**Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформатики та програмної інженерії**

**Звіт**

з лабораторної роботи № 4 з дисципліни «Технології паралельних обчислень»

**«Розробка паралельних програм з використанням пулів потоків, екзекьюторів та ForkJoinFramework»**

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-02, Зусько Владислав Юрійович*

Київ 2023

**Завдання #1**

Побудуйте алгоритм статистичного аналізу тексту та визначте характеристики випадкової величини «довжина слова в символах» з використанням ForkJoinFramework. **20 балів.** Дослідіть побудований алгоритм аналізу текстових документів на ефективність експериментально. **10 балів.**

Почнемо побудову алгоритму. Для цього перш за все реалізуємо клас файлу та папки, які допоможуть нам зручно взаємодіяти з файловою системою. Додамо у клас документу метод отримання рядків файлу.

Ще один клас буде слугувати допоміжним у роботі з словами, а саме виокремлювати їх з документу.

Так як нам доведеться працювати зі статистичними показниками, було додано клас для обчислення величин, зокрема мат. сподівання, ймовірності, відхилення, дисперсії. Задля використання фреймворку нам знадобиться 3 види задач: задача для папки, для файлу та задача безпосереднього запуску фреймворку.

Метою задачі директорії буде проходження по піддерикторіях, запуск підзадач для документів та акумулювання зібраної з підзадач інформації. Таким чином ми в цьому місці використовуємо рекурсивне створення задач, що є ефективним методом взаємодії в межах використаного фреймворку.

Задача документа буде розділяти файлові рядки на слова та повертати колекцію довжин слів.

Задача запуску ж буде стартувати фреймворк та його кореневу задачу.

Для тестування побудуємо аналогічний алгоритм, що працюватиме послідовно та порівняємо результати виконання.

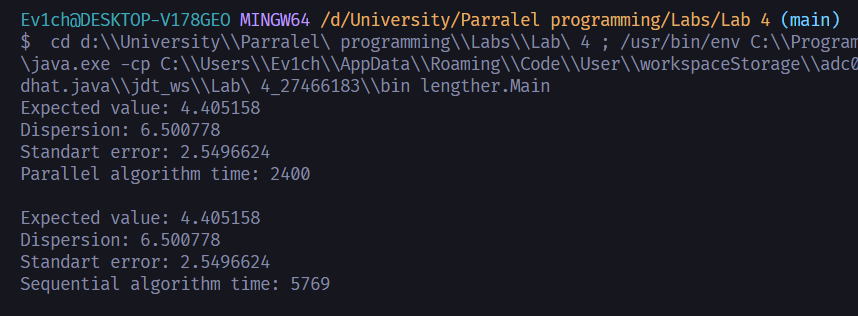


Рисунок – Результати виконання обох варіантів

Із замірів бачимо, що послідовний алгоритм в цьому випадку показав себе повільнішим ніж оптимізований більш ніж в 2 рази.

**Завдання #2**

Реалізуйте один з алгоритмів комп’ютерного практикуму 2 або 3 з використанням ForkJoinFramework та визначте прискорення, яке отримане за рахунок використання ForkJoinFramework. **20 балів.**

Як приклад було обрано задачу з банком з минулої лабораторної роботи. Для демонстрації деякі методи класу були видалені, залишено лише один метод трансферу грошей.

Далі, було реалізовано задачу та потік, які проводять трансфери з випадкових доступних рахунків на інші.

В межах тестування було заміряно час виконання обох видів паралелізації в однакових умовах та отримано наступний результат.

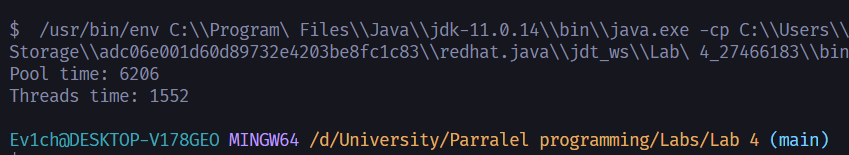


Рисунок – Результати тестування

Спираючи на результати, бачимо, що ForkJoinFramework у цій задачі виконувався довше, ніж звичайна багатопоточна реалізація. З цього робимо висновок, що для використання цієї технології варто підбирати коректні задачі, в межах цей фрейморк покаже себе якнайкраще. У конкретному випадку прискорення вірогідно не було отримано через те, що задачі радше були залежними одна від одної та паралельними, також їх фактично не можна розбити на менші рекурсивні підзадачі.

**Завдання #3**

Розробіть та реалізуйте алгоритм пошуку спільних слів в текстових документах з використанням ForkJoinFramework. **20 балів.**

Для виконання поставленої задачі було розширено клас роботи зі словами, зокрема додано метод для екстракції унікальних слів з документа. Також, було розроблено 3 задачі: для роботи з документами, папками та задачу запуску фреймворку.

Задача документу виконувала збір унікальних слів поміж його змісту та повертала множину таких слів.

Директорійна задача рекурсивно запускала подібні собі задачі над піддерикторіями та документні задачі на вмістом папки. Опісля виконувався перетин множин унікальних слів з документів, аби залишити лише спільні.

Задача запуску відповідно ініціювала сам пул та віддавала йому на виконання кореневу задачу.

**Завдання #4**

Розробіть та реалізуйте алгоритм пошуку текстових документів, які відповідають заданим ключовим словам (належать до області «Інформаційні технології»), з використанням ForkJoinFramework. **30 балів.**

У цій задачі є структура завдань, що аналогічна попереднім завданням, проте є деякі зміни. Зокрема документна задача спрямована на обраховування кількості кожного ключового слова, що трапляється у файлі. Для спрощення реалізації даної задачі і зберігання списку ключових слів, на які потрібно реагувати, було використано множину.