**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО”**

**ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ**

****

**Кафедра інформатики та програмної інженерії**

**Звіт до лабораторної роботи №1**

**з курсу**

**«Мультипарадигменне програмування»**

*студента 2 курсу*

*групи ІТ-01*

Степаненка Анатолія Юрійовича

*Викладач:*

Очеретяний О. К.

**Київ – 2022**

Завдання

Практична робота складається із трьох завдань, які самі по собі є досить простими. Але, оскільки задача - зрозуміти, як писали код наші славні пращури у 1950-х, ми введемо кілька обмежень:

* Заборонено використовувати функції
* Заборонено використовувати цикли

Для виконання потрібно взяти мову, що підтримує конструкцію GOTO

## **Завдання 1:**

Обчислювальна задача тут тривіальна: для текстового файлу ми хочемо відобразити N (наприклад, 25) найчастіших слів і відповідну частоту їх повторення, упорядковано за зменшенням. Слід обов’язково нормалізувати використання великих літер і ігнорувати стоп-слова, як «the», «for» тощо. Щоб все було просто, ми не піклуємося про порядок слів з однаковою частотою повторень. Ця обчислювальна задача відома як **term frequency**.

Реалізація виконана мовою С. Також у репозиторії є makefile для зручної компіляції та тестування.  
Спочатку ми зчитуємо аргумент командного рядка, що передав користувач при виклику програми. Потім приводимо текст користувача до lovercase.  
Далі для кожного слова підраховуємо кількість символів, щоб виділити необхідну кількість пам’яті і записуємо слово до розділяючого символу. Після того, як це виконано для всіх слів, з масиву видаляються стоп-слова по типу “and”, “in”, “by”, тощо. Далі підраховуюється кількість кожного слова і відбувається запис у масив структур, що містять слово та кількість разів, коли він був використаний. Після цього слова виводяться за спаданням кількості їх використання, очищується пам’ять, і програма завершується.

## **Завдання 2:**

Тепер, нам потрібно виконати задачу, що називається словниковим індексуванням. Для текстового файлу виведіть усі слова в алфавітному порядку разом із номерами сторінок, на яких Ці слова знаходяться. Ігноруйте всі слова, які зустрічаються більше 100 разів. Припустимо, що сторінка являє собою послідовність із 45 рядків.

У моїй реалізації спочатку считується назва файла. Потім підраховується кількість слів у файлі, для виділення пам’яті у масиві структур з необхідними даними для кожного слова і каретка повертається на початок файлу. Далі у «циклі»(насправді goto indexing) для кожного рядка з файлу виконується частково код завдання 1, після чого у «циклі»(насправді goto countwords) дані обробляються для кожного слова і вносяться у масив структур wordnpages, що містить необхідні дані для кожного слова. Дані виводяться в алфавітному порядку, пам’ять очищується, і програма завершується.

**Висновок:**

У ході лабораторної работи, я реалізував 2 програми з використанням принципів імперативного програмування. Мова С ідеально підходить для даної задачі, адже є імперативною. Також були введені і виконані обмеження на використання циклів та функцій.