**Міністерство освіти й науки України**

**Львівський національний університет імені Івана Франка**

Факультет електроніки та комп’ютерних технологій

з предмета: ***Комп‘ютерна лінгвістика***

Звіт

про виконання лабораторної роботи № 1

**«Препроцесинг текстових документів»**

Виконав:

Студент групи

ФеС-32с

Бойко Кирило

Львів 2024

***Завдання***

використовуючи видані програми, здійснити попереднє опрацювання окремих текстів і текстових баз різними мовами; порівняти функціонал та інтерфейс різних програм, їхні переваги та недоліки; дослідити залежність часу опрацювання великих текстових баз від їхнього розміру.

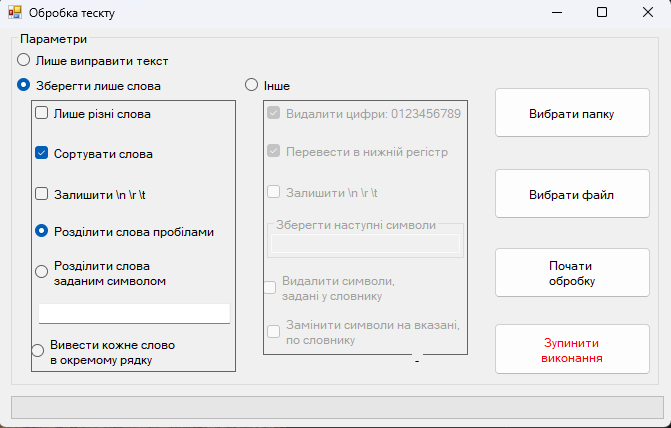
**Хід роботи**

Перед початком виконання лабораторної роботи, я обрав одну текстову базу, та один досліджуваний текст з архіву **main text corora2023-24.zip**

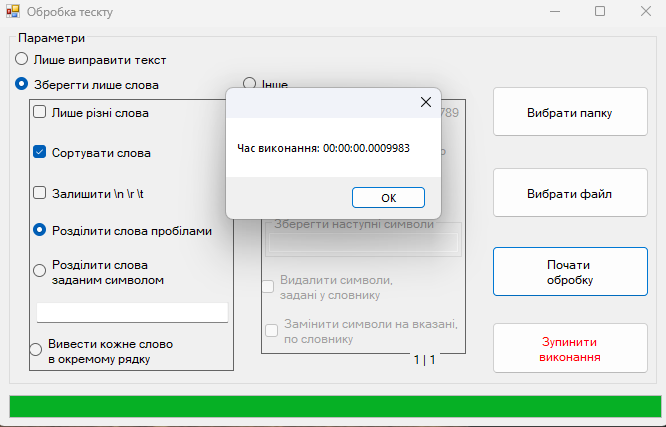
***Clemencia Novela de costumbres by Fernán Caballero***

***DONALD J. TRUMP January 20, 2017***

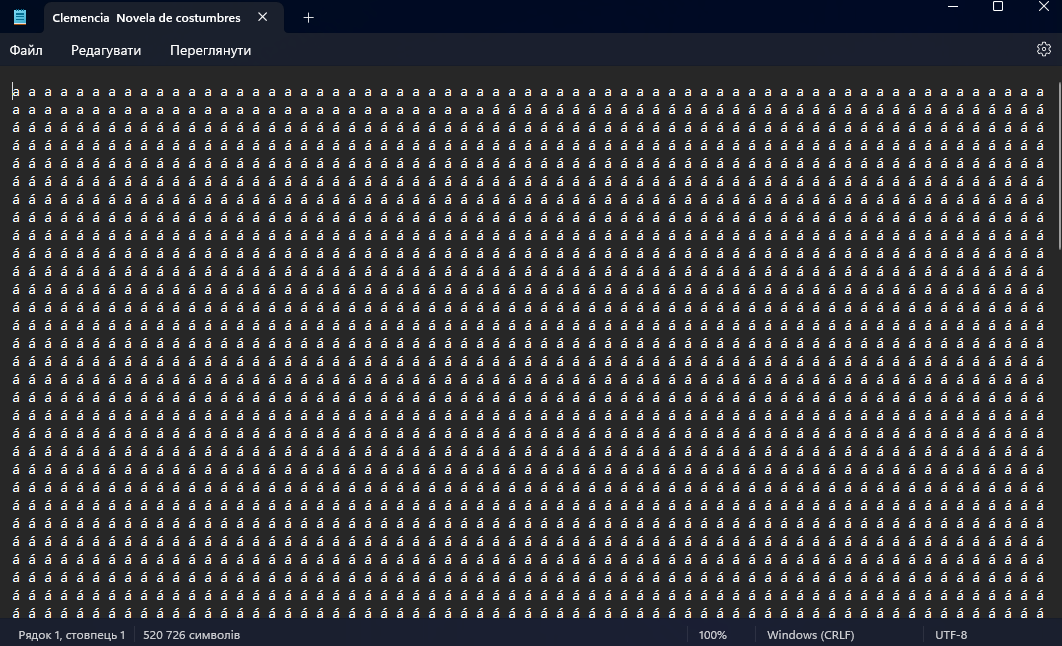
Я ініціював програму **+Text cleaner&processor (main)** для обробки тексту, завантажив файл **Clemencia Novela de costumbres by Fernán Caballero.txt** і встановив такі параметри для обробки:

**Інтерфейс утиліти

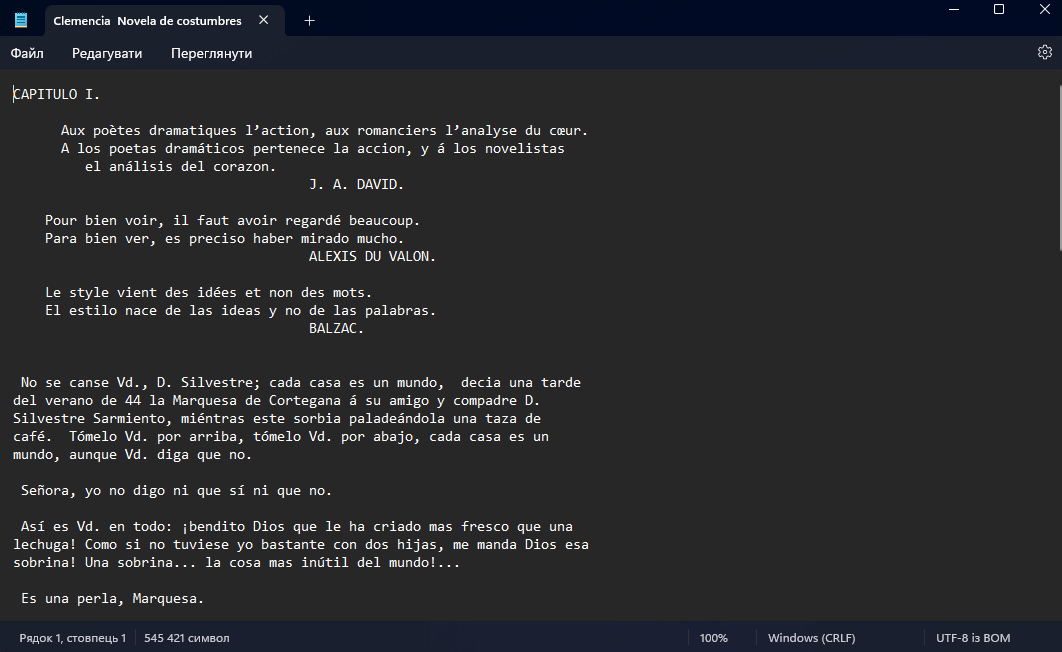
**Результати**

****

Інтерфейс утиліти після виконання роботи

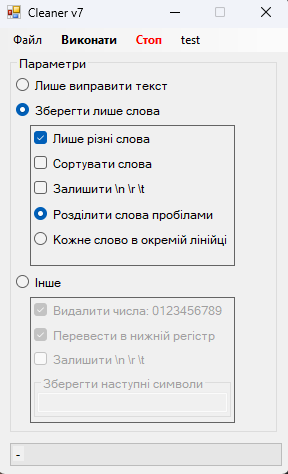
****

Оброблений файл



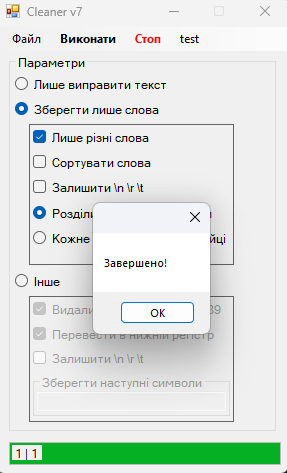
Оригінальний текстовий файл

Тепер виконую ті самі дії в програмі **+Text cleaner(for English only)** з файлом ***DONALD J. TRUMP January 20, 2017.txt*** та вибраними параметрами:

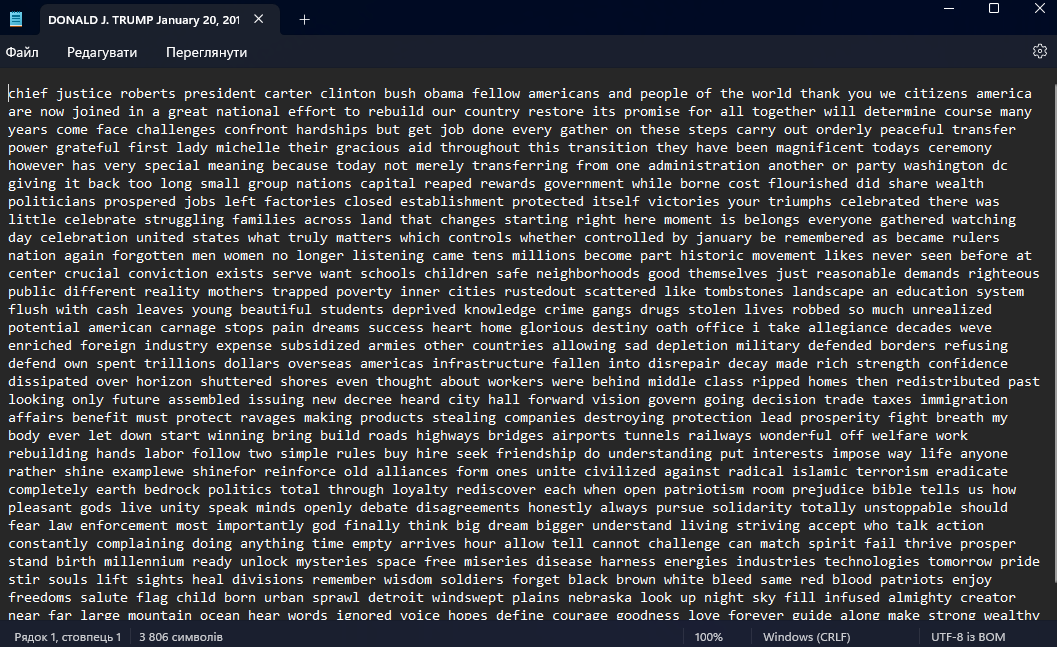
**

Інтерфейс програми та вибрані параметри

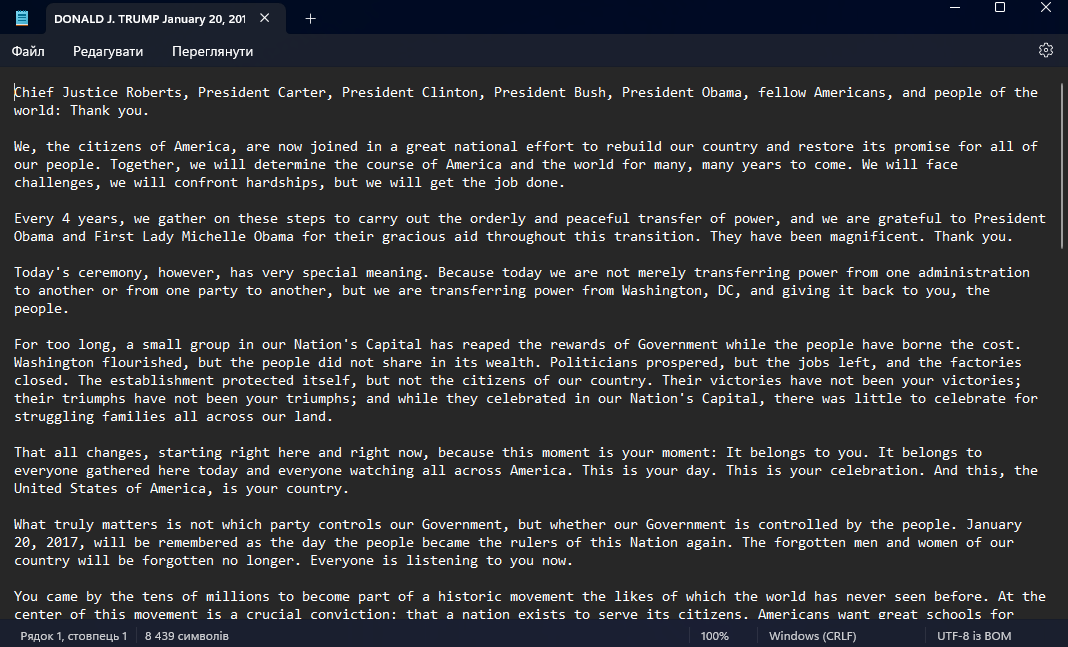
**Результати**



Результат виконання роботи



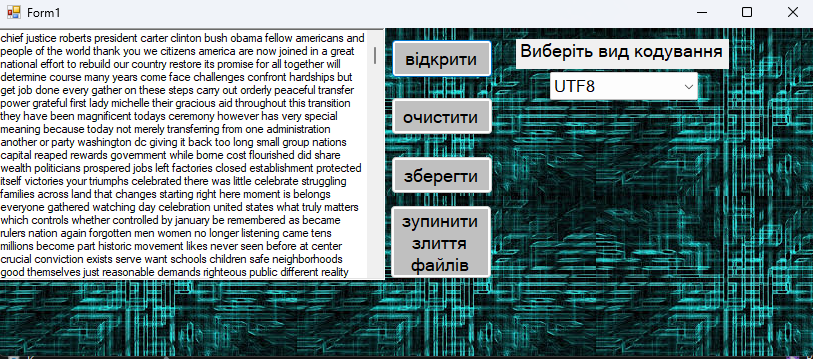
Текст після проведення чистки в програмі

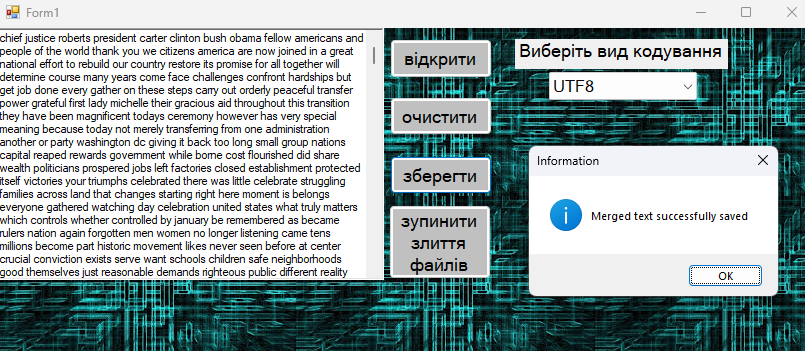


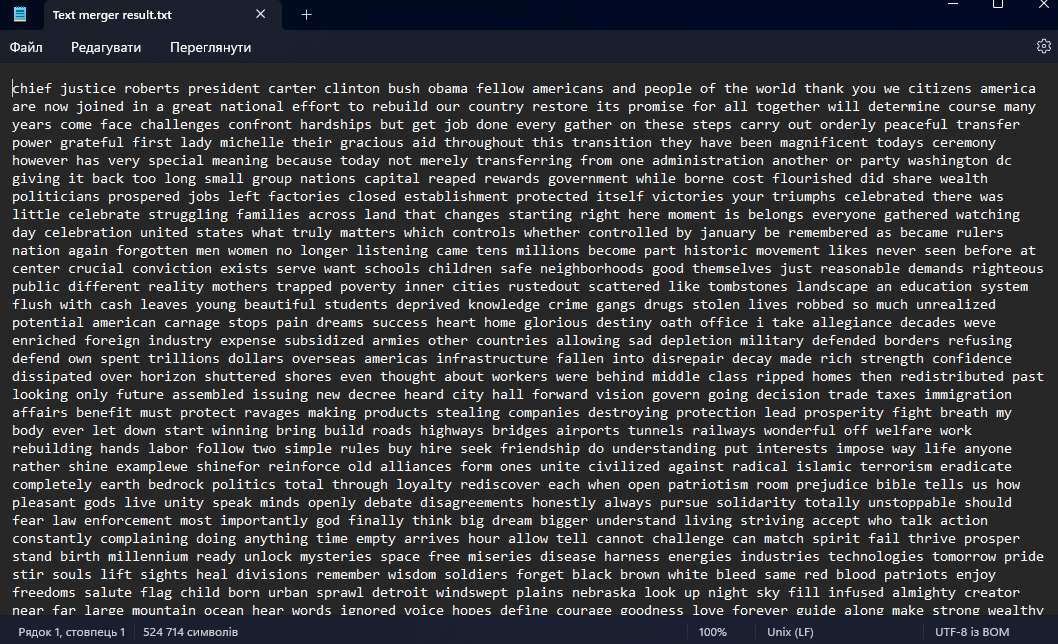
Оригінальний текст

Щоб об'єднати два текстові файли в один, я запускаю програму **+Text merger2022** та вибираю попередньо оброблені тексти

**Результати**

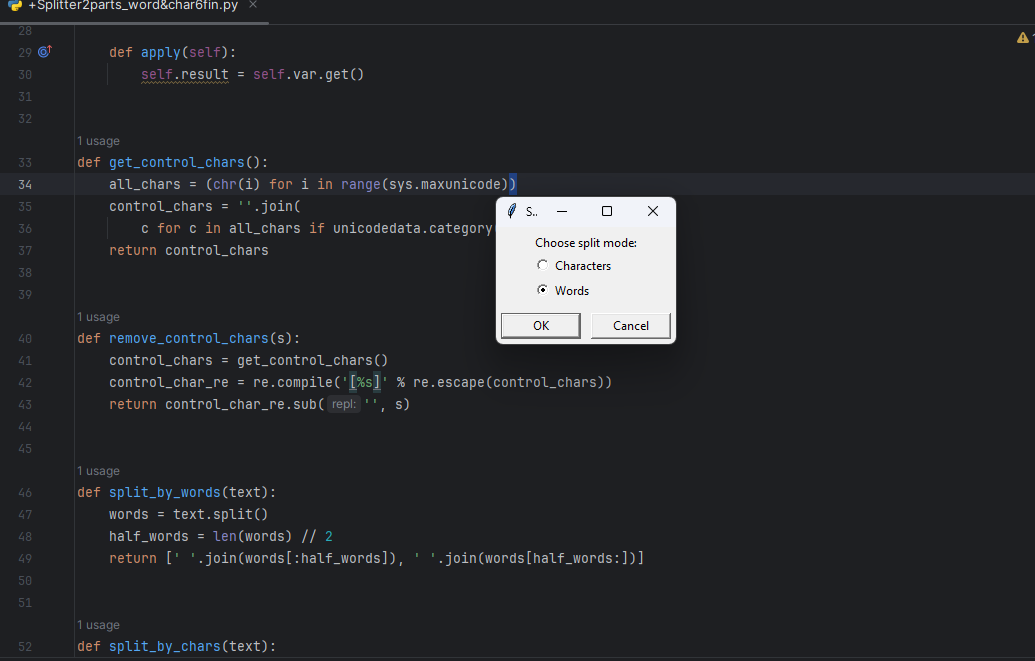
****Інтерфейс програми

****Збереження результату

****

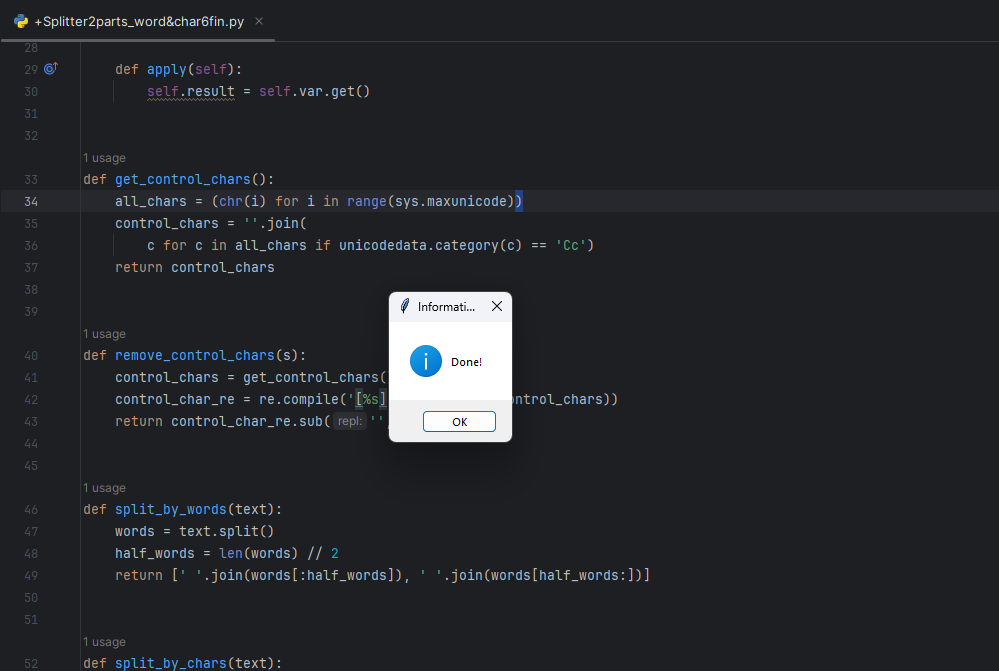
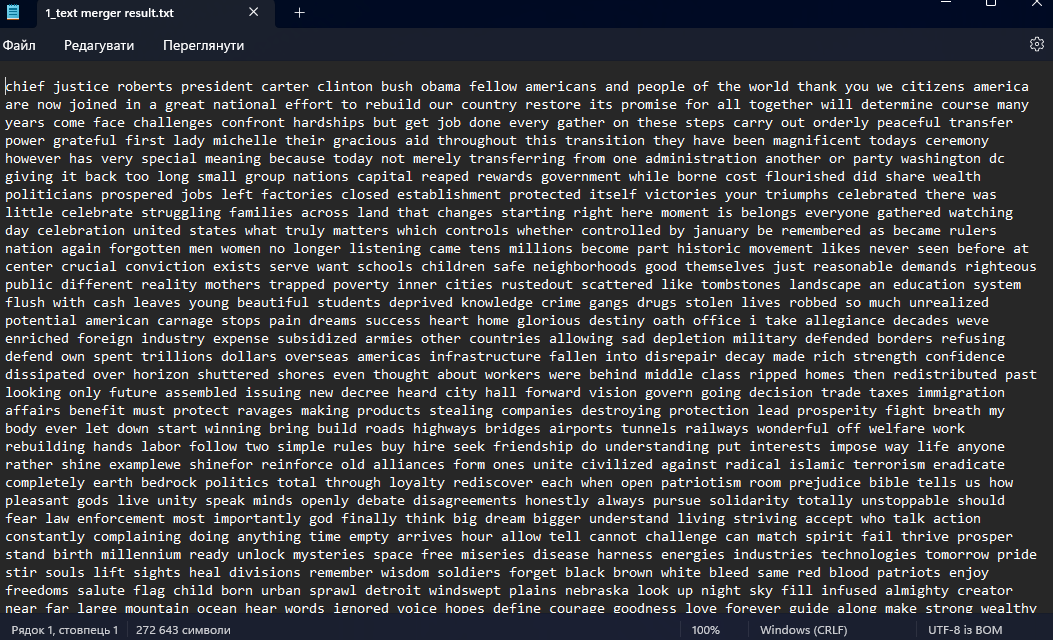
Результат злиття двох текстів

Для поділу двох файлів я відкрив програму **+TextSplitter(2halves)**, вказав папку зі злитими файлами та вибрав місце для збереження результату.

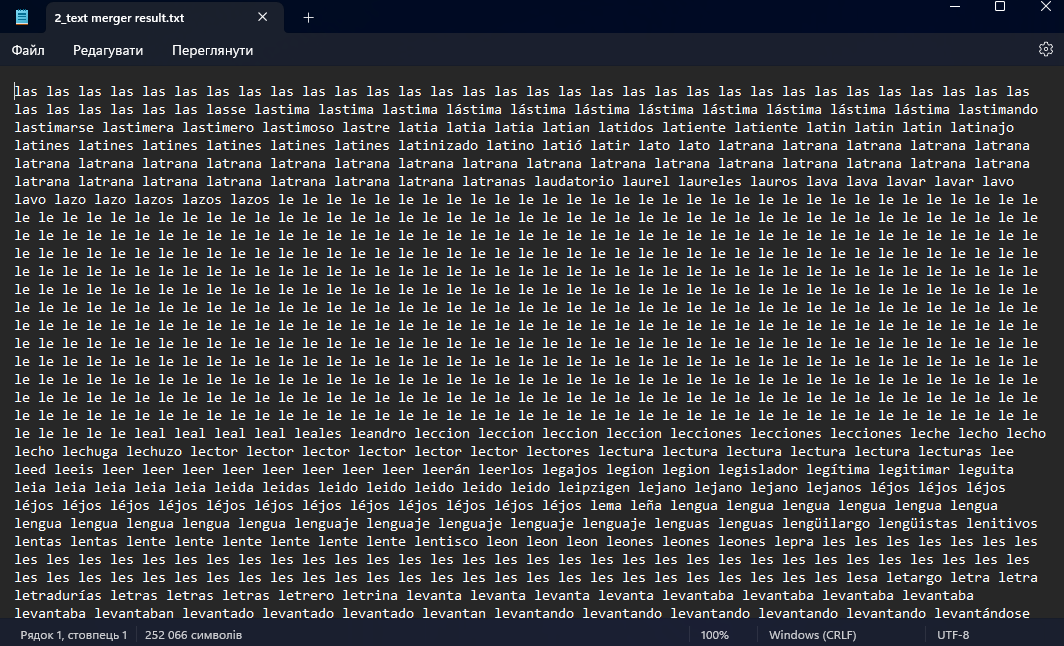


Інтерфейс програми

**Результати**

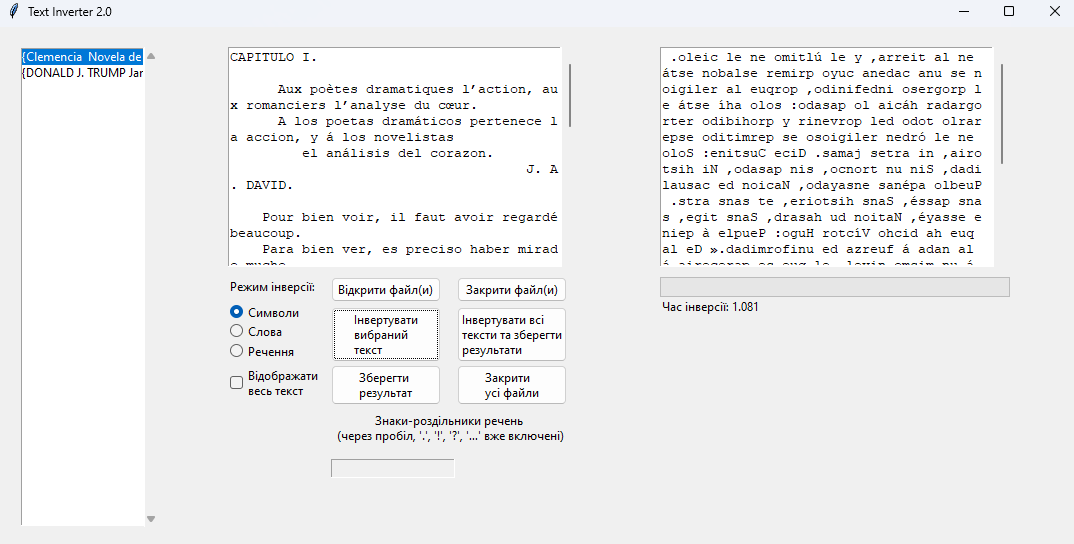
Результат розбиття 1 тексту



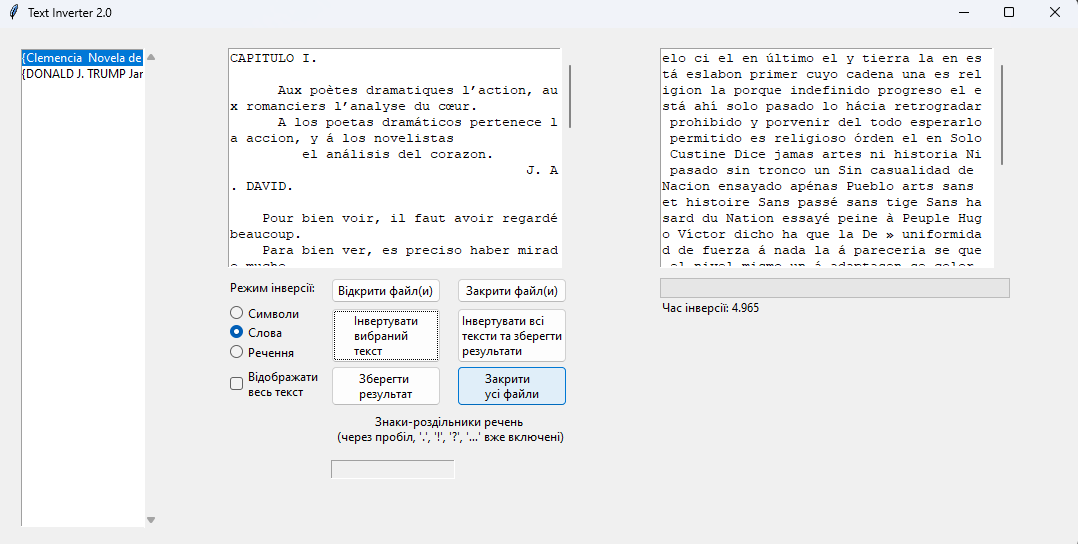
Результат розбиття 2 тексту

Я застосував програму **+TextInverter2О.Р.(py**) для інверсії тексту, послідовно змінюючи режими інверсії, щоб досягти необхідного результату

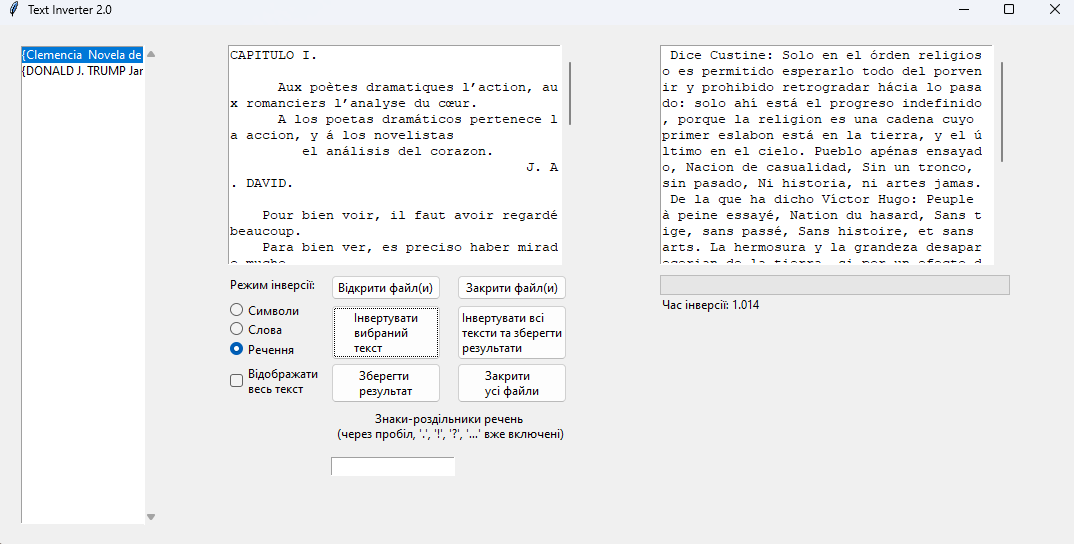
**Результати**



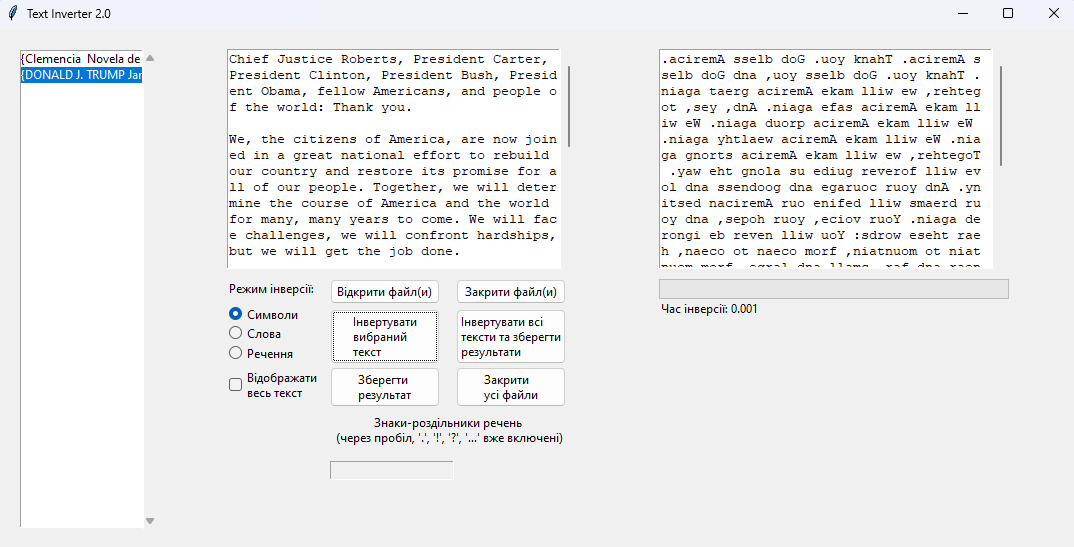
Виконую інверсію першого тексту в режимі "символи"



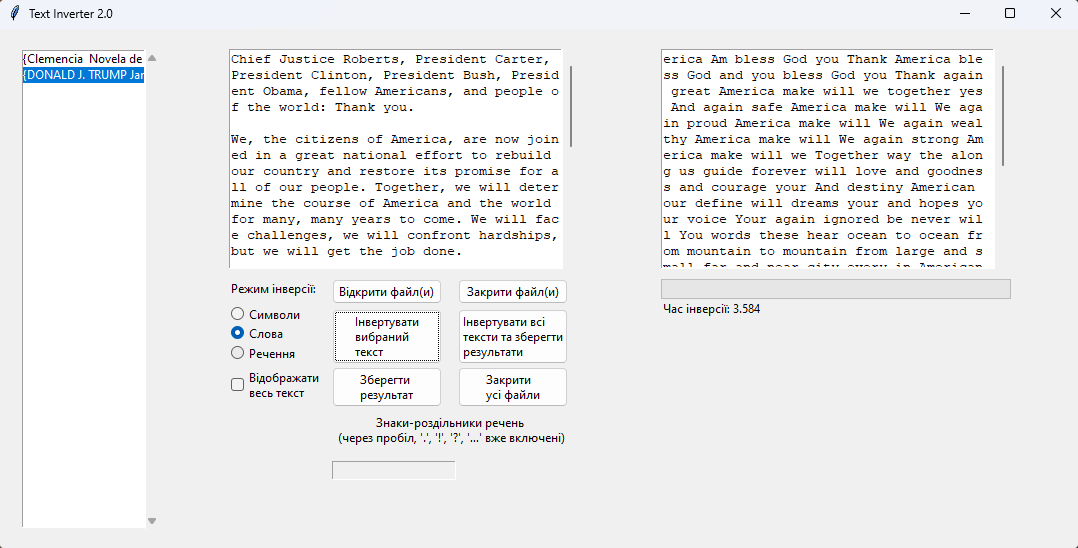
Виконую інверсію першого тексту в режимі "слова"



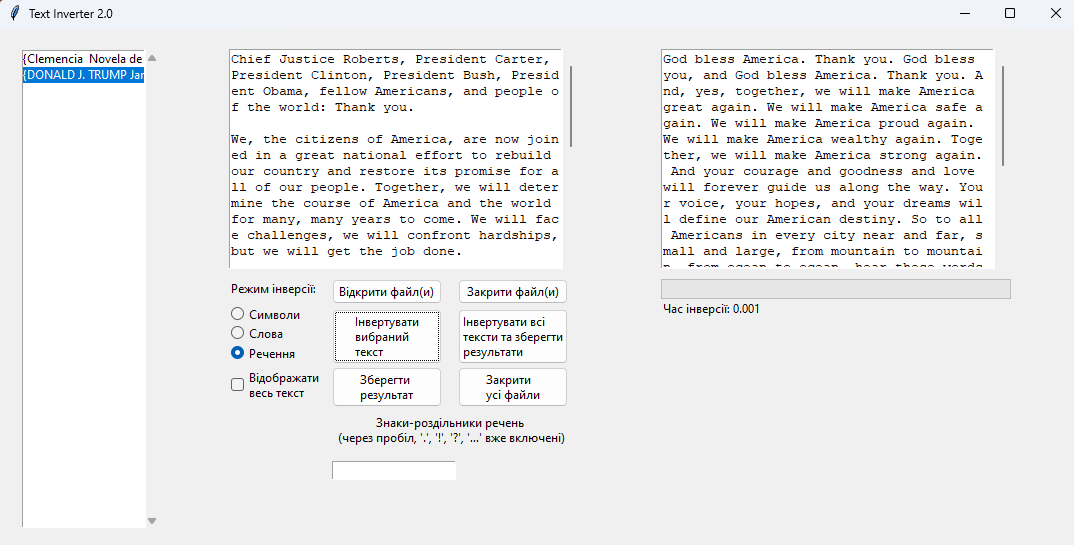
Виконую інверсію першого тексту в режимі "речення"



Виконую інверсію другого тексту в режимі "символи"

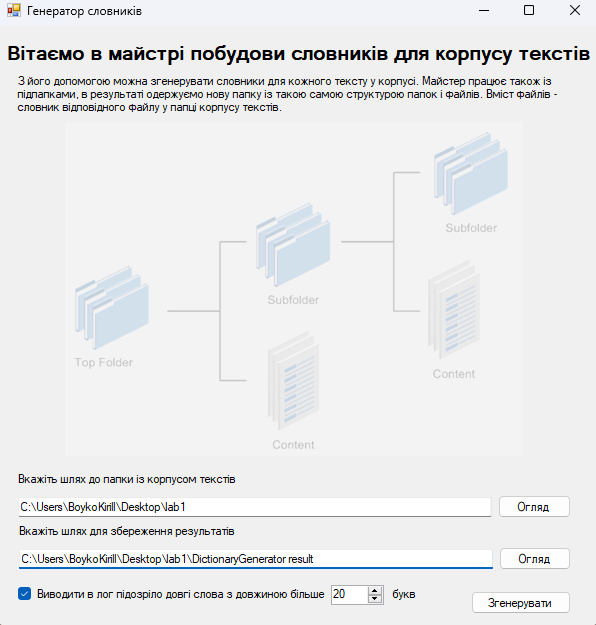


Виконую інверсію другого тексту в режимі "слова"

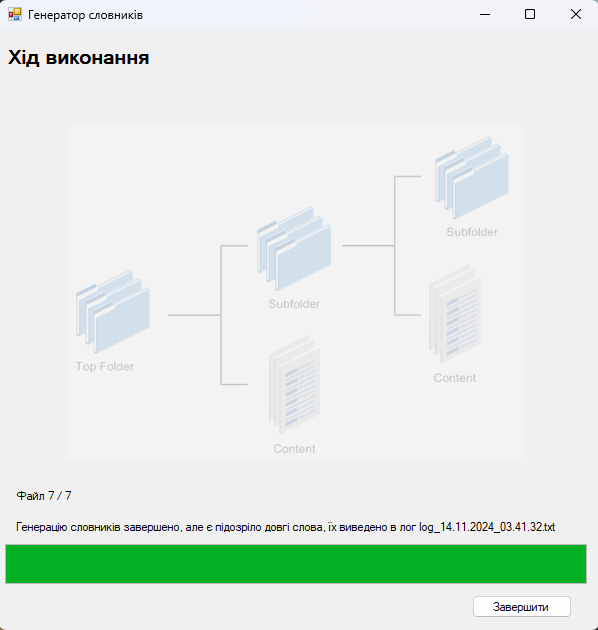


Виконую інверсію другого тексту в режимі "речення"

Для створення словника я використав утиліту **DictionaryGenerator**

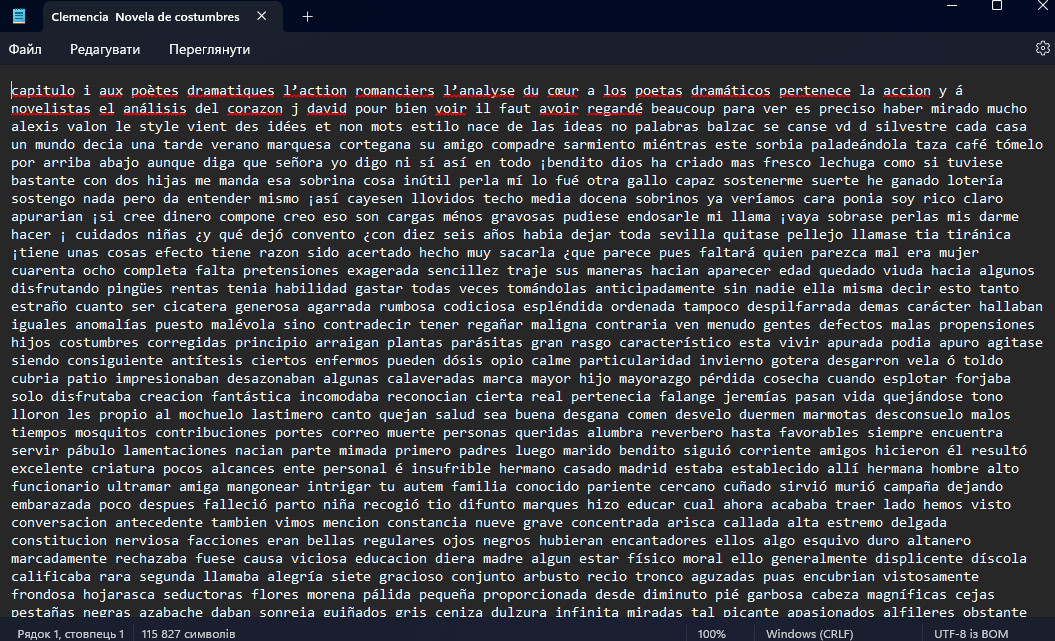
****

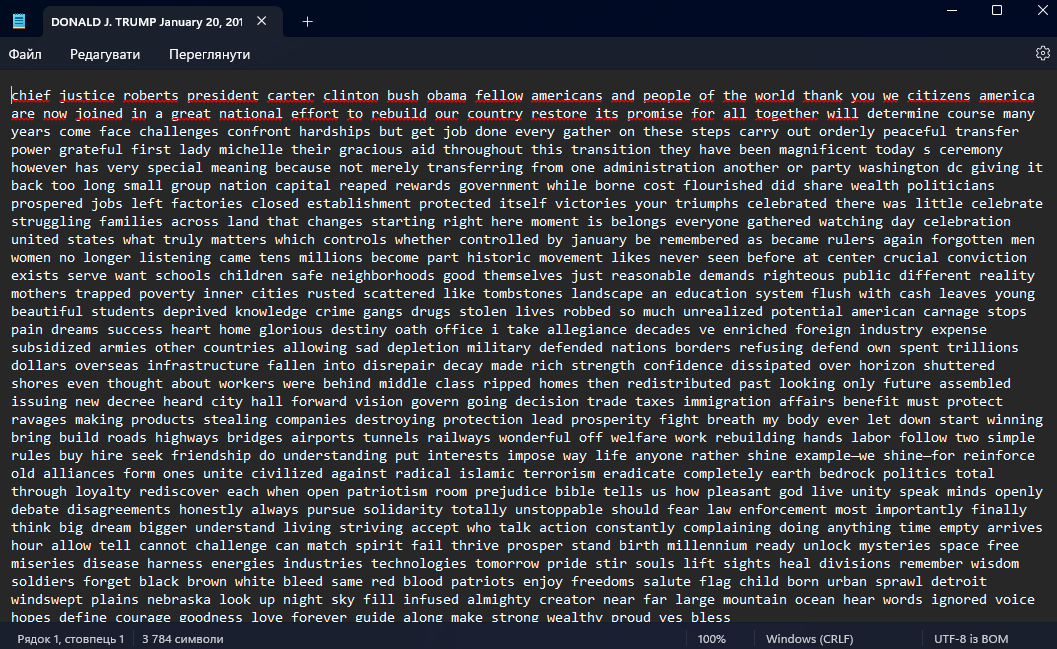
Інтерфейс програми



Інтерфейс програми вже з виконаною роботою

**Результати**





Зменшуючи розмір файлу поступово, я отримав відповідні результати та розробив програму для побудови графіків залежностей: t(L), log t(L), t(log L) і log t(log L) на основі зібраних даних

**Розмір файлу (L, МБ) Час обробки (t, секунди)**

**1.5 0.07**

**0.8 0.02**

**0.2 0.004**

**Код програми:**

import matplotlib.pyplot as plt

import numpy as np

sizes = np.array([1.5, 0.8, 0.2])

times = np.array([0.07, 0.02, 0.004])

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

axs[0, 0].plot(sizes, times, 'o-', color='purple', label='t(L)', markersize=8)

axs[0, 0].set\_xlabel("Розмір файлу (L, МБ)")

axs[0, 0].set\_ylabel("Час обробки (t, с)")

axs[0, 0].set\_title("Графік t(L)")

axs[0, 0].legend()

axs[0, 1].plot(sizes, np.log(times), 'o-', color='orange', label='log t(L)', markersize=8)

axs[0, 1].set\_xlabel("Розмір файлу (L, МБ)")

axs[0, 1].set\_ylabel("log Часу обробки (log t)")

axs[0, 1].set\_title("Графік log t(L)")

axs[0, 1].legend()

axs[1, 0].plot(np.log(sizes), times, 'o-', color='cyan', label='t(log L)', markersize=8)

axs[1, 0].set\_xlabel("log Розміру файлу (log L)")

axs[1, 0].set\_ylabel("Час обробки (t, с)")

axs[1, 0].set\_title("Графік t(log L)")

axs[1, 0].legend()

axs[1, 1].plot(np.log(sizes), np.log(times), 'o-', color='brown', label='log t(log L)', markersize=8)

axs[1, 1].set\_xlabel("log Розміру файлу (log L)")

axs[1, 1].set\_ylabel("log Часу обробки (log t)")

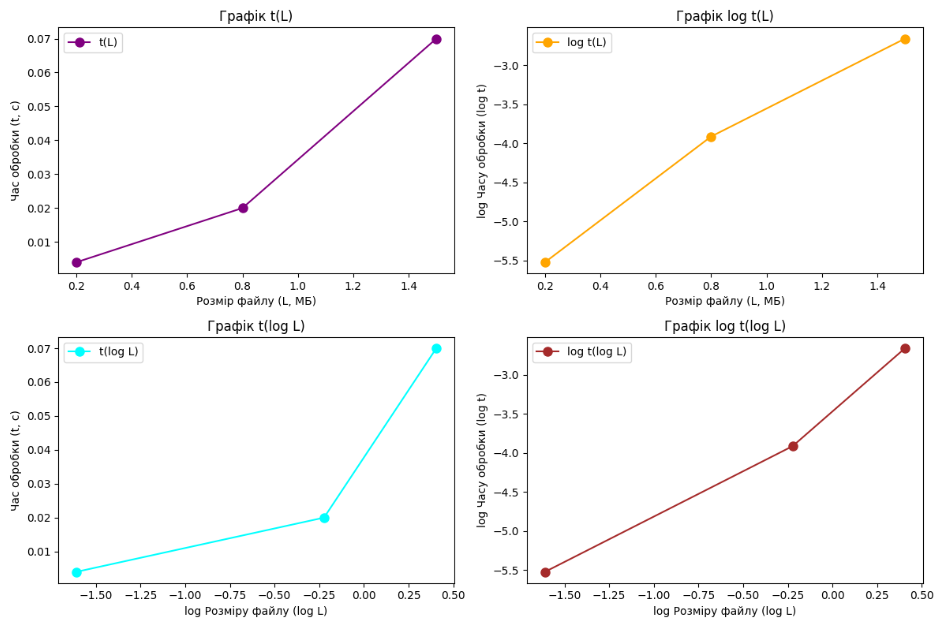
axs[1, 1].set\_title("Графік log t(log L)")

axs[1, 1].legend()

plt.tight\_layout()

plt.show()

**Результат**



**Висновок:** У цій лабораторній роботі я дослідив, як час обробки текстових файлів змінюється залежно від їхнього розміру при використанні обраної програми для препроцесингу текстів. Аналіз графіків показав, що з ростом розміру файлу час обробки зменшується, що можна пояснити особливостями алгоритму, який більш ефективно працює з більшими обсягами даних.