Universitatea Tehnică a Moldovei

Facultatea Calculatoare Informatică şi Microelectronică

Departamentul Ingineria Software și Automatică

**RAPORT**

Lucrarea de laborator nr. 6

# La disciplina „Programarea Declarativa”

Tema: “Manipularea datelor folosind limbajul Python”

A efectuat: st. gr. SI-211 A. Chihai

A verificat: V. Rusu

Chișinău – 2023

Lucrarea de laborator nr. 6

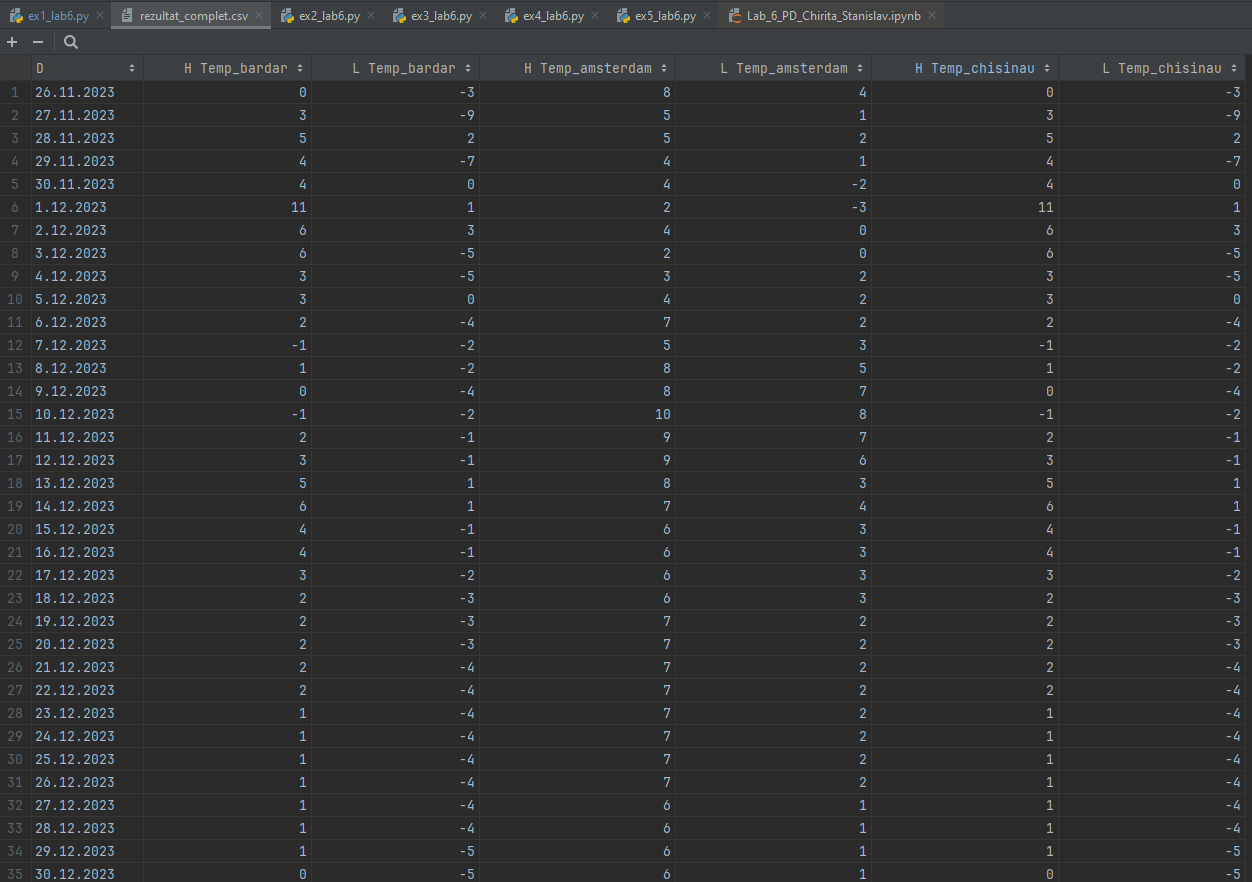
#### Sarcina de realizat:

1. Scrieți secvența de cod care regrupeză datele din cele 3 fișiere csv (obținute la laboratul

precedent) după o caracteristică de bază (de ex: cursul valutar de la 3 bănci pentru aceeași

zi).

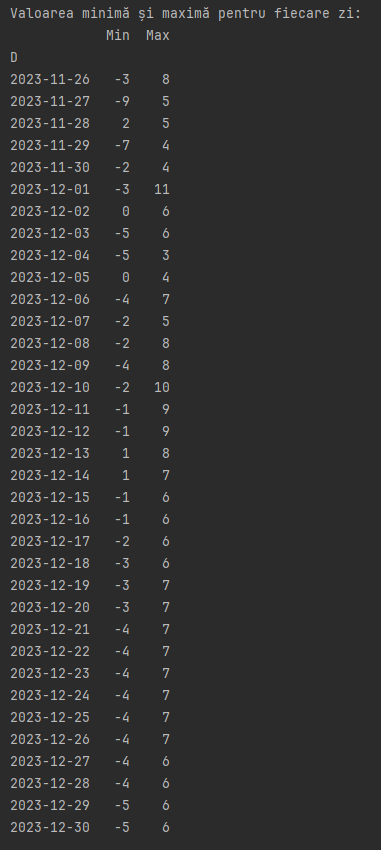
import pandas as pd  
  
def concatenate\_all():  
 bardar = pd.read\_csv('bardar\_weather.csv')  
 amsterdam = pd.read\_csv('amsterdam\_weather.csv')  
 chisinau = pd.read\_csv('chisinau\_weather.csv')  
  
 bardar.drop\_duplicates(subset=['D'], inplace=True)  
 amsterdam.drop\_duplicates(subset=['D'], inplace=True)  
 chisinau.drop\_duplicates(subset=['D'], inplace=True)  
  
 bardar.set\_index('D', inplace=True)  
 amsterdam.set\_index('D', inplace=True)  
 chisinau.set\_index('D', inplace=True)  
  
 bardar = bardar.apply(pd.to\_numeric, errors='coerce')  
 amsterdam = amsterdam.apply(pd.to\_numeric, errors='coerce')  
 chisinau = chisinau.apply(pd.to\_numeric, errors='coerce')  
  
 df\_regrupat = pd.merge(bardar, amsterdam, left\_index=True, right\_index=True, suffixes=('\_bardar', '\_amsterdam'))  
 df\_regrupat = pd.merge(df\_regrupat, chisinau.add\_suffix('\_chisinau'), left\_index=True, right\_index=True)  
  
 df\_regrupat.reset\_index(inplace=True)  
  
 pd.set\_option('display.max\_rows', None)  
 pd.set\_option('display.max\_columns', None)  
 print(df\_regrupat)  
  
 df\_regrupat.to\_csv('rezultat\_complet.csv', index=False)  
  
concatenate\_all()

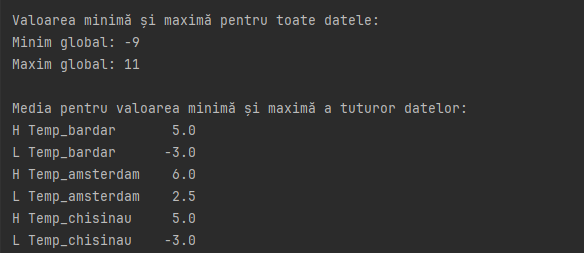


2. Determinați min, max, media pentru valorile regrupate pentru fiecare zi aparte și min, max

media pentru toate datele stocate.

import pandas as pd  
  
df = pd.read\_csv('rezultat\_complet.csv', delimiter=',')  
  
df['D'] = pd.to\_datetime(df['D'], format='%d.%m.%Y')  
  
df.set\_index('D', inplace=True)  
  
daily\_min = df.min(axis=1)  
daily\_max = df.max(axis=1)  
  
daily\_results = pd.DataFrame({'Min': daily\_min, 'Max': daily\_max})  
  
overall\_min = df.min().min()  
overall\_max = df.max().max()  
  
overall\_avg\_min\_max = df[['H Temp\_bardar', 'L Temp\_bardar', 'H Temp\_amsterdam', 'L Temp\_amsterdam', 'H Temp\_chisinau', 'L Temp\_chisinau']].agg(['min', 'max']).mean()  
  
print("Valoarea minimă și maximă pentru fiecare zi:")  
print(daily\_results)  
  
print("\nValoarea minimă și maximă pentru toate datele:")  
print(f"Minim global: {overall\_min}")  
print(f"Maxim global: {overall\_max}")  
  
print("\nMedia pentru valoarea minimă și maximă a tuturor datelor:")  
print(overall\_avg\_min\_max)



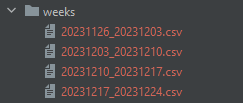


3. Scrieți secvența de cod care va împărți fișierul csv obținut după concatinare în N fișiere,

unde fiecare fișier individual va corespunde unei săptămâini (7 rânduri). Fișierele sunt

denumite după prima și ultima dată pe care le conțin (de ex 20231103\_20231109.csv).

import pandas as pd  
import os  
  
df = pd.read\_csv('rezultat\_complet.csv', delimiter=',')  
  
df['D'] = pd.to\_datetime(df['D'], format='%d.%m.%Y')  
  
df.set\_index('D', inplace=True)  
  
date\_ranges = pd.date\_range(start=df.index.min(), end=df.index.max(), freq='7D')  
  
output\_directory = 'weeks'  
  
if not os.path.exists(output\_directory):  
 os.makedirs(output\_directory)  
  
for i in range(len(date\_ranges) - 1):  
 start\_date = date\_ranges[i]  
 end\_date = date\_ranges[i + 1]  
  
 week\_data = df[(df.index >= start\_date) & (df.index < end\_date)]  
  
 if not week\_data.empty:  
 file\_path = os.path.join(output\_directory, f"{start\_date.strftime('%Y%m%d')}\_{end\_date.strftime('%Y%m%d')}.csv")  
 week\_data.to\_csv(file\_path, sep='\t')  
  
print("Fișierele au fost create cu succes în directorul 'weeks'.")

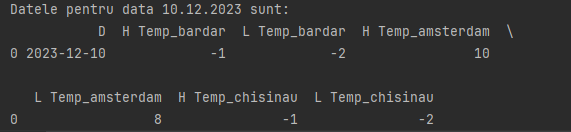


4. Scrieți funcția ce are o dată de intrare tip datetime și returnează liniile din DataFrame (din

fișierul cu date concatinate) pentru această dată sau None dacă nu există date pentru această

dată.

import pandas as pd  
  
def get\_data\_for\_date(target\_date):  
 df = pd.read\_csv('rezultat\_complet.csv', delimiter=',')  
  
 df['D'] = pd.to\_datetime(df['D'], format='%d.%m.%Y')  
  
 result\_data = df[df['D'] == target\_date]  
  
 if not result\_data.empty:  
 return result\_data.reset\_index(drop=True)  
 else:  
 return None  
  
target\_date = pd.to\_datetime('2023-12-10', format='%Y-%m-%d')  
result\_for\_date = get\_data\_for\_date(target\_date)  
  
if result\_for\_date is not None:  
 with pd.option\_context('display.max\_columns', None, 'display.max\_rows', None):  
 print(f"Datele pentru data {target\_date.strftime('%d.%m.%Y')} sunt:")  
 print(result\_for\_date)  
else:  
 print(f"Nu există date pentru data {target\_date.strftime('%d.%m.%Y')}.")



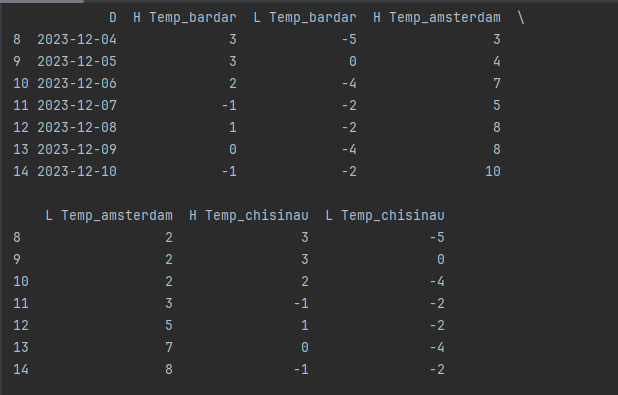


5. Scrieți secvența de cod care permite afișarea datelor pe secvențe de timp. De ex. pentru

fiecare săptămănă separat (funcția are o dată de intrare tip datetime) și returnează liniile

din DataFrame pentru această perioadă sau None dacă nu există date pentru zilele indicate.

import pandas as pd  
  
def display\_data\_for\_weeks(target\_date, df):  
 if 'D' not in df.columns or not pd.api.types.is\_datetime64\_any\_dtype(df['D']):  
 print("Coloana 'D' nu există sau nu este convertită corect la tipul de dată datetime.")  
 return  
  
 start\_of\_week = target\_date - pd.DateOffset(days=target\_date.weekday())  
 end\_of\_week = start\_of\_week + pd.DateOffset(days=6)  
  
 result\_data = df[(df['D'] >= start\_of\_week) & (df['D'] <= end\_of\_week)]  
  
 if not result\_data.empty:  
 with pd.option\_context('display.max\_columns', None, 'display.max\_rows', None):  
 print(f"Datele pentru săptămâna {start\_of\_week.strftime('%U-%Y')} sunt:")  
 print(result\_data)  
 else:  
 print(f"Nu există date pentru săptămâna {start\_of\_week.strftime('%U-%Y')}.")  
  
df = pd.read\_csv('rezultat\_complet.csv', delimiter=',')  
df['D'] = pd.to\_datetime(df['D'], format='%d.%m.%Y')  
  
target\_date = pd.to\_datetime('4.12.2023', format='%d.%m.%Y')  
display\_data\_for\_weeks(target\_date, df)



Concluzie:

În timpul laboratorului nr. 6, am învățat cum să combinăm datele din mai multe surse în Python și să analizăm informațiile. Am descoperit cum să identificăm valorile minime, maxime și să calculăm medii pentru datele noastre. De asemenea, am învățat să segmentăm datele în funcție de intervalul de timp, precum săptămâni. Această experiență ne-a consolidat cunoștințele în prelucrarea datelor cu Python.