .png
> "Labas, {}, duok {} eur?".format (vardas, 2.5)
'Labas, Petrai, duok 2.5 eur?'
```

Ciklas for:

```
> for x in range(10): print (x**3)
0
1
8
27
64
125
216
343
512
729
range(10) – seka 0,1,…,9

> for x in [5, 3, 6, 1]:
      print (x)
5
3
6
1
```

Sąrašai (masyvai):

```
> my_list = [4, 7, 5, -2, 13, 0]
> print (my_list[2])
5
> my_list[2] = 1000
> my_list.append( 13 ) # prijungti pabaigoje
> my_list
[4, 7, 1000, -2, 13, 0, 13]
> my_list[1], my_list[3] = my_list[3], my_list[1]
# sukeisti elementus vietomis
> my_list[2:5]
[1000, -2, 13]
> my_list[:3] # [3:] nuo trečio iki pabaigos
[4, 7, 1000]
>lyginiai = my_list[::2]
>nelyginiai = list1[1::2]
> del list1[0] # elemento šalinimas
> range(1, 11)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
> list1 = range(1, 21, 3)
> list1
[1, 4, 7, 10, 13, 16, 19]
> len(list1) # sąrašo ilgis
7
> range( len(my_list) ) # indeksų sąrašas
[0, 1, 2, 3, 4, 5, 6]
> sum(my_list)
70
```

Ciklas while:

```
> sar = [5, 3, 6, 1]
> i = 0   # skaitliukas
> while i < len(sar):
...     print (i, sar[i])  # skliaustų nėra
...     i = i+1          # svarbu eilutės postūmis
0 5
1 3
2 6
3 1
```

Atvaizdžiai (dictionary):

```
> kainos = {'bulves': 1.5,  'morkos': 2 }
> kainos['bulves']
1.5
> kainos['kefyras'] = 3
> kainos
{'kefyras': 3, 'morkos': 2, 'bulves': 1.5}
> kainos['kefyras'] = 2 # atnaujinimas
> kainos
{'kefyras': 2, 'morkos': 2, 'bulves': 1.5}
> kainos.pop('morkos') # šalinimas
2
> kainos.update ( {'kiauliena': 15, 'bulves': 3.5,
'saldainiai':15, } )  # papildymas kitu žodynu
> kainos.keys()
> kainos.values()
> kainos.items() # bendras sąrašas

Iteravimas atvaizdžiu:
> knights = {'gallahad': 'the pure', 'robin': 'the brave'}
> for k, v in knights.iteritems():
   print (k, v)
gallahad the pure
robin the brave
```

Failų skaitymas ir rašymas:

```
> f = open('/tmp/failas', 'w')
> print (f)
<open file '/tmp/failas', mode 'w' at 80a0960>

Failų turinio perskaitymas:
f.read(size) – baitų kiekis.
f.read() – viską.
f.readline() – vieną eilutę.
f.readlines()
['This is the first line of the file.\n', 'Second line of the
file\n']
> for line in f:
     print (line)
```

GET užklausa ir json duomenų nuskaitymas:

API grąžinami duomenys json formatu:

```
{
   "current": {
      “value1": 100,
      “value2": 14.0
    },
}
```

Kortežai (tuple):

```
> t = 12345, 54321, 'labas!'
> t[0]
12345
> t
(12345, 54321, 'labas!')
```

Funkcijos:

```
> def kvadratas(): # funkcija pradedama nuo def
...     for i in range(4):
...         pirmyn(100)
...         kairen(90)
> def kvadratas2(krastines_ilgis):
...     for i in range(4):
...         pirmyn(krastines_ilgis)
...         kairen(90)
> kvadratas()
> kvadratas(100)

> def kv_spalvotas(krastines_ilgis, spalva='black'):
...     pencolor ( spalva )
...     kvadratas( krastines_ilgis )
> kv_spalvotas(70) # spalva visada bus juoda

> def skaiciavimai(a, b):
...     suma = a + b
...     daugyba =  a * b
...     return suma, daugyba
> suma, daugyba = skaiciavimai(3, 4)
```

Archyvuoto failo skaitymas iš url:

```
from io import BytesIO
import sys
import requests
from zipfile import ZipFile
url = “https://adresas.lt/failas.zip”
filename = requests.get(url).content
zf = ZipFile( BytesIO(filename), 'r' )
for item in zf.namelist():
  file = zf.open(item)
```

Archyvuoto failo skaitymas:

```
import tarfile
file_path = “/some/path/file.tar.gz”
file = tarfile.open(file_path, "r:gz")
for filename in file.getmembers():
   f = file.extractfile(filename)
```

CSV failo skaitymas:

```
from io import TextIOWrapper
import csv
file = open('/tmp/failas', 'w')
csvreader = csv.reader(TextIOWrapper(file, 'utf-8’))
rows = []
for row in csvreader:
       rows.append(row)
```

```
import requests
import json
some_api = “https://some.url/api”
r = requests.get(url = some_api)
data = r.json()
print(json.dumps(data, indent=4, sort_keys=True))
data1 = data[‘current’][‘value1’]
data2 = data[‘current’][‘value2’]
print (str(data1) + " : " + str(data2))
```

```
file.close()
```

IP adreso konversija į int:

```
import ipaddress
ip = int(ipaddress.IPv4Address(“193.219.32.13”);
```

int konversija į IP adresą:

```
import ipaddress
ip = str(ipaddress.IPv4Address(21425436532);
```

Interneto domenų klasifikacijos. Internete egzistuoja nemažai svetainių, kuriose domenai klasifikuojami pagal populiarumą, t.y. reitinguojami. Viena iš populiariausių ir žinomiausių svetainių yra https://www.alexa.com/topsites Joje domenai yra reitinguojami globaliai, o taip pat pagal atskiras šalis. Pateikiami 500 populiariausių domenų globaliai pasaulio mastu o taip pat atskirai pagal kiekvieną šalį. Tačiau patogus API, kuriuo naudojantis būtų galima gauti reitingus JSON, CSV ar kokiu nors kitu formatu yra mokamas.

Taip pat egzistuoja ir kiti domenų reitingai, kurie yra nemokami ir kurie pateikiami patogiu suarchyvuotu CSV formato failu. Tokių reitingų failų adresai yra:

- Umbrella Top 1 mln (http://s3-us-west-1.amazonaws.com/umbrella-static/top-1m.csv.zip)
- Tranco Top 1 mln (https://tranco-list.eu/download_daily/99LZ2 , tačiau adresas gali keistis, naujausią adresą galima pasižiūrėti svetainėje https://tranco-list.eu/)
- Majectic Top 1 mln (https://downloads.majestic.com/majestic_million.csv)

Juose duomenys saugomi csv formatu: *<reitingo_vieta,domenas>*. Pvz:

```
1,google.com
2,amazonaws.com
3,facebook.com
4,microsoft.com
```

Arba tokiu formatu (Majectic):

```
GlobalRank,TldRank,Domain,TLD,RefSubNets,RefIPs,IDN_Domain,IDN_TLD,PrevGlobalRank,Pre
vTldRank,PrevRefSubNets,PrevRefIPs
1,1,google.com,com,477410,2238155,google.com,com,1,1,476626,2229677
2,2,facebook.com,com,471729,2328030,facebook.com,com,2,2,470873,2318648
3,3,youtube.com,com,425986,1912609,youtube.com,com,3,3,425083,1904269
4,4,twitter.com,com,414073,1839962,twitter.com,com,4,4,413210,1833029
```

IP adresų pasiskirstymo pagal šalis geografinė duomenų bazė. Interneto svetainėje db-ip.com yra pateikiama ir reguliariai atnaujinama IP adresų pasiskirstymo pagal šalis geografinė duomenų bazė. Nemokama jos versija yra supaprastintas pilnos duomenų bazės poaibis su sumažintu tikslumu (angl. IP to Country Lite geolocation database), kuri pateikiama adresu: https://db-ip.com/db/download/ip-to-country-lite

Duomenų bazė pateikiama kaip suarchyvuotas CSV failas, kurio formatas yra: *<IP rėžio pradžia,IP rėžio pabaiga,šalies kodas>*

```
223.255.248.0,223.255.251.255,HK
223.255.252.0,223.255.253.255,CN
223.255.254.0,223.255.254.255,SG
```

223.255.255.0,223.255.255.255,AU

Interneto paslaugos. Interneto tinklu yra teikiama daug ir įvairių paslaugų. Tokios paslaugos paprastai teikiamos naudojantis REST API, o vienas iš populiariausių interneto paslaugų duomenų teikimo formatas yra JSON. Tai reiškia, kad klientas atlieka REST užklausą ir gauna atsakymą JSON formatu.

Patogi nemokama interneto paslauga yra orų paslauga: http://weatherapi.com Atlikus GET užklausą pateikiami orų duomenys pagal vietovę JSON formatu. Užklausos pavyzdys:

http://api.weatherapi.com/v1/current.json?key=2d351b25a5ec46fab0694510221411&q=Kaunas

Užklausoje naudojami parametrai:

Užklausos parametras	Reikšmė	Apibūdinimas
key	2d351b25a5ec46fab0694510221411	API raktas, kuris gaunamas užsiregistravus paslaugos svetainėje
q	Kaunas	Miestas, ar vietovė, kurios orus norima gauti.

Užklausos rezultatas JSON formatu:


```
{
    "current": {
        "cloud": 100,
        "condition": {
            "code": 1009,
            "icon": "//cdn.weatherapi.com/weather/64x64/night/122.png",
            "text": "Overcast"
        },
        "feelslike_c": -13.8,
        "feelslike_f": 7.2,
        "gust_kph": 9.0,
        "gust_mph": 5.6,
        "humidity": 79,
        "is_day": 0,
        "last_updated": "2021-12-07 16:45",
        "last_updated_epoch": 1638888300,
        "precip_in": 0.0,
        "precip_mm": 0.0,
        "pressure_in": 30.09,
        "pressure_mb": 1019.0,
        "temp_c": -10.0,
        "temp_f": 14.0,
        "uv": 2.0,
        "vis_km": 10.0,
        "vis_miles": 6.0,
        "wind_degree": 12,
        "wind_dir": "NNE",
        "wind_kph": 0.0,
        "wind_mph": 0.0
    },
    "location": {
        "country": "Lithuania",
        "lat": 54.9,
        "localtime": "2021-12-07 16:52",
        "localtime_epoch": 1638888740,
        "lon": 23.9,
        "name": "Kaunas",
        "region": "Kauno Apskritis",
        "tz_id": "Europe/Vilnius"
    }
}
```


Modulis *requests*. Modulis importuojamas:

```
import request
```

1) GET užklausos atlikimas:

```
requests.get(url, params=None, **kwargs)
```

Parametrai:

- *url* – url adresas.
- *params* – žodynas, kortežų ar baitų sąrašas, kurie siunčiami su užklausa (nebūtina).
- ***kwargs* – nebūtini argumentai

Grąžina: requests.Response

Pvz.:

```
parameters = {'address':location, 'key':"somekey"}
r = requests.get(url = google_url, params = parameters)
data = r.json()
```

2) Užklausos atsakymo kodo tikrinimas:

```
r = requests.get(url)
if r.status_code != 200:
    print ("Rezultai yra nepasiekiami: ", r)
    print (json.dumps(r.json(), indent=4, sort_keys=True))
    sys.exit()
```

3) Url adresu pasiekiamo zip failo skaitymas:

```
from zipfile import ZipFile
from io import BytesIO
filename = requests.get(url).content
zf = ZipFile( BytesIO(filename), 'r' )
for item in zf.namelist():  # archyve gali būti ne vienas failas
    file = zf.open(item)
```

Modulis *csv*. Modulis importuojamas:

```
import csv
```

1) CSV failo skaitymas:

```
from io import TextIOWrapper
csvreader = csv.reader(TextIOWrapper(file, 'utf-8'))
    for row in csvreader: # row – sąrašas, galima naudoti row[0], row[1]
        rows.append(row)
```

Modulis *json*. Modulis importuojamas:

```
import json
```

1) Json duomenų objekto konversija į string eilutę:

```
json.dumps(data, indent=4, sort_keys=True)
```

Parametrai:

- *data* – json duomenys.
- *indent* – json identacija.
- *sort-keys* – ar raktai bus rikiuojami.

Grąžina: json eilutė

Pvz.:

```
print(json.dumps(data, indent=4, sort_keys=True))
```

2) Json duomenų laukelių gavimas iš JSON duomenų objekto:

```
print(data);
```

```
{
  "current": {
    "value1": 100,
    "value2": 14.0
  },
}
```

```
data1 = data['current']['value1']
data2 = data['current']['value2']
print (str(data1), str(data2))
```

Modulis *tarfile*. Modulis importuojamas:

```
import tarfile
```

1) Gzip archyvuoto failo skaitymas:

```
file = tarfile.open(file_path, "r:gz")
for filename in file.getmembers():  # archyve gali būti ne vienas failas
    f = file.extractfile(filename)
```

Modulis *ipaddress*. Modulis importuojamas:

```
import ipaddress
```

1) IP adreso konversija į int:

```
ip = int(ipaddress.IPv4Address("193.219.32.13"))
```

2) int konversija į IP adresą:

```
ip = str(ipaddress.IPv4Address(3252363277))
```

Ipv4 adreso konversija į int. Iš pradžių ip adresas (pvz. 193.219.32.13) suskaidomas į 4 skaičius. Po taikoma formulė:

```
(pirmas_skaicius * 256³) + (antras_skaicius * 256²) + (trecias_skaicius * 256) + (ketvirtas
skaičius) = (193 * 16777216) + (219 * 65536) + (32 * 256) + (13) = 3252363277
```

Modulis *sys*. Modulis importuojamas:

```
import sys
```

1) Argumentų gavimas:

```
import sys
print ("Argumentų sąrašas: ", str(sys.argv))
print ("Pirmas argumentas: ", sys.argv[0])
```

Užduotys

Serverio adresas: **158.129.0.113**. Prisijungimas per *putty* arba *Secure Shell Client*. Vartotojo vardas – *varpava*. Scenarijaus failo pradžioje reikia nurodyti kelią iki python interpretatoriaus:

```
#!/usr/bin/python3
```

Užduotis1. Parašykite scenarijų *uzd13.py*, kuris pasinaudojant http://api.weatherapi.com pateikiama sąsaja (angl. API) nustatytu kokia šiuo metu yra vietovės temperatūra, ir kada ši temperatūra paskutinį kartą buvo atnaujinta. Vietovė įvedama pirmu argumentu. Temperatūros adresas API json eilutėje: /current/temp_c, temperatūros atnaujinimo datos adresas: /current/last_updated. Naudokite API raktą: 2d351b25a5ec46fab0694510221411, o jeigu jis neveikia – susikurkite nuosavą.

Užduotis2. Parašykite scenarijų *uzd14.py*, kuris nustatytų pirmu argumentu įvesto domeno reitingus Tranco Top 1 mln ir Umbrella Top 1 mln domenų reitingų duomenų bazėse. Užduočiai atlikti galima csv failus nuskaityti į python žodyną (dictionary), kur raktas yra domenas, o reikšmė – reitingas. Tada atlikti paiešką žodyne pagal raktą.

Užduotis3. Parašykite scenarijų *uzd15.py*, kuris naudodamasis IP adresų pasiskirstymo pagal šalis geografine duomenų baze nustatytu pirmu argumentu įvesto IPv4 adreso šalį. Supaprastinta IPv4 adresų duomenų bazė pateikta faile */home/stud/stud/dbip-country-lite-2021-12.csv.tar.gz*. IP adresų palyginimo tikslu patogu IP adresų string eilutes paversti sveikaisiais int skaičiais. Užduočiai atlikti galima csv failą nuskaityti į dvimatį sąrašą (masyvą), po to atlikti nuoseklią paiešką sąraše.