**Звіт до лабораторної роботи №2**

**Завдання 1**

**Мета роботи:**

Розробити програму, яка:

1. Відкриває текстовий файл **data.txt** для читання.
2. Підраховує кількість рядків у файлі.
3. Виводить кожен рядок разом із кількістю слів у ньому.
4. Записує результати у новий файл **results.txt**.

**Опис програми:**

Програма *reader.py* реалізує функцію **processfile**, яка:

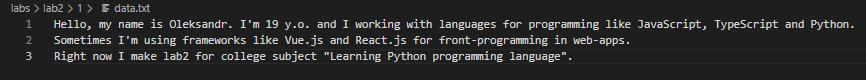
* Приймає назви вхідного та вихідного файлів як аргументи.
* Виконує обробку файлу згідно з вимогами завдання.

**Кроки виконання**

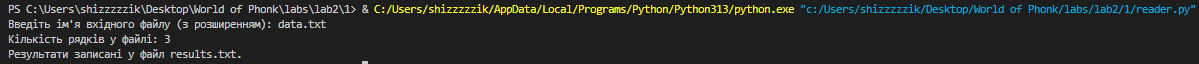
1. **Відкриття файлу для читання**:
   * За допомогою **with open(input\_filename, 'r', encoding='utf-8') as infile** програма відкриває файл для читання. Використання **encoding='utf-8'** забезпечує коректне відображення тексту в разі наявності нестандартних символів.
   * Усі рядки зчитуються за допомогою **infile.readlines()** і зберігаються у списку **lines**.
2. **Підрахунок кількості рядків**:
   * Програма визначає кількість рядків за допомогою **len(lines)** і виводить це значення в термінал.
3. **Обробка кожного рядка та підрахунок слів**:
   * За допомогою циклу **for i, line in enumerate(lines, start=1)** програма проходить кожен рядок, де **i** – номер рядка.
   * Кількість слів у рядку визначається за допомогою **len(line.split())**, де **line.split()** розділяє рядок на слова за пробілами.
   * Результат додається до списку *results* у форматі **"Рядок {i}: {line.strip()} (Кількість слів: {word\_count})\n"**.
4. **Запис результатів у файл results.txt**:
   * Програма відкриває новий файл **results.txt** для запису.
   * Спочатку записується загальна кількість рядків, а потім у кожному рядку виводиться текст разом із кількістю слів.
   * Програма виводить повідомлення про успішний запис результатів.
5. **Обробка помилок**:
   * У випадку, якщо вхідний файл не знайдено, програма виводить повідомлення про помилку *FileNotFoundError*.
   * Для інших можливих помилок додається загальне виключення **Exception**, яке виводить текст помилки.

**Тестування програми**

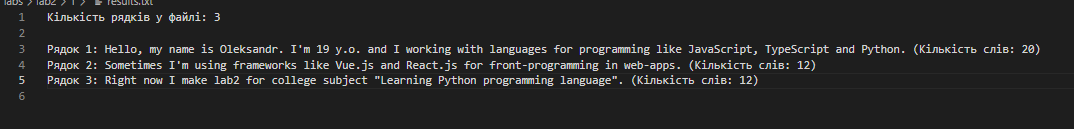
**data.txt**



**Запуск програми reader.py**



**results.txt**



**Висновок**

У процесі виконання завдання я розробив програму, яка обробляє текстовий файл: зчитує дані, аналізує їх, і записує результати в новий файл. Програма коректно обробляє випадки відсутності файлу, а також форматує вихідні дані, що дає змогу отримати зручний для читання файл результатів.

**Завдання 2**

**Мета роботи**

Розробити програму, яка:

1. Виконує GET-запит до API сервісу JSONPlaceholder для отримання списку постів.
2. Зберігає отримані дані у форматі JSON у файл posts.json.

**Опис програми**

Програма f**etch\_and\_save\_posts.py** використовує бібліотеки **requests** для виконання HTTP-запитів та **json** для збереження отриманих даних у форматі **JSON**. Головна функція програми – **fetch\_and\_save\_posts**, що приймає *URL* для запиту та ім’я файлу для збереження даних.

**Кроки виконання**

1. **Виконання GET-запиту:**
   * Програма ініціює запит **GET** до *API* сервісу за допомогою **requests.get(url)**.
   * Після виконання запиту перевіряється статус-код відповіді. Якщо запит успішний (*код 200*), дані передаються далі для обробки.
2. **Обробка отриманих даних:**
   * Якщо запит завершився успішно, програма використовує метод response.json() для отримання даних у форматі JSON. Це список об'єктів, де кожен об’єкт представляє окремий пост з полями, такими як userId, id, title та body.
3. **Збереження даних у файл:**
   * Отримані дані зберігаються у файл **posts** у форматі *JSON*, використовуючи **json.dump**.
   * Аргумент **ensure\_ascii=False** дозволяє зберігати текст у вигляді *Unicode*, а **indent=4** забезпечує читабельне форматування *JSON* із відступами.
4. **Обробка помилок:**
   * Якщо статус-код відповіді відрізняється від 200, програма виводить повідомлення про невдалий запит разом із кодом статусу.
   * Загальне виключення **Exception** обробляє будь-які інші помилки, що можуть виникнути під час виконання програми.

**Тестування програми**

**Запуск програми fetch\_and\_save\_posts.py**



**Збережені дані отримані з GET запиту у posts.json**Зображення, що містить знімок екрана, текст

Автоматично згенерований опис

**Висновок**

У процесі виконання цього завдання я навчився:

1. Виконувати HTTP-запити за допомогою бібліотеки requests.
2. Перетворювати отриману відповідь у формат JSON та зберігати її у файл.
3. Обробляти можливі помилки при виконанні запитів та збереженні файлів.

Програма коректно реалізує отримання даних з API та їх збереження, що робить її зручною для отримання та архівування інформації.