

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 № 1 з дисципліни
«Мультипарадигменне програмування»

«Імперативне програмування»

Виконав(ла)

IT-01 Гончаренко А. А.
(шифр, прізвище, ім'я, по батькові)

Перевірів

ас. Очеретяний О. К.
(прізвище, ім'я, по батькові)

Київ 2021

1. Завдання лабораторної роботи

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*.

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.

2. Опис використаних технологій

Для виконання роботи було використано бібліотеку `goto` мови програмування `python`. Так як динамічні структури даних заборонені, у коді списки створюються одразу з фіксованим розміром, що не змінюється під час роботи.

3. Опис програмного коду

[Посилання на GitHub](#)

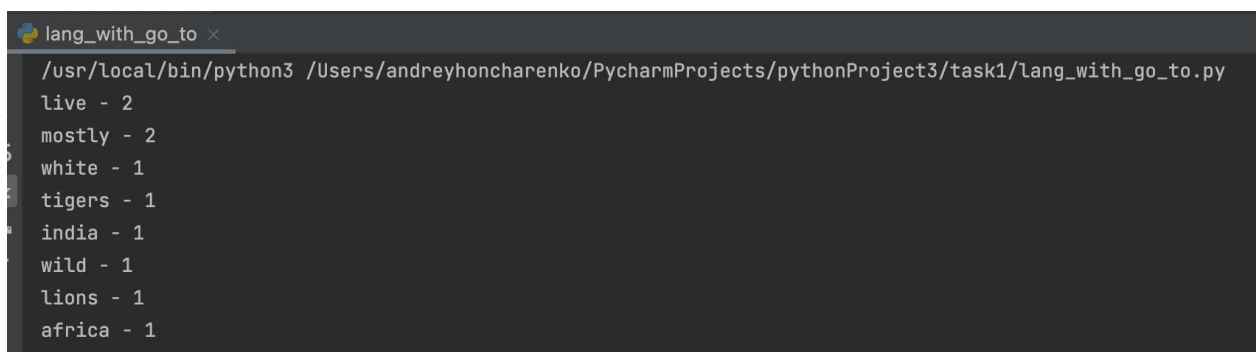
Для виконання першої задачі було створено такі функції: лінійний пошук у списку, підрахунок кількості слів у файлі, сортування бульбашкою, функція для підрахунку окремо кількості кожних слів та функція виводу відповіді у консоль. У функції окремого підрахунку ми зчитуємо файл по символу і додаємо цей символ у поточне слово. Якщо символ є пропуском або переносом строки, то ми перевіряємо чи не є це слово шумним, потім перевіряємо чи не зустрічалось воно вже в файлі (тобто чи не додали ми його в список слів), якщо його у списку немає додаємо та інкрементуємо відповідний елемент в списку підрахунку слів. Якщо слово вже зустрічалось

просто інкрементуємо відповідне йому число. Після цього очищаємо поточне слово та продовжуємо зчитувати символи до кінця файлу. Після цього ми сортуємо обидва масиви по частоті входження слова та виводимо результат функцією.

Для другої задачі все аналогічно, окрім того що список для підрахунку не одномірний, а двомірний. Для кожного слова він по першому індексу зберігає кількість сторінок на яких вже знайшли слово, а у наступних номера сторінок. Також робиться підрахунок строк у файлі, номер сторінки розраховується через цілочисленне ділення на кількість строк у сторінці.

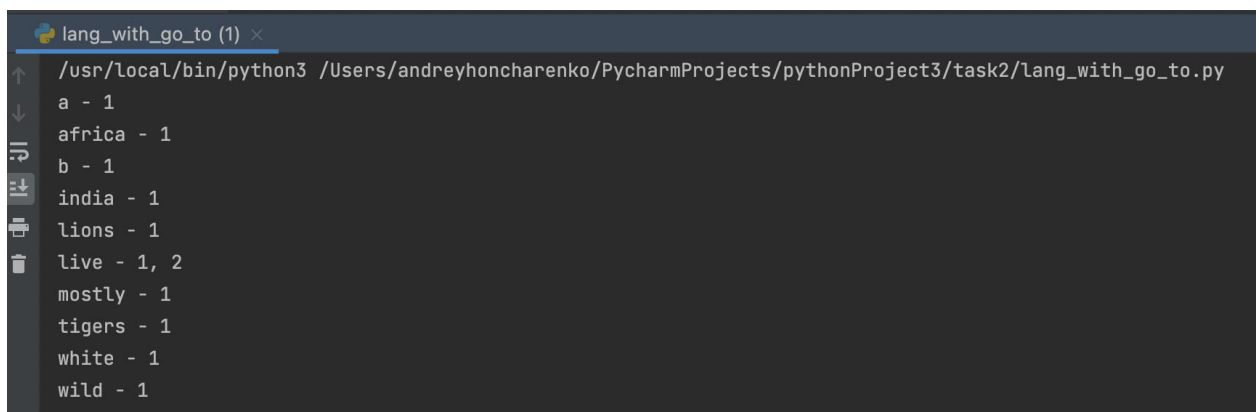
4. Скріншоти роботи програмного застосунку

Перша програма:



```
lang_with_go_to x
/usr/local/bin/python3 /Users/andreyhoncharenko/PycharmProjects/pythonProject3/task1/lang_with_go_to.py
live - 2
mostly - 2
white - 1
tigers - 1
india - 1
wild - 1
lions - 1
africa - 1
```

Друга програма:



```
lang_with_go_to (1) x
/usr/local/bin/python3 /Users/andreyhoncharenko/PycharmProjects/pythonProject3/task2/lang_with_go_to.py
a - 1
africa - 1
b - 1
india - 1
lions - 1
live - 1, 2
mostly - 1
tigers - 1
white - 1
wild - 1
```

5. Висновок

В ході даної лабораторної роботи було вирішено доволі прості задачі методами, якими їх вирішували у 1950-ті. Було вивчено роботу з goto.