Звіт з практичної роботи No14

Тема роботи: Робота з серіалізацією даних у Python (JSON, pickle).

Виконав: Кравченко Сергій, ТЦР-12

Дата: 02.06.2025

Мета роботи:

Опанувати навички серіалізації та десеріалізації даних, навчитися

зберігати та відновлювати складні структури даних.

Хід роботи:

Постановка завдання:

Завдання 1. "Збереження результатів гри"

● Створіть програму, яка зберігає статистику гравця у JSON-файлі.

● Реалізуйте можливість оновлення статистики після кожної гри.

Код програми:

import json

import os

def save\_game\_stats(player\_name, score, stats\_file='game\_stats.json'):

stats = {}

if os.path.exists(stats\_file):

with open(stats\_file, 'r') as f:

stats = json.load(f)

if player\_name in stats:

stats[player\_name]['games\_played'] += 1

stats[player\_name]['total\_score'] += score

if score > stats[player\_name]['high\_score']:

stats[player\_name]['high\_score'] = score

else:

stats[player\_name] = {

'games\_played': 1,

'total\_score': score,

'high\_score': score

}

with open(stats\_file, 'w') as f:

json.dump(stats, f, indent=4)

def task1\_example():

save\_game\_stats('Player1', 100)

save\_game\_stats('Player2', 150)

save\_game\_stats('Player1', 200)

Завдання 2. "Контакти в JSON"

● Реалізуйте програму, яка створює словник контактів і зберігає його у

JSON-файлі.

● Додайте можливість завантаження контактів із файлу для

подальшого використання.

Код програми:

import json

def manage\_contacts(contacts\_file='contacts.json'):

try:

with open(contacts\_file, 'r') as f:

contacts = json.load(f)

except FileNotFoundError:

contacts = {}

while True:

print("\n1. Додати контакт")

print("2. Показати контакти")

print("3. Вийти")

choice = input("Виберіть дію: ")

if choice == '1':

name = input("Ім'я: ")

phone = input("Телефон: ")

contacts[name] = phone

with open(contacts\_file, 'w') as f:

json.dump(contacts, f, indent=4)

elif choice == '2':

print("\nКонтакти:")

for name, phone in contacts.items():

print(f"{name}: {phone}")

elif choice == '3':

break

Приклад роботи програми:

1. Додати контакт

2. Показати контакти

3. Вийти

Виберіть дію: 1

Ім'я: Олексій

Телефон: 0991234567

1. Додати контакт

2. Показати контакти

3. Вийти

Виберіть дію: 2

Контакти:

Олексій: 0991234567

Завдання 3. "Динамічна база даних"

● Створіть програму, яка створює базу даних клієнтів у форматі JSON.

(можна використати MongoDB).

● Додайте можливість пошуку, оновлення та видалення записів.

● Забезпечте збереження змін у JSON-файлі після кожної операції.

Код програми:

import json

class CustomerDB:

def init(self, db\_file='customers.json'):

self.db\_file = db\_file

try:

with open(db\_file, 'r') as f:

self.customers = json.load(f)

except FileNotFoundError:

self.customers = {}

def save(self):

with open(self.db\_file, 'w') as f:

json.dump(self.customers, f, indent=4)

def add\_customer(self, customer\_id, name, email):

self.customers[customer\_id] = {'name': name, 'email': email}

self.save()

def find\_customer(self, customer\_id):

return self.customers.get(customer\_id)

def update\_customer(self, customer\_id, name=None, email=None):

if customer\_id in self.customers:

if name:

self.customers[customer\_id]['name'] = name

if email:

self.customers[customer\_id]['email'] = email

self.save()

return True

return False

def delete\_customer(self, customer\_id):

if customer\_id in self.customers:

del self.customers[customer\_id]

self.save()

return True

return False

def task3\_example():

db = CustomerDB()

db.add\_customer('001', 'Іван Петренко', 'ivan@example.com')

db.add\_customer('002', 'Марія Сидоренко', 'maria@example.com')

db.update\_customer('001', email='ivan.p@example.com')

db.delete\_customer('002')

Приклад роботи програми:

db = CustomerDB()

db.add\_customer('001', 'Іван Петренко', 'ivan@example.com')

db.add\_customer('002', 'Марія Сидоренко', 'maria@example.com')

print(db.find\_customer('001'))

db.update\_customer('001', email='ivan.p@example.com')

db.delete\_customer('002')

Висновки

У ході виконання практичної роботи було опановано роботу з серіалізацією даних у Python за допомогою модуля json. Реалізовано три програми, які демонструють різні аспекти роботи з JSON:

1. Систему збереження і оновлення ігрової статистики

2. Менеджер контактів з можливістю збереження та завантаження даних

3. Просту базу даних клієнтів з CRUD-функціоналом

Кожна програма коректно зберігає дані у JSON-файли та відновлює їх при наступному запуску. Робота демонструє ефективне використання JSON для зберігання структурованих даних у Python.