

Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

Лабораторна робота №1
“Імперативне програмування”
з дисципліни
“Мультипарадигменне програмування”

Виконала
студентка групи ІТ-03:

Цуканова М.С.

Перевірив:

Очеретяний О.К.
Глушко Б.С.

Тема: імперативне програмування

Завдання:

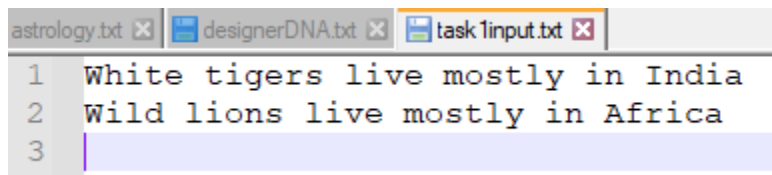
Завдання складається з двох задач. Заборонено використовувати функції та цикли. Для виконання лабораторної роботи було обрано мову програмування C#, оскільки вона підтримує конструкцію GOTO, як і потребується в завданні.

Задача 1

Завдання

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.

Склад вхідного файлу(мал.1):

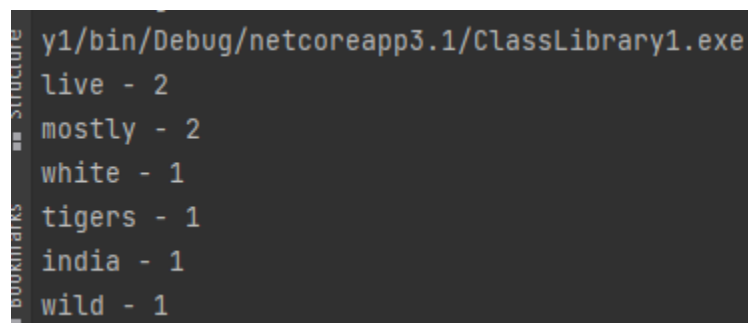


The screenshot shows a text editor with three tabs: 'astrology.txt', 'designerDNA.txt', and 'task1input.txt'. The 'task1input.txt' tab is active, displaying the following text:

```
1 White tigers live mostly in India
2 Wild lions live mostly in Africa
3
```

мал. 1, склад вхідного файлу

Результат роботи програми(мал.2):



The screenshot shows a command prompt window with the following output:

```
y1/bin/Debug/netcoreapp3.1/ClassLibrary1.exe
live - 2
mostly - 2
white - 1
tigers - 1
india - 1
wild - 1
```

мал. 2, результат роботи програми

Опис роботи програми:

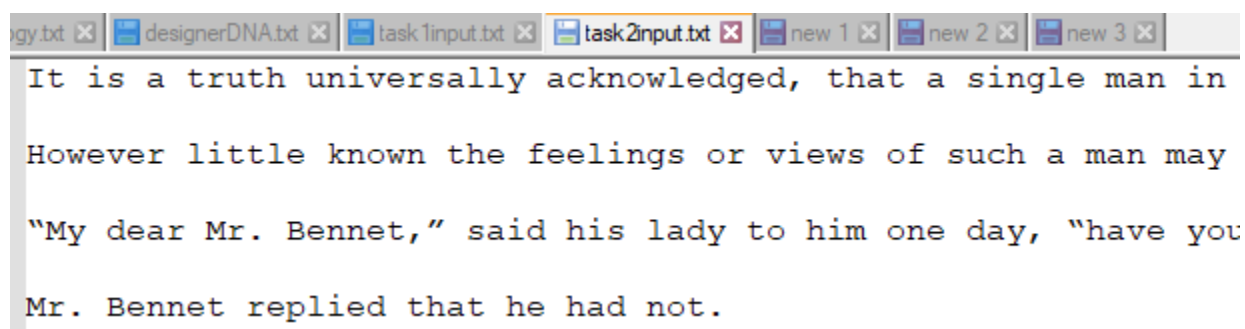
Для виконання цього завдання було використано `StreamReader` для зчитування файлу. Після зчитування файлу у масив, відбувається переведення символів в нижній регістр, де це потрібно. Символи мають тип `char`, отже, з таблиці ASCII, їх можна перевести в нижній регістр за допомогою додавання 32. Після переведення в нижній регістр, відбувається перевірка чи належить слово до стоп-слів. Список стоп-слів (частково) був взятий з ресурсу [за посиланням](#). Наступний крок - перевірка, чи слово зустрічалося раніше. Якщо так, то `counter` цього слова збільшується на одиницю. Якщо ні і масив слів заповнений, то створюється дублікат масиву і в кінець додається нове слово. Після закінчення роботи з додаванням слів у масив переходимо до його сортування. Для сортування масиву з цього завданні був обраний `Bubble Sort`. Чи потрібно сортувати масив визначається за допомогою масиву `counter`, а саме, якщо `counter[i] < counter[i + 1]`, то в масивах `counter` и `words` елементи з індексами `i(sort)` змінюються місцями. Останній крок - виведення результату. По одному виводяться значення масиву `words` та їх кількість (`counter`). Якщо кількість слів в масиві `words` більше 25 (з умови задачі) або досягнуто кінець масиву, вивід завершується.

Задача 2. Словникове індексування

Завдання:

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.

Склад вхідного файлу(мал.3):



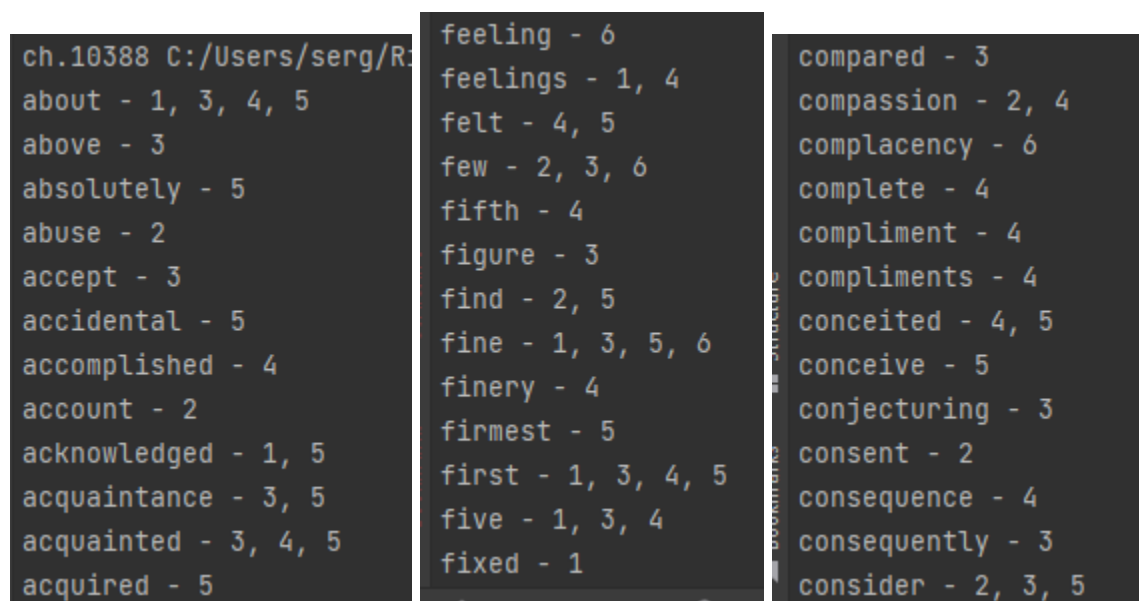
gy.txt x designerDNA.txt x task1input.txt x task2input.txt x new 1 x new 2 x new 3 x

It is a truth universally acknowledged, that a single man in
However little known the feelings or views of such a man may
"My dear Mr. Bennet," said his lady to him one day, "have you
Mr. Bennet replied that he had not.

мал. 3, склад вхідного файлу

Вхідний файл має довжину в 239 строк (6 сторінок)

Результат роботи (мал. 4-6):



```
ch.10388 C:/Users/serg/R
about - 1, 3, 4, 5
above - 3
absolutely - 5
abuse - 2
accept - 3
accidental - 5
accomplished - 4
account - 2
acknowledged - 1, 5
acquaintance - 3, 5
acquainted - 3, 4, 5
acquired - 5

feeling - 6
feelings - 1, 4
felt - 4, 5
few - 2, 3, 6
fifth - 4
figure - 3
find - 2, 5
fine - 1, 3, 5, 6
finery - 4
firmest - 5
first - 1, 3, 4, 5
five - 1, 3, 4
fixed - 1

compared - 3
compassion - 2, 4
complacency - 6
complete - 4
compliment - 4
compliments - 4
conceited - 4, 5
conceive - 5
conjecturing - 3
consent - 2
consequence - 4
consequently - 3
consider - 2, 3, 5
```

мал. 4-6, приклад виводу результату

Опис роботи програми:

Перший крок - ініціалізація змінних та масивів. На відміну від попереднього завдання для виконання цього завдання було використано три масиви counter, words та pages. Далі йде зчитування файлу, якщо кількість строк дорівнює 45(з умови задачі), то лічильник сторінок збільшується на 1. Наступний крок - перетворення символів в нижній регістр. Для цього було використано код з попереднього завдання, де для зміни регістру до char додавалось 32, де це

було необхідно. Далі відбувається перевірка чи є слово з масиви words. Механізм роботи такий самий, що і в попередньому завданні. Було додано перевірку на заповненість масиву pages, який при заповненні дублюється, збільшується та заповнюється даними, що вже існують. Для додавання нового слова збільшуються, дублюються масиви words, counter та pages. Наступний крок - сортування. Оскільки тепер для сортування слів не використовується їх кількість, а потрібно відсортувати їх за алфавітом. Для цього порівнюються перші символи слів і т.д., для швидшого сортування був використаний insertion sort. Останній крок - вивід результату. Для слів в яких counter менше 100 виводиться слово та його перша сторінка, далі через кому всі інші сторінки на яких воно зустрічається. Закінчення програми - досягнення кінця масиву.

Висновки:

У ході лабораторної роботи було розроблено дві програми в імперативній парадигмі програмування з використанням конструкції goto, яка слугувала заміною циклів. Також було мінімізовано використання функцій. Мова програмування C#. Серед розробки - Rider (JetBrains)