МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП'ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра ІСМ

******

Звіт

до лабораторної роботи №7

На тему “Робота з API та веб-сервісами”

З дисципліни “Спеціалізовані мови програмування”

*Виконав:*

*ст. гр. ІТ-31*

*Іштван Комоній*

*Прийняв:*

*Щербак С. С.*

*Львів - 2023*

**Мета роботи:** Створення консольного об’єктно - орієнтованого додатка з використанням API.

**Хід роботи**

**Програмний код:**

/web\_service/web\_service

import requests

from helpers.helpers import make\_get\_request

class WebService:

def \_\_init\_\_(self, route, size):

self.route = route

self.size = size

def set\_route(self, val):

self.route = val

def set\_size(self, val):

self.size = val

def get\_data(self):

try:

url = 'https://random-data-api.com/api/v2/' + self.route

if self.size > 1:

url += '?size=' + str(self.size)

response = make\_get\_request(url)

return response

except requests.exceptions.RequestException:

return {}

/client\_application/client\_application

from command.receiver import Receiver

from command.i\_command import ICommand

from command.invoker import Invoker

from client\_application.history.history import History

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

from client\_application.commands import GetApiDataCommand, GetHistoryDataCommand, SaveDataToJsonFileCommand, SaveDataToTxtFileCommand

class ClientApplication():

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

self.route = 'users'

self.size = 5

self.display\_type = 'original'

self.color = 'green'

self.app\_receiver = ClientApplicationReceiver()

self.app\_invoker = Invoker()

self.history = History()

def set\_route(self, val):

self.route = val

def set\_size(self, val):

self.size = val

def set\_display\_type(self, val):

self.display\_type = val

def set\_color(self, val):

self.color = val

def set\_primary\_data(self):

route = input(

"set route(users, addresses, banks, appliances, beers, blood\_types, credit\_cards): ")

if len(route):

self.set\_route(route)

size = int(input("set size: "))

self.set\_size(size)

display\_type = input(

"set display type(original, list, table): ")

if len(display\_type):

self.set\_display\_type(display\_type)

color = input(

"set color(blue, green, red, magenta, yellow, white, cyan): ")

if len(color):

self.set\_color(color)

def get\_api\_data\_for\_current\_config(self):

get\_api\_data\_command = GetApiDataCommand(

self.app\_receiver, self.route, self.size, self.history, self.display\_type, self.color)

self.app\_invoker.add\_command(get\_api\_data\_command)

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

def get\_history\_data\_for\_current\_config(self):

get\_history\_data\_command = GetHistoryDataCommand(

self.app\_receiver, self.history, self.display\_type, self.color)

self.app\_invoker.add\_command(get\_history\_data\_command)

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

def save\_to\_txt\_file(self):

filename = input("enter filename before saving: ")

save\_data\_to\_txt\_file\_command = SaveDataToTxtFileCommand(

self.app\_receiver, self.history, self.display\_type, filename)

self.app\_invoker.add\_command(save\_data\_to\_txt\_file\_command)

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

def save\_to\_json\_file(self):

filename = input("enter filename before saving: ")

save\_data\_to\_json\_file\_command = SaveDataToJsonFileCommand(

self.app\_receiver, self.history, self.display\_type, filename)

self.app\_invoker.add\_command(save\_data\_to\_json\_file\_command)

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

@staticmethod

def show\_menu():

print("choose menu option")

print("[ 1 ] - get data")

print("[ 2 ] - set route(users, addresses, banks, appliances, beers, blood\_types, credit\_cards)")

print("[ 3 ] - set size")

print("[ 4 ] - set display type(original, list, table)")

print("[ 5 ] - set color(blue, green, red, magenta, yellow, white, cyan)")

print("[ 6 ] - show history")

print("[ 7 ] - save to txt")

print("[ 8 ] - save to json")

print("[ 0 ] - exit")

def loop\_menu(self):

while True:

self.show\_menu()

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

if (menu\_choice == 1):

self.get\_api\_data\_for\_current\_config()

elif (menu\_choice == 2):

new\_route = input("enter new route: ")

self.set\_route(new\_route)

elif (menu\_choice == 3):

new\_size = int(input("enter new size: "))

self.set\_size(new\_size)

elif (menu\_choice == 4):

new\_display\_type = input("enter new display type: ")

self.set\_display\_type(new\_display\_type)

elif (menu\_choice == 5):

new\_color = input("enter new color: ")

self.set\_color(new\_color)

elif (menu\_choice == 6):

self.get\_history\_data\_for\_current\_config()

elif (menu\_choice == 7):

self.save\_to\_txt\_file()

elif (menu\_choice == 8):

self.save\_to\_json\_file()

else:

break

def launch(self):

self.set\_primary\_data()

self.get\_api\_data\_for\_current\_config()

self.loop\_menu()

/client\_application/client\_application\_receiver

from command.receiver import Receiver

from helpers.helpers import api\_data\_to\_type, api\_data\_to\_type\_painted, json\_file\_saver, text\_file\_saver

from web\_service.web\_service import WebService

class ClientApplicationReceiver(Receiver):

@staticmethod

def run\_get\_api\_data(route, size, history, display\_type, color):

ws = WebService(route, size)

data = ws.get\_data()

history.log(data)

print(api\_data\_to\_type\_painted(data, display\_type, color))

@staticmethod

def run\_get\_history\_data(history, display\_type, color):

history\_log = history.get\_history()

result\_log = ''

for log in history\_log:

result\_log += api\_data\_to\_type\_painted(log, display\_type, color)

print(result\_log)

@staticmethod

def run\_save\_data\_to\_txt\_file(history, display\_type, filename):

history\_log = history.get\_history()

result\_log = ''

for log in history\_log:

result\_log += api\_data\_to\_type(log, display\_type)

text\_file\_saver(filename + '.txt', result\_log)

@staticmethod

def run\_save\_data\_to\_json\_file(history, display\_type, filename):

history\_log = history.get\_history()

json\_file\_saver(filename + '.json', history\_log)

/client\_application/commands

from command.i\_command import ICommand

class GetApiDataCommand(ICommand):

def \_\_init\_\_(self, \_app\_receiver, route, size, history, display\_type, color):

self.\_app\_receiver = \_app\_receiver

self.route = route

self.size = size

self.history = history

self.display\_type = display\_type

self.color = color

def execute(self):

self.\_app\_receiver.run\_get\_api\_data(

self.route, self.size, self.history, self.display\_type, self.color)

class GetHistoryDataCommand(ICommand):

def \_\_init\_\_(self, \_app\_receiver, history, display\_type, color):

self.\_app\_receiver = \_app\_receiver

self.history = history

self.display\_type = display\_type

self.color = color

def execute(self):

self.\_app\_receiver.run\_get\_history\_data(

self.history, self.display\_type, self.color)

class SaveDataToTxtFileCommand(ICommand):

def \_\_init\_\_(self, \_app\_receiver, history, display\_type, filename):

self.\_app\_receiver = \_app\_receiver

self.history = history

self.display\_type = display\_type

self.filename = filename

def execute(self):

self.\_app\_receiver.run\_save\_data\_to\_txt\_file(

self.history, self.display\_type, self.filename)

class SaveDataToJsonFileCommand(ICommand):

def \_\_init\_\_(self, \_app\_receiver, history, display\_type, filename):

self.\_app\_receiver = \_app\_receiver

self.history = history

self.display\_type = display\_type

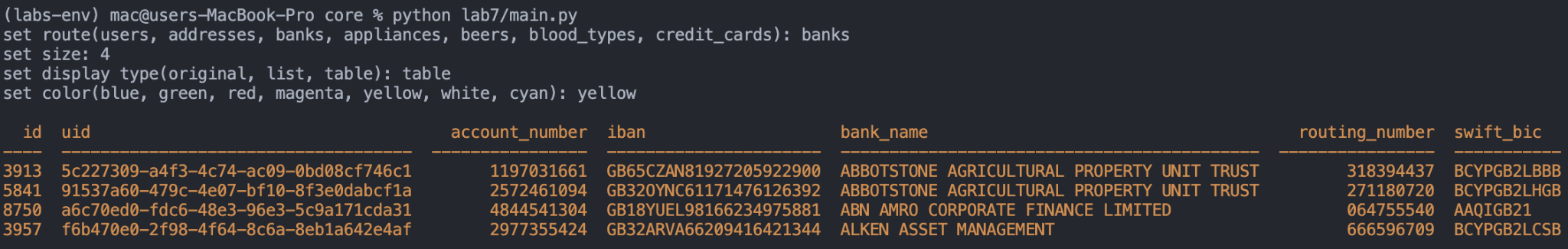
self.filename = filename

def execute(self):

self.\_app\_receiver.run\_save\_data\_to\_json\_file(

self.history, self.display\_type, self.filename)

Результат виконання програми:



**Висновок:** Під час виконання даної лабораторної роботи було створено консольний об’єктно - орієнтований додаток÷ з використанням API.