Міністерство освіти і науки України
Національний університет «Львівська політехніка»

Звіт про виконання лабораторної роботи №8
з дисципліни «Спеціалізовані мови програмування»
на тему «Візуалізація та обробка даних
за допомогою спеціалізованих бібліотек Python»

Виконав:
студент групи РІ-32
Гусак Віктор
Прийняв:
Щербак С.С.

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

План 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)

Хід роботи

csv_data_analyzer.py:

```
class DataAnalyzer:
    @staticmethod
    def find_extremes(data):
        print("\nЕкстремальні значення по кожному стовпчику:")

        # Обробка числових стовпчиків
        for column in
data.select_dtypes(include='number').columns:
            print(f"{column}: Мінімум = {data[column].min()},
Максимум = {data[column].max()}")

        # Обробка текстових стовпчиків
        for column in
data.select_dtypes(include='object').columns:
            print(f"{column}: Початок = {data[column].min()},
Кінець = {data[column].max()}")
```

csv_data_loader.py:

```
import pandas as pd

class DataLoader:
    def __init__(self, file_path):
        self.file_path = file_path
        self.data = None

    def load_data(self):
        try:
            self.data = pd.read_csv(self.file_path)
            print("Дані успішно завантажено.")
        except FileNotFoundError:
            print(f"Файл {self.file_path} не знайдено.")
        except pd.errors.EmptyDataError:
            print("Файл порожній.")
        except Exception as e:
            print(f"Помилка при завантаженні даних: {e}")
```

```
    def get_data(self):
        return self.data
```

csv_data_visualizer.py:

```
import matplotlib.pyplot as plt

class DataVisualizer:
    def __init__(self, data):
        self.data = data

    def plot_line_chart(self, x_column, y_column):
        plt.figure(figsize=(8, 6))
        plt.plot(self.data[x_column], self.data[y_column],
marker='o', linestyle='-', color='blue')
        plt.title(f"Лінійний графік: {y_column} відносно
{x_column}")
        plt.xlabel(x_column)
        plt.ylabel(y_column)
        plt.grid()
        plt.show()

    def plot_bar_chart(self, x_column, y_column):
        plt.figure(figsize=(8, 6))
        plt.bar(self.data[x_column], self.data[y_column],
color='green')
        plt.title(f"Стовпчиковий графік: {y_column} відносно
{x_column}")
        plt.xlabel(x_column)
        plt.ylabel(y_column)
        plt.xticks(rotation=45)
        plt.show()

    def plot_scatter_chart(self, x_column, y_column):
        plt.figure(figsize=(8, 6))
        plt.scatter(self.data[x_column], self.data[y_column],
color='red')
        plt.title(f"Діаграма розсіювання: {y_column} відносно
{x_column}")
        plt.xlabel(x_column)
        plt.ylabel(y_column)
        plt.grid()
        plt.show()

    def create_subplots(self, plots):
        # Перевіряємо кількість піддіаграм
```

```
        fig, axes = plt.subplots(1, len(plots), figsize=(16, 6))

        # Якщо одна піддіаграма, обертаємо її у список
        if len(plots) == 1:
            axes = [axes]

        for ax, (x_column, y_column) in zip(axes, plots):
            ax.plot(self.data[x_column], self.data[y_column],
marker='o', linestyle='-', color='blue')
            ax.set_title(f"{y_column} відносно {x_column}")
            ax.set_xlabel(x_column)
            ax.set_ylabel(y_column)
        plt.tight_layout()
        plt.show()

    def save_visualization(self, file_name):
        plt.savefig(file_name, format='png')
        print(f"Візуалізацію збережено у файл {file_name}.")
        plt.close()
```

csv_visualizer_app.py:

```
import sys
import os

sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(
__file__))))

from classes.csv_data_loader import DataLoader
from classes.csv_data_analyzer import DataAnalyzer
from classes.csv_data_visualizer import DataVisualizer

class App:
    def __init__(self, file_path):
        self.data_loader = DataLoader(file_path)

    def run(self):
        self.data_loader.load_data()
        data = self.data_loader.get_data()

        if data is not None:
            DataAnalyzer.find_extremes(data)
            visualizer = DataVisualizer(data)

            # Лінійний графік
            x_column = input("Введіть стовпчик для осі X
(наприклад, 'Date'): ")
```

```
            y_column = input("Введіть стовпчик для осі Y
(наприклад, 'Value'): ")
            visualizer.plot_line_chart(x_column, y_column)

            # Стовпчиковий графік
            visualizer.plot_bar_chart(x_column, y_column)

            # Діаграма розсіювання
            visualizer.plot_scatter_chart(x_column, y_column)

            # Кілька піддіаграм
            visualizer.create_subplots([(x_column, y_column)])

            # Збереження візуалізації
            file_name = input("Введіть ім'я файлу для збереження
(наприклад, 'visualization.png'): ")
            visualizer.save_visualization(file_name)
```

csv_visualizer_app.py:

```
import sys
import os

sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(
__file__))))

from classes.csv_visualizer_app import App

app = App('../assets/data.csv')
app.run()
```

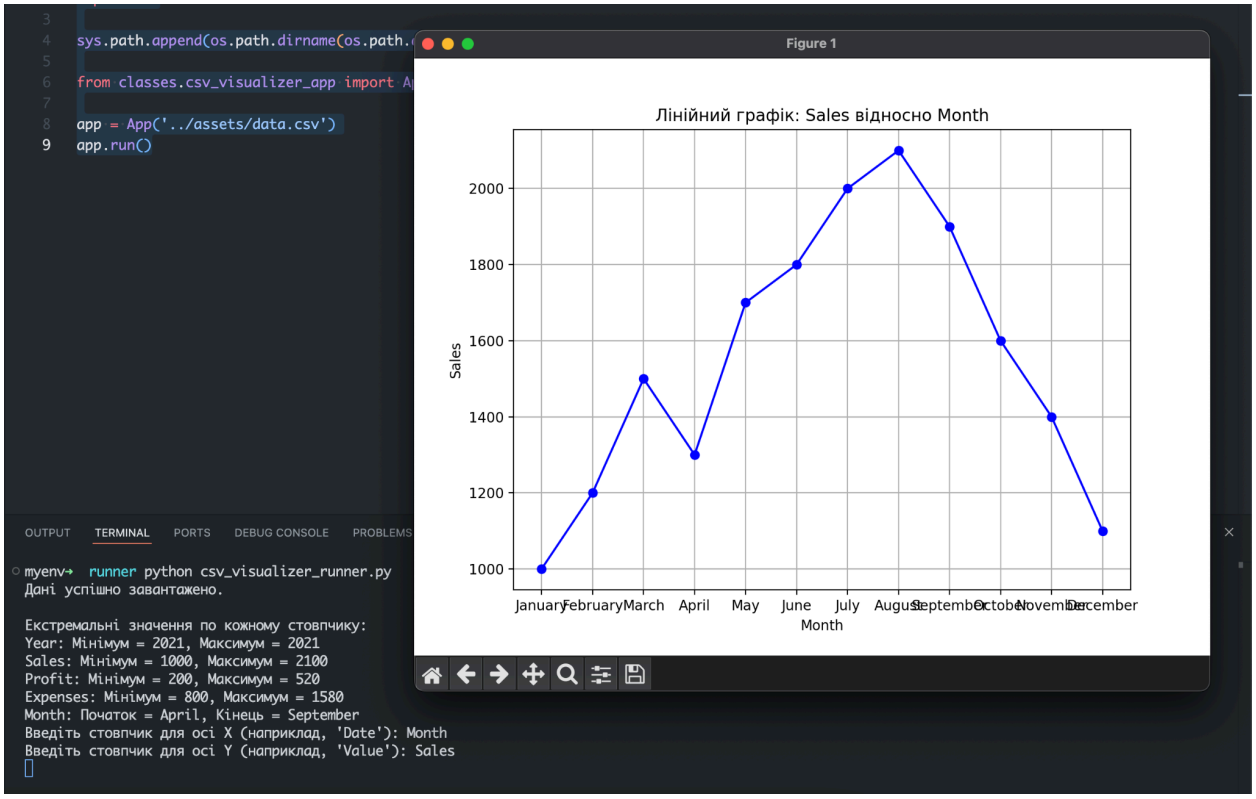

Рис. 1. Приклад роботи програми

Висновки: Виконавши ці завдання, було створено багатофункціональний додаток для візуалізації CSV-наборів даних за допомогою Matplotlib.