МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП'ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра ІСМ

******

Звіт

до лабораторної роботи №8

На тему “Візуалізація та обробка даних за допомогою спеціалізованих бібліотек Python”

З дисципліни “Спеціалізовані мови програмування”

*Виконав:*

*ст. гр. ІТ-31*

*Іштван Комоній*

*Прийняв:*

*Щербак С. С.*

*Львів - 2023*

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

**Хід роботи**

**Програмний код:**

/data.csv

date,income,products\_sold,ads\_company

2023-12-01, 100, 5, push ads

2023-12-02, 120, 6, push ads

2023-12-03, 105, 4, push ads

2023-12-04, 98, 3, push ads

2023-12-05, 50, 1, push ads

2023-12-06, 324, 20, seo traffic

2023-12-07, 1050, 50, seo traffic

2023-12-08, 980, 43, seo traffic

2023-12-09, 2040, 80, seo traffic

2023-12-10, 673, 20, push ads

2023-12-11, 200, 7, push ads

2023-12-12, 500, 18, seo traffic

/client\_application/client\_application

from command.invoker import Invoker

from client\_application.client\_application\_receiver import ClientApplicationReceiver

from client\_application.commands import DisplayAllcharts, DisplayBarchart, DisplayLinechart, DisplayPiechart, SaveChartCommand

import pandas as pd

class ClientApplication():

def \_\_init\_\_(self):

self.data\_source = 'lab8/data.csv'

self.data\_frame = None

self.app\_receiver = ClientApplicationReceiver()

self.app\_invoker = Invoker()

def set\_data\_source(self, val):

self.data\_source = val

def set\_data\_frame(self, val):

self.data\_frame = val

def set\_primary\_data(self):

data\_source = input(

"enter filename of datasource(skip for default): ")

if len(data\_source):

self.set\_data\_source(data\_source)

self.set\_data\_frame(pd.read\_csv(self.data\_source))

def display\_barchart(self):

display\_barchart\_command = DisplayBarchart(

self.app\_receiver, self.data\_frame)

self.app\_invoker.add\_command(display\_barchart\_command)

self.app\_invoker.execute\_current\_command()

def display\_piechart(self):

display\_piechart\_command = DisplayPiechart(

self.app\_receiver, self.data\_frame)

self.app\_invoker.add\_command(display\_piechart\_command)

self.app\_invoker.execute\_current\_command()

def display\_linechart(self):

display\_linechart\_command = DisplayLinechart(

self.app\_receiver, self.data\_frame)

self.app\_invoker.add\_command(display\_linechart\_command)

self.app\_invoker.execute\_current\_command()

def display\_allcharts(self):

display\_allcharts\_command = DisplayAllcharts(

self.app\_receiver, self.data\_frame)

self.app\_invoker.add\_command(display\_allcharts\_command)

self.app\_invoker.execute\_current\_command()

def save\_chart(self, filename):

save\_chart\_command = SaveChartCommand(

self.app\_receiver, self.data\_frame, filename)

self.app\_invoker.add\_command(save\_chart\_command)

self.app\_invoker.execute\_current\_command()

@staticmethod

def show\_menu():

print("choose menu option")

print("[ 1 ] - display bar chart")

print("[ 2 ] - display pie chart")

print("[ 3 ] - display line chart")

print("[ 4 ] - display all charts")

print("[ 5 ] - change sourcefile")

print("[ 6 ] - save chart")

print("[ 0 ] - exit")

def loop\_menu(self):

while True:

self.show\_menu()

menu\_choice = int(input("menu key: "))

if (menu\_choice == 1):

self.display\_barchart()

elif (menu\_choice == 2):

self.display\_piechart()

elif (menu\_choice == 3):

self.display\_linechart()

elif (menu\_choice == 4):

self.display\_allcharts()

elif (menu\_choice == 5):

new\_sourcefile = input("enter filename of new datasource: ")

if len(new\_sourcefile):

self.set\_data\_source(new\_sourcefile)

self.set\_data\_frame(pd.read\_csv(new\_sourcefile))

elif (menu\_choice == 6):

filename = input("enter filename: ")

if len(filename):

self.save\_chart(filename)

else:

self.save\_chart('chart.png')

else:

break

def launch(self):

self.set\_primary\_data()

self.display\_barchart()

self.loop\_menu()

/client\_application/client\_application\_receiver

from command.receiver import Receiver

import matplotlib.pyplot as plt

class ClientApplicationReceiver(Receiver):

@staticmethod

def run\_display\_barchart(data\_frame):

data\_frame.plot(kind='bar', x='date', y='income', label='Income')

plt.title('Income by date')

plt.xticks(rotation=90)

plt.tight\_layout()

plt.show()

@staticmethod

def run\_display\_piechart(data\_frame):

grouped\_data = data\_frame.groupby('ads\_company')['products\_sold'].sum()

grouped\_data.plot(kind='pie', autopct='%1.1f%%', startangle=90)

plt.title('Products Sold by Ads Company')

plt.tight\_layout()

plt.show()

@staticmethod

def run\_display\_linechart(data\_frame):

data\_frame.plot(kind='line', x='date', y=['income', 'products\_sold'])

plt.title('Income and units sold')

plt.legend(['Income', 'Products Sold'])

plt.xticks(rotation=90)

plt.tight\_layout()

plt.show()

@staticmethod

def run\_display\_allcharts(data\_frame):

fig, axs = plt.subplots(2, 2, figsize=(12, 8))

axs[0, 0].bar(data\_frame['date'], data\_frame['income'], label='Income')

axs[0, 0].set\_title('Income by date')

axs[0, 0].legend()

grouped\_data = data\_frame.groupby('ads\_company')['products\_sold'].sum()

axs[0, 1].pie(grouped\_data, autopct='%1.1f%%', startangle=90)

axs[0, 1].set\_title('Products Sold by Ads Company')

axs[1, 0].plot(data\_frame['date'],

data\_frame[['income', 'products\_sold']])

axs[1, 0].set\_title('Income and units sold')

axs[1, 0].legend(['Income', 'Products Sold'])

axs[1, 1].axis('off')

plt.setp(axs[1, 0].xaxis.get\_majorticklabels(), rotation=90)

plt.setp(axs[0, 0].xaxis.get\_majorticklabels(), rotation=90)

plt.tight\_layout()

plt.show()

@staticmethod

def run\_save\_chart(data\_frame, filename):

fig, axs = plt.subplots(2, 2, figsize=(12, 8))

axs[0, 0].bar(data\_frame['date'], data\_frame['income'], label='Income')

axs[0, 0].set\_title('Income by date')

axs[0, 0].legend()

grouped\_data = data\_frame.groupby('ads\_company')['products\_sold'].sum()

axs[0, 1].pie(grouped\_data, autopct='%1.1f%%', startangle=90)

axs[0, 1].set\_title('Products Sold by Ads Company')

axs[1, 0].plot(data\_frame['date'],

data\_frame[['income', 'products\_sold']])

axs[1, 0].set\_title('Income and units sold')

axs[1, 0].legend(['Income', 'Products Sold'])

axs[1, 1].axis('off')

plt.setp(axs[1, 0].xaxis.get\_majorticklabels(), rotation=90)

plt.setp(axs[0, 0].xaxis.get\_majorticklabels(), rotation=90)

plt.tight\_layout()

plt.savefig(filename)

/client\_application/commands

from command.i\_command import ICommand

class DisplayBarchart(ICommand):

def \_\_init\_\_(self, \_app\_receiver, data\_frame):

self.\_app\_receiver = \_app\_receiver

self.data\_frame = data\_frame

def execute(self):

self.\_app\_receiver.run\_display\_barchart(self.data\_frame)

class DisplayPiechart(ICommand):

def \_\_init\_\_(self, \_app\_receiver, data\_frame):

self.\_app\_receiver = \_app\_receiver

self.data\_frame = data\_frame

def execute(self):

self.\_app\_receiver.run\_display\_piechart(self.data\_frame)

class DisplayLinechart(ICommand):

def \_\_init\_\_(self, \_app\_receiver, data\_frame):

self.\_app\_receiver = \_app\_receiver

self.data\_frame = data\_frame

def execute(self):

self.\_app\_receiver.run\_display\_linechart(self.data\_frame)

class DisplayAllcharts(ICommand):

def \_\_init\_\_(self, \_app\_receiver, data\_frame):

self.\_app\_receiver = \_app\_receiver

self.data\_frame = data\_frame

def execute(self):

self.\_app\_receiver.run\_display\_allcharts(self.data\_frame)

class SaveChartCommand(ICommand):

def \_\_init\_\_(self, \_app\_receiver, data\_frame, filename):

self.\_app\_receiver = \_app\_receiver

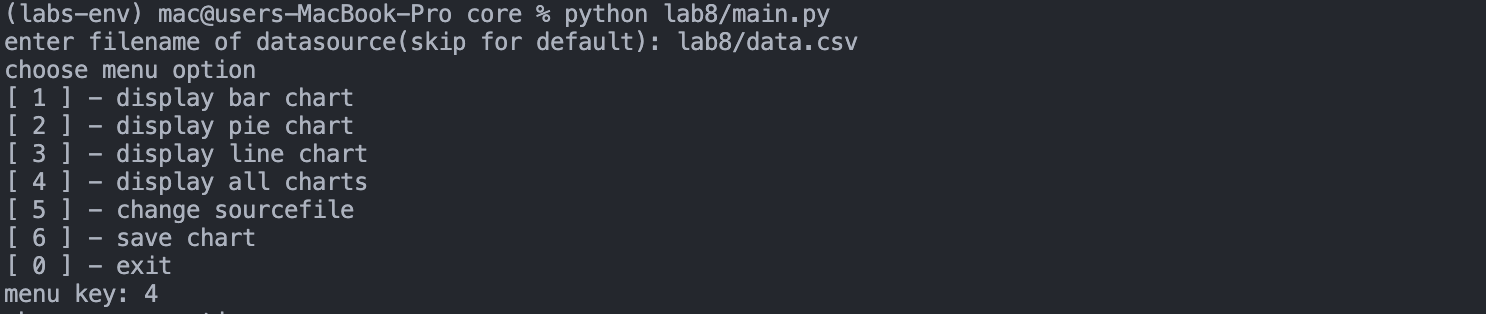
self.data\_frame = data\_frame

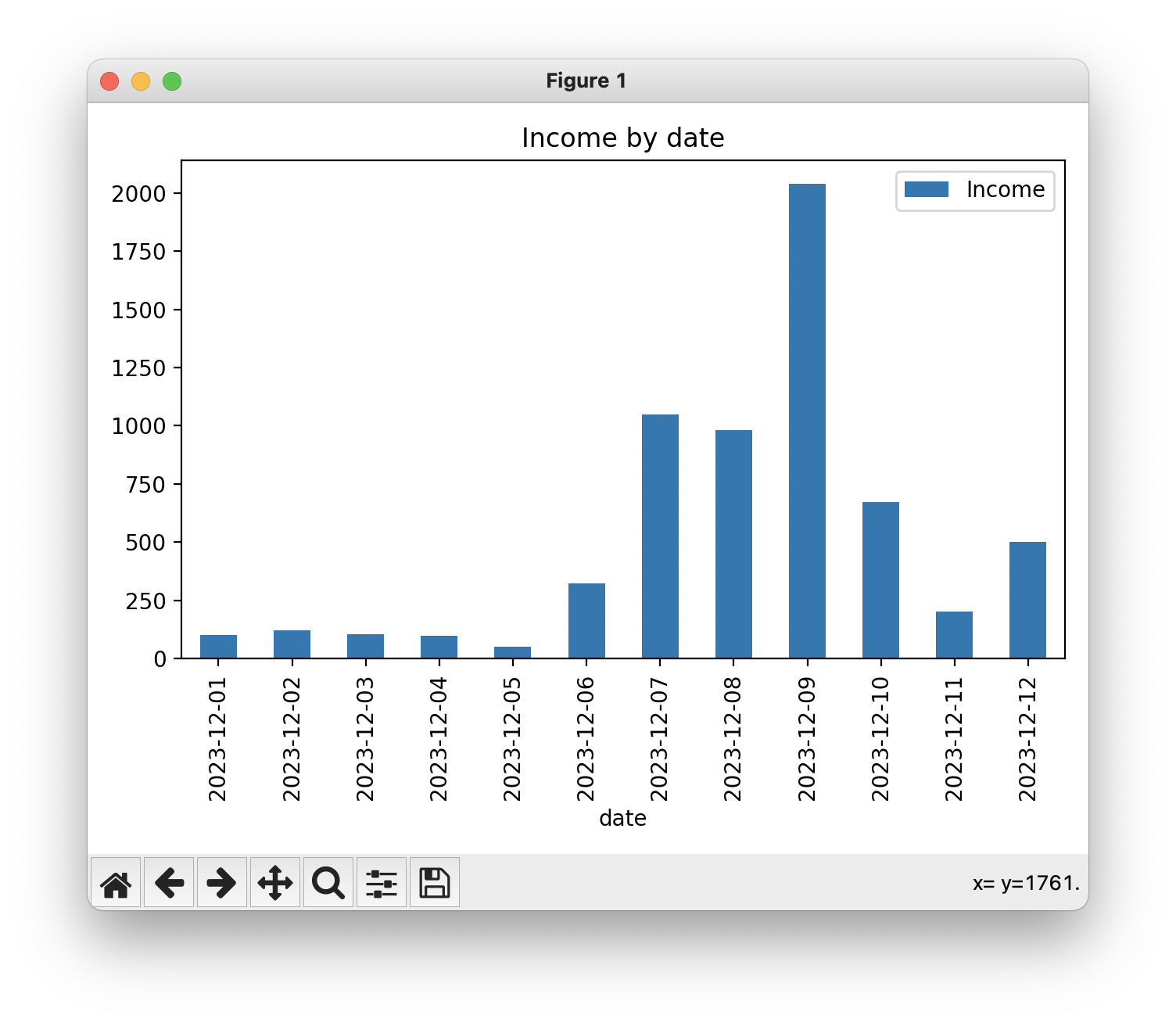
self.filename = filename

def execute(self):

self.\_app\_receiver.run\_save\_chart(self.data\_frame, self.filename)

Результат виконання програми:







**Висновок:** Під час виконання даної лабораторної роботи було створено додаток для візуалізації CSV-наборів даних за допомогою Matplotlib.