МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

**ЗВІТ**

**ПРО ВИКОНАННЯ ПРАКТИЧНОЇ РОБОТИ №9**

з дисципліни «Програмування мовою Python»

на тему «Робота з даними формату CSV та JSON»

Виконав студент групи КН-45-5/1

Чесной В.С.

Перевірила Парфененко Ю.В.

Варіант 19

Суми 2025

**Завдання 1**

1. **Постановка задачі**

Напишіть програму для обробки .csv файла та збереження результатів у інший .csv файл. У програмі реалізувати обробку помилок відкриття файлу за допомогою конструкції try-except.

Знайти дані Exports of goods and services (% of GDP) для уcіх країн світу за 2015 та 2019 рік . Вивести вміст .csv файлу на екран. Організувати пошук даних у діапазоні, який ввів користувач з клавіатури та записати результат пошуку у новий .csv файл.

1. **Текст програми**

import csv

import os

# Поточна директорія

current\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

input\_file = os.path.join(current\_dir, "Exports of goods and services (% of GDP)15and19.csv")

output\_file = os.path.join(current\_dir, "filtered\_exports.csv")

try:

# Визначаємо роздільник (',' або ';')

with open(input\_file, 'r', encoding='utf-8') as f:

sample = f.read(1024)

delimiter = ';' if sample.count(';') > sample.count(',') else ','

# Зчитуємо файл

with open(input\_file, newline='', encoding='utf-8') as csvfile:

reader = csv.DictReader(csvfile, delimiter=delimiter)

data = list(reader)

# Перевіримо, які саме колонки є

print("Колонки у файлі:", reader.fieldnames)

# Знайдемо колонки, які містять 2015 і 2019

col\_2015 = next((c for c in reader.fieldnames if "2015" in c), None)

col\_2019 = next((c for c in reader.fieldnames if "2019" in c), None)

if not col\_2015 or not col\_2019:

raise KeyError("Не знайдено колонки для 2015 або 2019 року")

print("\nВміст файлу:\n")

for row in data:

print(row)

# Введення діапазону

print("\nВведіть діапазон для пошуку даних (у % ВВП):")

lower = float(input("Мінімальне значення: "))

upper = float(input("Максимальне значення: "))

filtered = []

for row in data:

try:

val\_2015 = float(row[col\_2015]) if row[col\_2015] else None

val\_2019 = float(row[col\_2019]) if row[col\_2019] else None

except ValueError:

continue

if (val\_2015 and lower <= val\_2015 <= upper) or (val\_2019 and lower <= val\_2019 <= upper):

filtered.append(row)

# Запис у новий файл

if filtered:

with open(output\_file, 'w', newline='', encoding='utf-8') as f\_out:

writer = csv.DictWriter(f\_out, fieldnames=reader.fieldnames, delimiter=delimiter)

writer.writeheader()

writer.writerows(filtered)

print(f"\n Результати збережено у файл: {output\_file}")

else:

print("\n Дані у вказаному діапазоні не знайдені.")

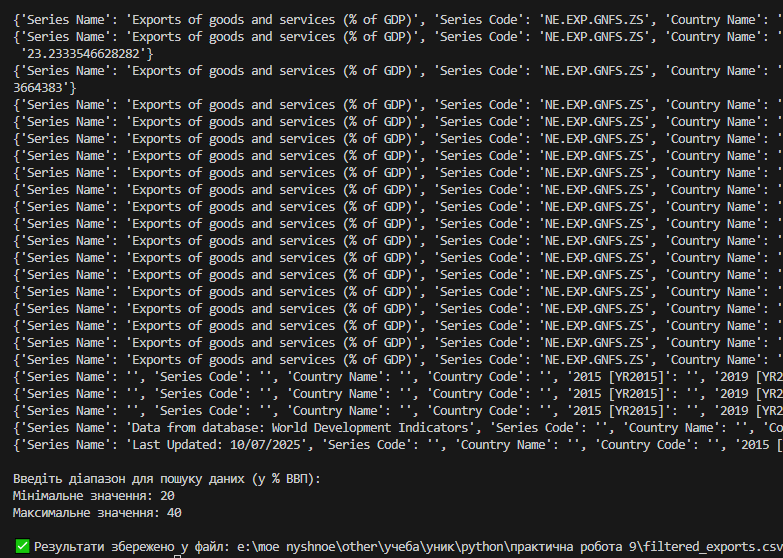
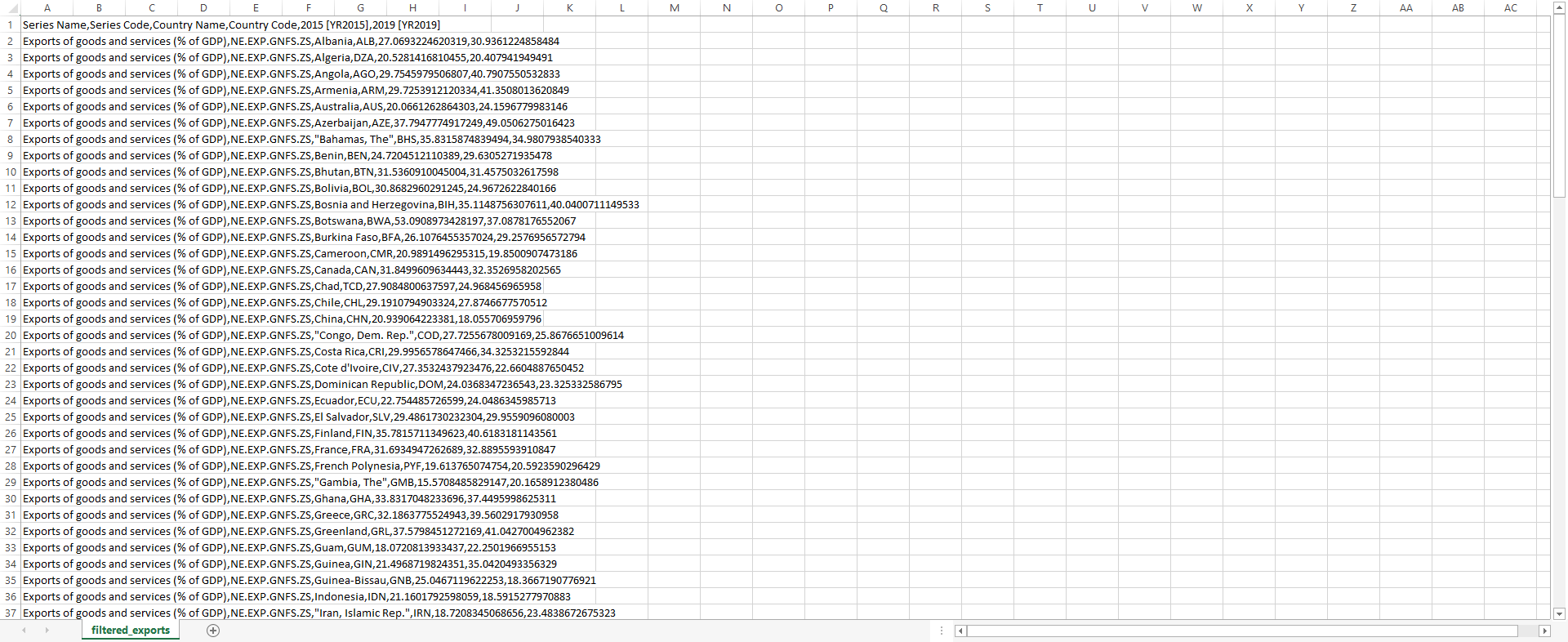
except FileNotFoundError:

print(f" Помилка: файл '{os.path.basename(input\_file)}' не знайдено!")

except Exception as e:

print(f" Сталася помилка при обробці файлу: {e}")

1. **Приклад роботи програми**

****

**Завдання 2**

1. **Постановка задачі**

Відповідно до свого варіанту написати програму, яка створює об’єкт JSON для збереження даних із заданої предметної області та виконує обробку даних. Реалізувати у вигляді окремих функцій та організувати їх виклик користувачем у діалоговому режимі:

1. Виведення на екран вмісту JSON файлу;

2. Додавання (видалення) нового запису у JSON файл;

3. Пошук даних у JSON файлі за одним із полів на вибір;

4. Розв’язання завдань відповідно до варіанту, результат виконання завдання записати в інший JSON файл.

Задано дані про розклад поїздів, які проходять через певну станцію: номер поїзда, призначення (звідки куди, наприклад, Київ – Харків), час (години та хвилини) прибуття, час (години та хвилини) відправлення. Година та хвилина – цілі, додатні числа; значення годин не перевищує «23», значення хвилин – «59». Загальна кількість поїздів n=10. Поїзди приходять кожен день. Скласти програму, яка визначає, які поїзди (номер і призначення) стоять у визначений момент часу на станції.

1. **Текст програми**

import json

import os

from datetime import time

# ====== ФАЙЛИ ======

BASE\_DIR = os.path.dirname(os.path.abspath(\_\_file\_\_))

SOURCE\_FILE = os.path.join(BASE\_DIR, "trains.json") # вихідний (не змінюємо)

RESULT\_FILE = os.path.join(BASE\_DIR, "trains\_at\_station.json") # робочий (усі зміни сюди)

# ====== ДОПОМІЖНІ ФУНКЦІЇ ======

def load\_data():

"""Завантажує дані з основного або результатного файлу, якщо вони не порожні"""

# спроба відкрити результатний файл

if os.path.exists(RESULT\_FILE) and os.path.getsize(RESULT\_FILE) > 0:

try:

with open(RESULT\_FILE, "r", encoding="utf-8") as f:

return json.load(f)

except json.JSONDecodeError:

print("⚠️ Файл trains\_at\_station.json пошкоджений або порожній. Використовую вихідні дані.")

# якщо результатного нема або він пустий

with open(SOURCE\_FILE, "r", encoding="utf-8") as f:

return json.load(f)

def save\_data(data):

"""Записує дані у результатний файл (trains\_at\_station.json)"""

with open(RESULT\_FILE, "w", encoding="utf-8") as f:

json.dump(data, f, ensure\_ascii=False, indent=4)

def show\_data(data):

"""Виводить вміст даних"""

if not data:

print("Дані відсутні.")

else:

print("\n--- ПОТОЧНІ ДАНІ ---")

for t in data:

print(f"Поїзд №{t['number']}: {t['route']}, "

f"прибуття {t['arrival']['hour']:02d}:{t['arrival']['minute']:02d}, "

f"відправлення {t['departure']['hour']:02d}:{t['departure']['minute']:02d}")

# ====== ОСНОВНІ ОПЕРАЦІЇ ======

def add\_train(data):

"""Додає новий запис"""

number = input("Номер поїзда: ")

route = input("Маршрут (звідки – куди): ")

arrival\_hour = int(input("Година прибуття (0–23): "))

arrival\_min = int(input("Хвилини прибуття (0–59): "))

departure\_hour = int(input("Година відправлення (0–23): "))

departure\_min = int(input("Хвилини відправлення (0–59): "))

train = {

"number": number,

"route": route,

"arrival": {"hour": arrival\_hour, "minute": arrival\_min},

"departure": {"hour": departure\_hour, "minute": departure\_min}

}

data.append(train)

save\_data(data)

print("✅ Запис додано у trains\_at\_station.json!")

def delete\_train(data):

"""Видаляє запис за номером"""

num = input("Введіть номер поїзда для видалення: ")

new\_data = [t for t in data if t["number"] != num]

if len(new\_data) < len(data):

save\_data(new\_data)

print("🚆 Запис видалено з trains\_at\_station.json.")

else:

print("❌ Поїзд не знайдено.")

def search\_train(data):

"""Пошук по маршруту"""

route = input("Введіть напрям (або частину назви): ").lower()

found = [t for t in data if route in t["route"].lower()]

if found:

print("\n🔎 Знайдені поїзди:")

show\_data(found)

save\_data(found)

print("✅ Результати пошуку записано у trains\_at\_station.json.")

else:

print("Нічого не знайдено.")

def trains\_at\_time(data):

"""Знаходить поїзди, що стоять на станції у певний момент часу"""

hour = int(input("Година (0–23): "))

minute = int(input("Хвилина (0–59): "))

current = time(hour, minute)

standing = []

for t in data:

arrival = time(t["arrival"]["hour"], t["arrival"]["minute"])

departure = time(t["departure"]["hour"], t["departure"]["minute"])

if arrival <= current <= departure:

standing.append(t)

if standing:

print("\n🚉 На станції зараз стоять:")

show\_data(standing)

else:

print("На станції зараз немає поїздів.")

save\_data(standing)

print("✅ Результати записано у trains\_at\_station.json.")

# ====== МЕНЮ ======

def main():

data = load\_data()

while True:

print("\n--- МЕНЮ ---")

print("1. Показати всі записи")

print("2. Додати запис")

print("3. Видалити запис")

print("4. Пошук за маршрутом")

print("5. Визначити поїзди, що стоять на станції")

print("0. Вихід")

choice = input("Ваш вибір: ")

if choice == "1":

show\_data(data)

elif choice == "2":

add\_train(data)

data = load\_data()

elif choice == "3":

delete\_train(data)

data = load\_data()

elif choice == "4":

search\_train(data)

elif choice == "5":

trains\_at\_time(data)

elif choice == "0":

print("Вихід із програми.")

break

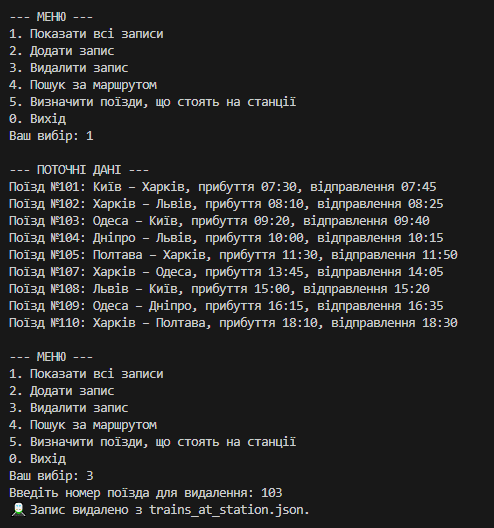
else:

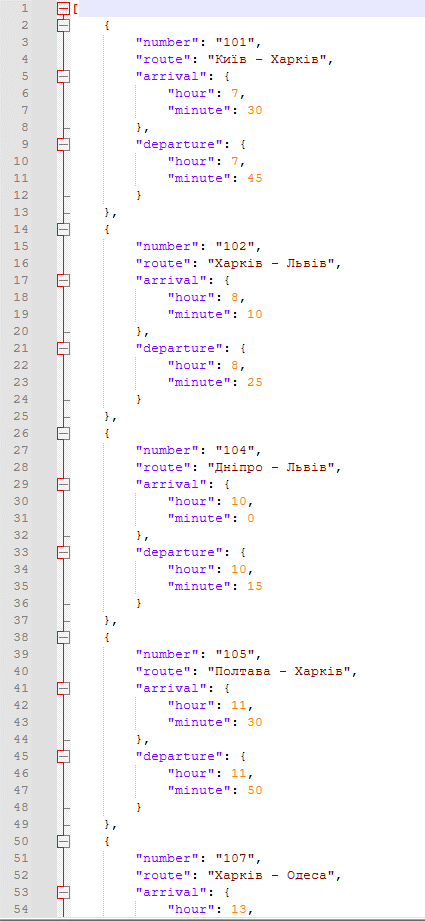
print("Невірний вибір!")

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

main()

1. **Приклад виконання програми**





Посилання на GitHub з програмами [**тут**](https://github.com/TenderBly/Python-Laboratory)