Šiame laboratoriniame darbe kartu išnagrinėsime ir panaudosime 2 API paslaugas, mums reikia API grąžinančio geografines koordinates ir kito API, gražinančio orų prognozę pagal koordinates.

Geoxyz API

radome tokį API, grąžinantį geo koordinates. Žingsnis po žingsnio bandome jį pakalbinti ir pasižiūrėti jo grąžinamų duomenų formatus.

In [52]:

```
import requests
import time
import json

paieskos_fraze = 'Kaunas'

API_raktas = "4292875920476927253373x35374"
# 'https://geocode.xyz/Vilnius?json=1&auth=API_key'
get_parametrai = {'json':'1', 'key': API_raktas}
time.sleep(2)
r = requests.get(f'https://geocode.xyz/{paieskos_fraze}', params=get_parametrai)
```

In [53]:

```
r
```

Out[53]:

```
<Response [200]>
```

In [54]:

```
r.text
```

Out[54]:

```
'{   "standard" : {      "addresst" : {},      "statename" : {},      "city"
: "Kaunas",      "prov" : "LT",      "countryname" : "Lithuania",      "post
al" : {},      "confidence" : "0.90"   },   "longt" : "23.91697",   "alt" :
{      "loc" : [         {            "longt" : "23.9137990434782",
"prov" : "LT",            "city" : "Kaunas",            "postal" : "48442",
"score" : "230",            "latt" : "54.9504823478261"         },         {
"longt" : "23.91380",            "prov" : "LT",            "city" : "Kauna
s",            "countryname" : "Lithuania",            "postal" : "48442",
"region" : {},            "latt" : "54.95048"         }      ]   },   "eleva
tion" : {},   "remaining_credits" : "-32",   "latt" : "54.89710"}'
```

In [55]:

```
geo_d = json.loads(r.text)
```

In [56]:

```
1 geo_d
```

Out[56]:


```
{'standard': {'addresst': {},
  'statename': {},
  'city': 'Kaunas',
  'prov': 'LT',
  'countryname': 'Lithuania',
  'postal': {},
  'confidence': '0.90'},
 'longt': '23.91697',
 'alt': {'loc': [{'longt': '23.9137990434782',
    'prov': 'LT',
    'city': 'Kaunas',
    'postal': '48442',
    'score': '230',
    'latt': '54.9504823478261'},
   {'longt': '23.91380',
    'prov': 'LT',
    'city': 'Kaunas',
    'countryname': 'Lithuania',
```


šiame žingsnyje mes jau radome kur randasi reikalingos koordinatės, tai bus geo_d['latt'] - platuma ir geo_d['longt'] - ilguma

In [ ]:

```
1 latt_ = geo_d['latt']
```

In [58]:

```
1 longt_ = geo_d['longt']
2 print(latt_, longt_)
```

```
54.89710 23.91697
```

In [63]:

```
1 type(latt_)
```

Out[63]:

```
str
```

https://open-meteo.com/en (https://open-meteo.com/en)

---

suradę tokį API, grąžinantį orus pagal koordinates, pabandome pakalbinti ir jį.

In [77]:

```python
# https://api.open-meteo.com/v1/forecast?latitude=52.52&longitude=13.41&hourly=temperat
METEO_END_POINT = "https://api.open-meteo.com/v1/forecast"

def meteo(lat, lon, end_point=METEO_END_POINT,):
    get_parametrai = {'latitude': lat, 'longitude': lon, 'hourly': 'temperature_2m'}
    time.sleep(2)
    r = requests.get(end_point, params=get_parametrai)
    if r.ok:
        return r.text
    else:
        raise Exception("meteo func failed")
```

In [78]:

```
1 meteo(latt_, longt_)
```

Out[78]:

'{"latitude":54.875,"longitude":23.9375,"generationtime_ms":0.30303001403808
594,"utc_offset_seconds":0,"timezone":"GMT","timezone_abbreviation":"GMT","e
levation":36.0,"hourly_units":{"time":"iso8601","temperature_2m":"°C"},"hour
ly":{"time":["2022-08-12T00:00","2022-08-12T01:00","2022-08-12T02:00","2022-
08-12T03:00","2022-08-12T04:00","2022-08-12T05:00","2022-08-12T06:00","2022-
08-12T07:00","2022-08-12T08:00","2022-08-12T09:00","2022-08-12T10:00","2022-
08-12T11:00","2022-08-12T12:00","2022-08-12T13:00","2022-08-12T14:00","2022-
08-12T15:00","2022-08-12T16:00","2022-08-12T17:00","2022-08-12T18:00","2022-
08-12T19:00","2022-08-12T20:00","2022-08-12T21:00","2022-08-12T22:00","2022-
08-12T23:00","2022-08-13T00:00","2022-08-13T01:00","2022-08-13T02:00","2022-
08-13T03:00","2022-08-13T04:00","2022-08-13T05:00","2022-08-13T06:00","2022-
08-13T07:00","2022-08-13T08:00","2022-08-13T09:00","2022-08-13T10:00","2022-
08-13T11:00","2022-08-13T12:00","2022-08-13T13:00","2022-08-13T14:00","2022-
08-13T15:00","2022-08-13T16:00","2022-08-13T17:00","2022-08-13T18:00","2022-
08-13T19:00","2022-08-13T20:00","2022-08-13T21:00","2022-08-13T22:00","2022-
08-13T23:00","2022-08-14T00:00","2022-08-14T01:00","2022-08-14T02:00","2022-
08-14T03:00","2022-08-14T04:00","2022-08-14T05:00","2022-08-14T06:00","2022-
08-14T07:00","2022-08-14T08:00","2022-08-14T09:00","2022-08-14T10:00","2022-
08-14T11:00","2022-08-14T12:00","2022-08-14T13:00","2022-08-14T14:00","2022-
08-14T15:00","2022-08-14T16:00","2022-08-14T17:00","2022-08-14T18:00","2022-
08-14T19:00","2022-08-14T20:00","2022-08-14T21:00","2022-08-14T22:00","2022-
08-14T23:00","2022-08-15T00:00","2022-08-15T01:00","2022-08-15T02:00","2022-
08-15T03:00","2022-08-15T04:00","2022-08-15T05:00","2022-08-15T06:00","2022-
08-15T07:00","2022-08-15T08:00","2022-08-15T09:00","2022-08-15T10:00","2022-
08-15T11:00","2022-08-15T12:00","2022-08-15T13:00","2022-08-15T14:00","2022-
08-15T15:00","2022-08-15T16:00","2022-08-15T17:00","2022-08-15T18:00","2022-
08-15T19:00","2022-08-15T20:00","2022-08-15T21:00","2022-08-15T22:00","2022-
08-15T23:00","2022-08-16T00:00","2022-08-16T01:00","2022-08-16T02:00","2022-
08-16T03:00","2022-08-16T04:00","2022-08-16T05:00","2022-08-16T06:00","2022-
08-16T07:00","2022-08-16T08:00","2022-08-16T09:00","2022-08-16T10:00","2022-
08-16T11:00","2022-08-16T12:00","2022-08-16T13:00","2022-08-16T14:00","2022-
08-16T15:00","2022-08-16T16:00","2022-08-16T17:00","2022-08-16T18:00","2022-
08-16T19:00","2022-08-16T20:00","2022-08-16T21:00","2022-08-16T22:00","2022-
08-16T23:00","2022-08-17T00:00","2022-08-17T01:00","2022-08-17T02:00","2022-
08-17T03:00","2022-08-17T04:00","2022-08-17T05:00","2022-08-17T06:00","2022-
08-17T07:00","2022-08-17T08:00","2022-08-17T09:00","2022-08-17T10:00","2022-
08-17T11:00","2022-08-17T12:00","2022-08-17T13:00","2022-08-17T14:00","2022-
08-17T15:00","2022-08-17T16:00","2022-08-17T17:00","2022-08-17T18:00","2022-
08-17T19:00","2022-08-17T20:00","2022-08-17T21:00","2022-08-17T22:00","2022-
08-17T23:00","2022-08-18T00:00","2022-08-18T01:00","2022-08-18T02:00","2022-
08-18T03:00","2022-08-18T04:00","2022-08-18T05:00","2022-08-18T06:00","2022-
08-18T07:00","2022-08-18T08:00","2022-08-18T09:00","2022-08-18T10:00","2022-
08-18T11:00","2022-08-18T12:00","2022-08-18T13:00","2022-08-18T14:00","2022-
08-18T15:00","2022-08-18T16:00","2022-08-18T17:00","2022-08-18T18:00","2022-
08-18T19:00","2022-08-18T20:00","2022-08-18T21:00","2022-08-18T22:00","2022-
08-18T23:00"],"temperature_2m":[15.0,14.8,14.6,14.4,14.6,16.5,19.5,22.0,24.
3,25.8,26.7,27.5,28.1,28.4,28.5,28.3,27.8,26.5,24.5,22.6,21.4,20.0,18.9,18.
0,17.2,16.6,16.1,15.7,15.8,17.9,20.6,22.8,25.0,26.6,27.5,28.1,28.7,29.0,29.
0,28.8,28.1,26.7,24.7,22.7,21.3,20.3,19.6,18.8,18.3,17.7,17.0,16.4,16.8,18.
5,20.6,22.7,24.5,25.6,26.8,28.6,29.9,30.6,30.7,30.2,29.5,27.9,26.8,25.5,24.
3,23.6,23.0,22.4,21.7,21.0,20.6,20.2,20.5,21.3,23.0,24.9,26.9,28.3,29.3,30.
6,30.8,30.7,30.1,29.1,28.2,27.2,25.9,25.1,24.3,23.3,22.7,22.0,21.4,20.9,20.
5,20.5,21.0,21.8,23.2,24.5,26.2,28.0,28.9,29.6,30.1,30.3,30.3,29.9,29.0,27.
7,26.1,25.1,24.1,22.9,22.2,21.5,20.6,19.8,19.0,18.5,19.2,20.4,21.7,23.4,24.
7,25.8,25.6,24.8,23.8,23.4,23.2,22.9,22.5,22.1,21.5,21.1,20.6,20.0,19.5,19.

1,18.7,18.6,18.7,19.1,19.5,20.2,21.2,22.1,23.1,24.4,25.2,26.0,26.3,25.7,24.
6,23.2,22.5,21.9,21.0,20.1,19.2,18.0,17.2,16.5]}}'

In [61]:

```
1 r.url
```

Out[61]:

'https://api.open-meteo.com/v1/forecast?latitude=54.89710&longitude=23.91697
&hourly=temperature_2m'

In [62]:

```
1 r.text
```

Out[62]:

'{"latitude":54.875,"longitude":23.9375,"generationtime_ms":0.28204917907714
844,"utc_offset_seconds":0,"timezone":"GMT","timezone_abbreviation":"GMT","e
levation":36.0,"hourly_units":{"time":"iso8601","temperature_2m":"°C"},"hour
ly":{"time":["2022-08-12T00:00","2022-08-12T01:00","2022-08-12T02:00","2022-
08-12T03:00","2022-08-12T04:00","2022-08-12T05:00","2022-08-12T06:00","2022-
08-12T07:00","2022-08-12T08:00","2022-08-12T09:00","2022-08-12T10:00","2022-
08-12T11:00","2022-08-12T12:00","2022-08-12T13:00","2022-08-12T14:00","2022-
08-12T15:00","2022-08-12T16:00","2022-08-12T17:00","2022-08-12T18:00","2022-
08-12T19:00","2022-08-12T20:00","2022-08-12T21:00","2022-08-12T22:00","2022-
08-12T23:00","2022-08-13T00:00","2022-08-13T01:00","2022-08-13T02:00","2022-
08-13T03:00","2022-08-13T04:00","2022-08-13T05:00","2022-08-13T06:00","2022-
08-13T07:00","2022-08-13T08:00","2022-08-13T09:00","2022-08-13T10:00","2022-
08-13T11:00","2022-08-13T12:00","2022-08-13T13:00","2022-08-13T14:00","2022-
08-13T15:00","2022-08-13T16:00","2022-08-13T17:00","2022-08-13T18:00","2022-
08-13T19:00","2022-08-13T20:00","2022-08-13T21:00","2022-08-13T22:00","2022-
08-13T23:00","2022-08-14T00:00","2022-08-14T01:00","2022-08-14T02:00","2022-
08-14T03:00","2022-08-14T04:00","2022-08-14T05:00","2022-08-14T06:00","2022-
08-14T07:00","2022-08-14T08:00","2022-08-14T09:00","2022-08-14T10:00","2022-
08-14T11:00","2022-08-14T12:00","2022-08-14T13:00","2022-08-14T14:00","2022-
08-14T15:00","2022-08-14T16:00","2022-08-14T17:00","2022-08-14T18:00","2022-
08-14T19:00","2022-08-14T20:00","2022-08-14T21:00","2022-08-14T22:00","2022-
08-14T23:00","2022-08-15T00:00","2022-08-15T01:00","2022-08-15T02:00","2022-
08-15T03:00","2022-08-15T04:00","2022-08-15T05:00","2022-08-15T06:00","2022-
08-15T07:00","2022-08-15T08:00","2022-08-15T09:00","2022-08-15T10:00","2022-
08-15T11:00","2022-08-15T12:00","2022-08-15T13:00","2022-08-15T14:00","2022-
08-15T15:00","2022-08-15T16:00","2022-08-15T17:00","2022-08-15T18:00","2022-
08-15T19:00","2022-08-15T20:00","2022-08-15T21:00","2022-08-15T22:00","2022-
08-15T23:00","2022-08-16T00:00","2022-08-16T01:00","2022-08-16T02:00","2022-
08-16T03:00","2022-08-16T04:00","2022-08-16T05:00","2022-08-16T06:00","2022-
08-16T07:00","2022-08-16T08:00","2022-08-16T09:00","2022-08-16T10:00","2022-
08-16T11:00","2022-08-16T12:00","2022-08-16T13:00","2022-08-16T14:00","2022-
08-16T15:00","2022-08-16T16:00","2022-08-16T17:00","2022-08-16T18:00","2022-
08-16T19:00","2022-08-16T20:00","2022-08-16T21:00","2022-08-16T22:00","2022-
08-16T23:00","2022-08-17T00:00","2022-08-17T01:00","2022-08-17T02:00","2022-
08-17T03:00","2022-08-17T04:00","2022-08-17T05:00","2022-08-17T06:00","2022-
08-17T07:00","2022-08-17T08:00","2022-08-17T09:00","2022-08-17T10:00","2022-
08-17T11:00","2022-08-17T12:00","2022-08-17T13:00","2022-08-17T14:00","2022-
08-17T15:00","2022-08-17T16:00","2022-08-17T17:00","2022-08-17T18:00","2022-
08-17T19:00","2022-08-17T20:00","2022-08-17T21:00","2022-08-17T22:00","2022-
08-17T23:00","2022-08-18T00:00","2022-08-18T01:00","2022-08-18T02:00","2022-
08-18T03:00","2022-08-18T04:00","2022-08-18T05:00","2022-08-18T06:00","2022-
08-18T07:00","2022-08-18T08:00","2022-08-18T09:00","2022-08-18T10:00","2022-
08-18T11:00","2022-08-18T12:00","2022-08-18T13:00","2022-08-18T14:00","2022-
08-18T15:00","2022-08-18T16:00","2022-08-18T17:00","2022-08-18T18:00","2022-
08-18T19:00","2022-08-18T20:00","2022-08-18T21:00","2022-08-18T22:00","2022-
08-18T23:00"],"temperature_2m":[15.0,14.8,14.6,14.4,14.6,16.5,19.1,21.5,23.
6,25.1,26.2,27.0,27.6,27.9,27.9,27.7,27.2,25.9,23.9,22.1,21.2,19.9,18.9,18.
2,17.3,16.8,16.2,15.8,15.9,17.9,20.3,22.4,24.3,25.9,26.7,27.4,28.0,28.3,28.
4,28.0,27.3,26.1,24.2,22.2,20.9,19.8,19.5,18.8,18.1,17.6,17.1,16.9,17.3,18.
8,20.5,22.0,23.2,24.6,25.9,27.4,28.1,28.8,28.9,28.6,27.9,27.0,25.3,24.0,23.
2,22.3,21.7,21.3,20.9,20.8,20.5,20.2,20.5,21.2,22.6,24.0,25.6,27.4,28.5,29.
3,29.9,29.6,28.9,27.7,27.0,26.1,25.0,24.3,23.6,22.7,22.1,21.6,20.9,20.4,19.
9,19.8,20.3,21.1,22.7,24.2,26.0,28.0,28.9,29.4,29.8,30.0,30.0,29.5,28.7,27.
5,25.9,24.8,23.7,22.4,21.5,20.8,20.1,19.3,18.7,18.5,19.2,20.4,22.1,23.4,24.
7,25.8,25.6,24.8,23.8,23.4,23.2,22.9,22.5,22.1,21.5,21.1,20.6,20.0,19.5,19.

```
1,18.7,18.6,18.7,19.1,19.5,20.2,21.2,22.1,23.1,24.4,25.2,26.0,26.3,25.7,24.
6,23.2,22.5,21.9,21.0,20.1,19.2,18.0,17.2,16.5]}}'
```

In [38]:

```
1 res_d = json.loads(r.text)
```

In [ ]:

```
1 res_d
```

In [ ]:

```
1 res_d['hourly']
```

In [44]:

```
1 print(res_d['hourly']['time'][:5])
```

```
['2022-08-12T00:00', '2022-08-12T01:00', '2022-08-12T02:00', '2022-08-12T03:
00', '2022-08-12T04:00']
```

In [45]:

```
1 print(res_d['hourly']['temperature_2m'][:5])
```

```
[14.4, 14.4, 14.3, 14.1, 14.1]
```

In [ ]:

```
1 print(res_d['hourly'])
```

supratome, kad valandos ir temperatūrų prognozė joms randasi 2-juose listuose ir duomenys siejami pagal listų indeksus. Tai ir viskas ko reikėjo.

Toliau savarankiška užduotis. Rašome programą(arba funkcijas, vartotojo įvestis nebūtina) sujungiančią abu API, kuri įvedus miestus ir valandas grąžintų ir atprintintų orų prognozę nurodytam valandų skaičiui.

ĮVYKDYTA UŽDUOTIS, MARIAUS VARIANTAS

In [2]:

```python
from datetime import datetime
import requests
import json
import time

API_key = "66842470606826897881 8x70951"
COORD_LINK = "https://geocode.xyz/"
COORD_PAYLOAD = {"json": "1", "key": API_key}
WEATHER_LINK = "https://api.open-meteo.com/v1/forecast"

def koordinaciu_uzklausa(valandos, *miestai, link=COORD_LINK, param=COORD_PAYLOAD):
    time.sleep(2)
    for i in miestai:
        try:
            r = requests.get(f"{link}{i}", params=param)
            print(i)
            koordinates(r.text, valandos)
        except ConnectionError:
            print("GEOCODE Connection Error")


def koordinates(info, valandos):
    geo_d = json.loads(info)
    latt_ = geo_d["latt"]
    longt_ = geo_d["longt"]
    temperaturos_uzklausa(latt_, longt_, valandos)


def temperaturos_uzklausa(lat, longt, valandos, link=WEATHER_LINK):
    time.sleep(2)
    weather_payload = {"latitude": lat, "longitude": longt, "hourly": "temperature_2m"}
    try:
        r = requests.get(link, params=weather_payload)
        temperaturos(r.text, valandos)
    except ConnectionError:
        print("OPEN-METEO Connection Error")


def temperaturos(info, valandos):
    res = json.loads(info)
    atskaita = res["hourly"]["time"].index(datetime.now().strftime("%Y-%m-%dT%H:00"))
    laikai = res["hourly"]["time"][atskaita : atskaita + valandos]
    temperaturos = res["hourly"]["temperature_2m"][atskaita : atskaita + valandos]
    for date, temp in zip(laikai, temperaturos):
        print(f"\t{date[:10]} {date[11:]}val Temperatūra: {temp}°C")

koordinaciu_uzklausa(10, "Kaunas", "Vilnius", "Klaipeda", "Alytus")
```

```
Kaunas
        2022-08-16 00:00val Temperatūra: 22.3°C
        2022-08-16 01:00val Temperatūra: 21.8°C
        2022-08-16 02:00val Temperatūra: 21.3°C
        2022-08-16 03:00val Temperatūra: 21.1°C
        2022-08-16 04:00val Temperatūra: 21.3°C
        2022-08-16 05:00val Temperatūra: 21.8°C
        2022-08-16 06:00val Temperatūra: 22.9°C
        2022-08-16 07:00val Temperatūra: 24.3°C
        2022-08-16 08:00val Temperatūra: 25.5°C
```

```
        2022-08-16 09:00val Temperatūra: 26.6°C
Vilnius
        2022-08-16 00:00val Temperatūra: 20.3°C
        2022-08-16 01:00val Temperatūra: 20.2°C
        2022-08-16 02:00val Temperatūra: 19.7°C
        2022-08-16 03:00val Temperatūra: 19.4°C
        2022-08-16 04:00val Temperatūra: 19.6°C
        2022-08-16 05:00val Temperatūra: 20.4°C
        2022-08-16 06:00val Temperatūra: 21.4°C
        2022-08-16 07:00val Temperatūra: 22.6°C
        2022-08-16 08:00val Temperatūra: 24.0°C
        2022-08-16 09:00val Temperatūra: 25.7°C
Klaipeda
        2022-08-16 00:00val Temperatūra: 22.5°C
        2022-08-16 01:00val Temperatūra: 22.5°C
        2022-08-16 02:00val Temperatūra: 22.6°C
        2022-08-16 03:00val Temperatūra: 22.6°C
        2022-08-16 04:00val Temperatūra: 22.5°C
        2022-08-16 05:00val Temperatūra: 22.8°C
        2022-08-16 06:00val Temperatūra: 23.5°C
        2022-08-16 07:00val Temperatūra: 24.8°C
        2022-08-16 08:00val Temperatūra: 26.6°C
        2022-08-16 09:00val Temperatūra: 28.1°C
Alytus
        2022-08-16 00:00val Temperatūra: 21.7°C
        2022-08-16 01:00val Temperatūra: 21.3°C
        2022-08-16 02:00val Temperatūra: 20.9°C
        2022-08-16 03:00val Temperatūra: 20.4°C
        2022-08-16 04:00val Temperatūra: 20.5°C
        2022-08-16 05:00val Temperatūra: 21.3°C
        2022-08-16 06:00val Temperatūra: 22.3°C
        2022-08-16 07:00val Temperatūra: 23.1°C
        2022-08-16 08:00val Temperatūra: 24.1°C
        2022-08-16 09:00val Temperatūra: 24.4°C
```

In [ ]:

```
1
```