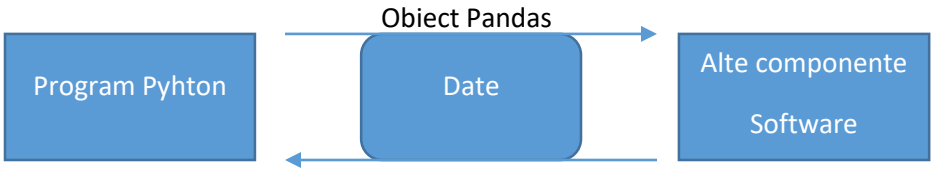
Utilizarea API

Application Program Interface

Un API permite comunicarea intre doua aplicatii. Daca avem un program, date si alte component soft. Prin intermediul API se realizeaza o comunicare prin intputuri si outputuri. In mod asemanator unei functii, fara a cunoaste neaparat modul de functionare al API-ului, acesta poate fi utilizat prin comunicare.

Pandas este o colectie de aplicatii (nu neaparat scrise in Python) prin intermediul carora se poate realiza comunicarea intre program si alte component software.



Instalare pandas:

```
pip install –user pandas
```

Nota: In Google Colab biblioteca *pandas* este deja instalata.

Exemplu de utilizare:

```
import pandas as pd
dict_={'a':[11,21,31],'b':[12,22,32]} #construim un dictionar
df=pd.DataFrame(dict_)
```

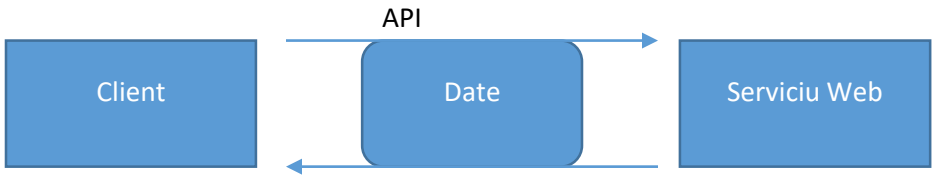
Constructorul DataFrame din API-ul Pandas face ca datele din dictionar sa fie transferate catre data frame pentru a se putea face comunicarea.

Metoda *Head* extrage primele randuri din dataframe.

```
print(df.head())
print(df.mean())# calculeaza media
```

REST API

REST API (Representational State Transfer) sunt utilizate pentru comunicarea cu servicii web si prezinta un set de reguli pnetru Comunicare, Input (Request) si Output (Response).



Solicitarea(request) se transmite sub forma unui mesaj HTTP. Serviciul executa cerinta si returneaza raspunsul in mod similar, de obicei printr-un fisier JSON.

Exemplu utilziare nba_api pentru analiza rezultatelor

Inainte de prima utilizare trebuie instalata biblioteca nba_api

```
pip install –user nba_api
```

In Google Colab utilizam:

```
!pip install nba_api
```

In continuare:

```
from nba_api.stats.static import teams #https://pypi.org/project/nba-
api/
import matplotlib.pyplot as plt

nba_teams = teams.get_teams() # preia echipele

print(nba_teams[0:3])
```

Pentru simplificare putem folosi o functie one_dict:

```
def one_dict(list_dict):
    keys=list_dict[0].keys()
    out_dict={key:[] for key in keys}
```

```
    for dict_ in list_dict:
        for key, value in dict_.items():
            out_dict[key].append(value)
    return out_dict
```

Aceasta transforma dictionarul intr-o tabela, urmand ca prin intermediul DataFrame datele sa fie transformate intr-un data frame.

```
dict_nba_team=one_dict(nba_teams)
df_teams=pd.DataFrame(dict_nba_team)
print(df_teams.head())
```

Prin utilizarea campului “nickname” este gasit identificatorul echipei Warriors.

```
df_warriors=df_teams[df_teams['nickname']=='Warriors']
print(df_warriors)

id_warriors=df_warriors[['id']].values[0][0]
print(id_warriors)
```

In continuare solicitam prin intermediul functiei League Game Finder extragerea tuturor jocurilor echipei Warriors.

```
from nba_api.stats.endpoints import leaguegamefinder
gamefinder =
leaguegamefinder.LeagueGameFinder(team_id_nullable=id_warriors)
```

Prin intermediul functiei get_data_frames() obtinem informatii legate de datele importate.

```
games = gamefinder.get_data_frames()[0]
print(games.head())
```

Campul Plus_minus defineste diferenta de puncte pentru fiecare joc (- pt jocurile pierdute, + pt jocurile castigate), iar Matchup se refera la echipa impotriva careia a jucat.

Pentru a extrage informatiile legate de meciurile jucate impotriva echipei Raptors, ne folosim de urmatoarele instructiuni pentru a crea doua data-frame-uri.

```
games_home=games [games ['MATCHUP']=='GSW vs. TOR'] #meciuri acasa
games_away=games [games ['MATCHUP']=='GSW @ TOR'] #meciuri in deplasare
```

In continuare calculam media diferentelor de puncte:

```
games_home.mean()['PLUS_MINUS']
games_away.mean()['PLUS_MINUS']
```

In final afisam intr-un grafic coloana Plus_Minus pentru cele doua data-frame-uri construite.

```
fig, ax = plt.subplots()

games_away.plot(x='GAME_DATE',y='PLUS_MINUS', ax=ax)
games_home.plot(x='GAME_DATE',y='PLUS_MINUS', ax=ax)
ax.legend(["away", "home"])
plt.show()
```

Exercitii:

1. Creati data-frame-uri si pt echipa “Cavaliers”.
2. Afisati pe grafic si informatiile Cavaliers vs Raptors.
3. Citiţi mai multe despre pachetul pandas: https://pandas.pydata.org/
4. Studiati biblioteca nba_api pentru a cunoaste mai multe facilitati disponibile: https://pypi.org/project/nba-api/