МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №8

з дисципліни

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

на тему

Візуалізація та обробка даних за допомогою спеціалізованих бібліотек Python

Виконав:

ст. гр. РІ-21сп

Павлусь В.П

Прийняв:

Шербак С.С.

Львів-2024

**Мета лабораторної роботи:** Розробка додатка для візуалізації CSV-наборів даних за допомогою Matplotlib та базових принципів ООП (наслідування, інкапсуляція, поліморфізм)

**Хід роботи**

**Завдання 1: Вибір CSV-набору даних**

Оберіть CSV-набір даних, який ви хочете візуалізувати. Переконайтеся, що він містить відповідні дані для створення змістовних візуалізацій.

**Завдання 2: Завантаження даних з CSV**

Напишіть код для завантаження даних з CSV-файлу в ваш додаток Python. Використовуйте бібліотеки, такі як Pandas, для спрощення обробки даних.

**Завдання 3: Дослідження даних**

Визначте екстремальні значення по стовцям

**Завдання 4: Вибір типів візуалізацій**

Визначте, які типи візуалізацій підходять для представлення вибраних наборів даних. Зазвичай це може бути лінійні графіки, стовпчикові діаграми, діаграми розсіювання, гістограми та секторні діаграми.

**Завдання 5: Підготовка даних**

Попередньо обробіть набір даних за необхідністю для візуалізації. Це може включати виправлення даних, фільтрацію, агрегацію або трансформацію.

**Завдання 6: Базова візуалізація**

Створіть базову візуалізацію набору даних, щоб переконатися, що ви можете відображати дані правильно за допомогою Matplotlib. Розпочніть з простої діаграми для візуалізації однієї змінної.

**Завдання 7: Розширені візуалізації**

Реалізуйте більш складні візуалізації, виходячи з характеристик набору. Поекспериментуйте з різними функціями Matplotlib та налаштуваннями.

**Завдання 8: Декілька піддіаграм**

Навчіться створювати кілька піддіаграм в межах одного малюнка для відображення декількох візуалізацій поруч для кращого порівняння.

**Завдання 9: Експорт і обмін**

Реалізуйте функціональність для експорту візуалізацій як зображень (наприклад, PNG, SVG) або інтерактивних веб-додатків (наприклад, HTML)

**Виконання роботи**

**Папка Classes:**

**Вміст файлу bar\_chart\_visualizer.py:**

from matplotlib import pyplot as plt

from Classes.data\_visualizer import DataVisualizer

class BarChartVisualizer(DataVisualizer):

def process\_data(self):

self.data = self.data.groupby('Year')['Price'].sum()

print("Data processed for bar chart.") # стовпчикова діаграма

def plot(self):

self.data.plot(kind='bar', color='skyblue')

plt.title("Bar Chart")

plt.xlabel("Year")

plt.ylabel("Price Bitcoin")

print("Bar chart is build.")

**Вміст файлу combined\_chart\_visualizer.py:**

from Classes.data\_visualizer import DataVisualizer

import matplotlib.pyplot as plt

class CombinedChartVisualizer(DataVisualizer):

def process\_data(self):

self.bar\_data = self.data.groupby('Price')['Change'].sum()

self.line\_data = self.data.sort\_values('Year')

self.scatter\_data = self.data[['X', 'Y']]

def plot(self):

fig, axes = plt.subplots(1, 3, figsize=(15, 5))

# Стовпчикова діаграма

self.bar\_data.plot(kind='bar', color='skyblue', ax=axes[0])

axes[0].set\_title("Bar Chart")

axes[0].set\_xlabel("Year")

axes[0].set\_ylabel("Price Bitcoin")

# Лінійна діаграма

axes[1].plot(self.line\_data['Year'], self.line\_data['Change'], marker='o', linestyle='-', color='green')

axes[1].set\_title("Line Chart")

axes[1].set\_xlabel("Year")

axes[1].set\_ylabel("Change")

axes[1].tick\_params(axis='x', rotation=45)

# Діаграма розсіювання

axes[2].scatter(self.scatter\_data['X'], self.scatter\_data['Y'], color='red', alpha=0.5)

axes[2].set\_title("Scatter Chart")

axes[2].set\_xlabel("X")

axes[2].set\_ylabel("Y")

plt.tight\_layout()

**Вміст файлу data\_visualizer.py:**

from abc import ABC, abstractmethod

import pandas as pd

import matplotlib.pyplot as plt

import os

class DataVisualizer(ABC):

def \_\_init\_\_(self, file\_path):

self.file\_path = file\_path

self.data = None

def visualize(self):

try:

self.load\_data()

self.process\_data()

self.plot()

self.export()

except KeyError as e:

print(f"Error: Required column missing from file - {e}!")

except FileNotFoundError:

print("Error: File not found!")

except pd.errors.EmptyDataError:

print("Error: File is empty or wrong format!")

def load\_data(self):

self.data = pd.read\_csv(self.file\_path)

print("Data loaded successfully!")

@abstractmethod

def process\_data(self):

pass

@abstractmethod

def plot(self):

pass

def export(self):

save\_option = input("Do you want to save chart? (y/n): ").strip().lower()

if save\_option == 'y':

file\_format = input("Select file format to save, png or svg: ").strip().lower()

if file\_format not in ['png', 'svg']:

print("Invalid format. The chart will be saved in PNG format by default.")

file\_format = 'png'

file\_name = input("Enter file name without extension: ").strip()

if not file\_name:

print("Name is empty, so file was saved as 'chart'")

file\_name = "chart"

folder\_path = 'Lab8/data'

if not os.path.exists(folder\_path):

os.makedirs(folder\_path)

file\_name\_with\_extension = os.path.join(folder\_path, f"{file\_name}.{file\_format}")

try:

plt.tight\_layout()

plt.savefig(file\_name\_with\_extension)

print(f"Chart was saved in '{file\_name\_with\_extension}'.")

except Exception as e:

print(f"Error saving file {e}")

else:

print("Error: Chart was not saved!")

plt.show()

**Вміст файлу line\_chart\_visualizer.py:**

from Classes.data\_visualizer import DataVisualizer

import matplotlib.pyplot as plt

class LineChartVisualizer(DataVisualizer):

def process\_data(self):

self.data.sort\_values('Year', inplace=True)

print("Data processed for line chart.")

def plot(self):

plt.plot(self.data['Year'], self.data['Change'], marker='o', linestyle='-', color='green')

plt.title("Line Chart") # лінійна діаграма

plt.xlabel("Year")

plt.ylabel("Change")

plt.xticks(rotation=45)

print("Line chart is build.")

**Вміст файлу min\_max\_visualizer.py:**

import pandas as pd

from Classes.data\_visualizer import DataVisualizer

class MinMaxVisualizer(DataVisualizer):

def visualize(self):

try:

self.load\_data()

self.process\_data()

except FileNotFoundError:

print("Error: File not found!")

except pd.errors.EmptyDataError:

print("Error: File is empty or wrong format!")

def process\_data(self):

print("\nMin and max values for each column:")

for column in self.data.columns:

column\_min = self.data[column].min()

column\_max = self.data[column].max()

print(f"{column}: Min = {column\_min}, Max = {column\_max}")

def plot(self):

pass

**Вміст файлу scatter\_plot\_visualizer.py:**

from Classes.data\_visualizer import DataVisualizer

import matplotlib.pyplot as plt

class ScatterPlotVisualizer(DataVisualizer):

def process\_data(self):

print("Data processed for scatter chart.")

def plot(self):

plt.scatter(self.data['X'], self.data['Y'], color='red', alpha=0.5)

plt.title("Scatter Chart") # Діаграма розсіювання

plt.xlabel("X")

plt.ylabel("Y")

print("Scatter chart is build.")

**Папка Config:**

**Вміст файлу data.csv:**

Price,Change,Year,X,Y

0,+0,2009,1,1

0.06,+0.06,2010,2,3

6.05,+5.99,2011,4,5

8.46,+2.41,2012,5,6

189.00,+180.54,2013,8,9

526.02,+337.02,2014,9,11

272.60,-253.42,2015,11,14

567.23,+294.63,2016,12,17

4064.00,+3496.77,2017,13,19

7378.00,+3314.00,2018,14,20

7167.00,-211.00,2019,17,21

11335.00,+4168.00,2020,18,23

47683.00,+36348.00,2021,20,24

19475.00,-28208.00,2022,21,26

26874.00,+7399.00,2023,23,28

46627.78,+19753.78,2024,25,29

**Папка UI:**

**Вміст файлу user\_interface.py:**

import os

import sys

lab8\_root = os.path.abspath(os.path.join(os.path.dirname(\_\_file\_\_), ".."))

sys.path.append(lab8\_root)

from Classes.bar\_chart\_visualizer import BarChartVisualizer

from Classes.line\_chart\_visualizer import LineChartVisualizer

from Classes.scatter\_plot\_visualizer import ScatterPlotVisualizer

from Classes.min\_max\_visualizer import MinMaxVisualizer

from Classes.combined\_chart\_visualizer import CombinedChartVisualizer

def main():

while True:

print("\nMain menu:")

print("1. Bar Chart.")

print("2. Line Chart.")

print("3. Scatter Chart.")

print("4. Show max and min values.")

print("5. Show all charts.")

print("6. Exit.")

choice = input("Select option: ")

if choice == '6':

print("Exit from programm!")

break

file\_path = input("Enter path to CSV-file: ")

if choice == '1':

visualizer = BarChartVisualizer(file\_path)

elif choice == '2':

visualizer = LineChartVisualizer(file\_path)

elif choice == '3':

visualizer = ScatterPlotVisualizer(file\_path)

elif choice == '4':

visualizer = MinMaxVisualizer(file\_path)

elif choice == '5':

visualizer = CombinedChartVisualizer(file\_path)

else:

print("Error: Incorrect option. Please, try again!")

continue

visualizer.visualize()

**В кореневій папці лабораторної файл runner.py:**

from UI.user\_interface import main

if \_\_name\_\_ == "\_\_main\_\_":

main()

Результат виконання: а саме виведення усіх діаграм зображено на рис. 1 та виведення мінімальних та максимальних значень на рис. 2.

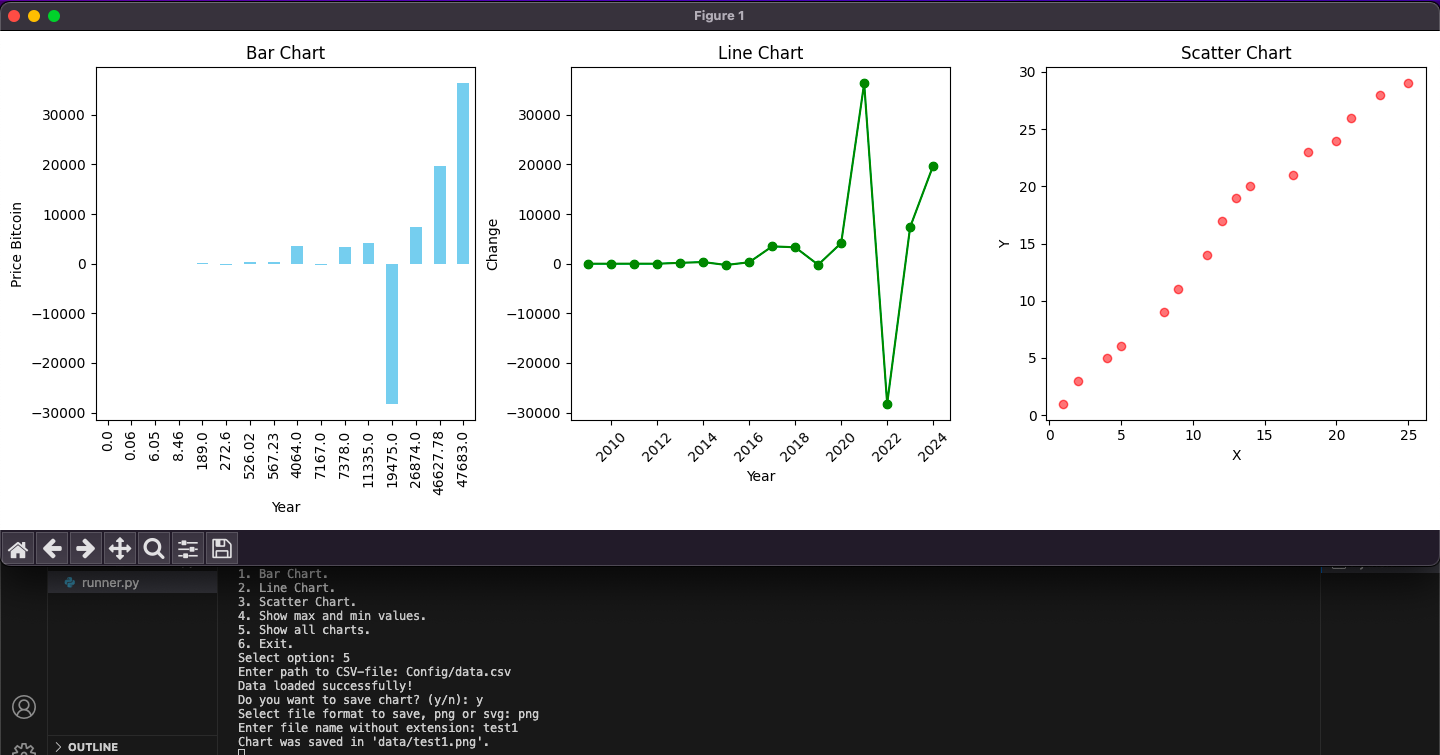


Рис. 1 – Вивід трьох діаграм

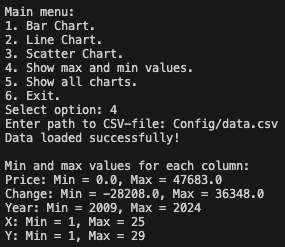


Рис. 2 – Вивід мінімальних та максимальних значень

**Висновок:** У ході виконання ЛР я створив багатофункціональний додаток для візуалізації CSV-наборів даних за допомогою Matplotlib. Дана програма генерує стовпчикову діаграму, лінійний графік та діаграму розсіювання, а також дає можливість відобразити усі діаграми разом та показати мінімальні та максимальні значення для даних.