

Білько Олексій, ІП-02

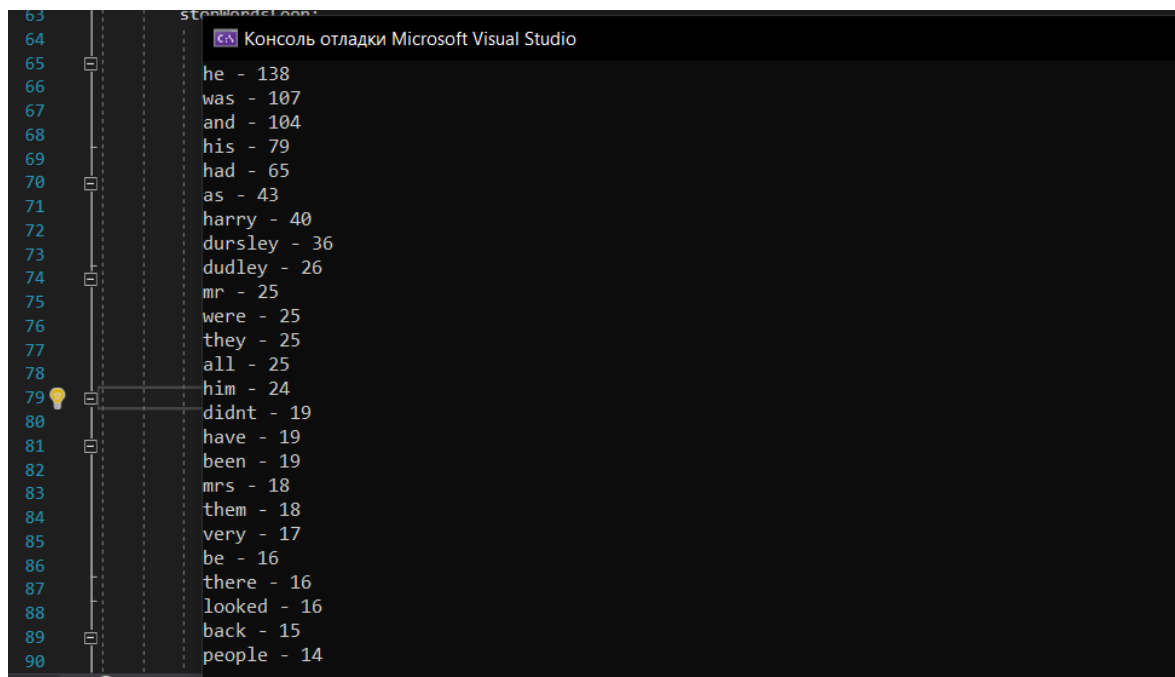
Для виконання даної лабораторної роботи було обрано мову C#

## 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**.

### Вхідний файл

Результат роботи програми:



```
63  
64  
65 he - 138  
66 was - 107  
67 and - 104  
68 his - 79  
69 had - 65  
70 as - 43  
71 harry - 40  
72 dursley - 36  
73 dudley - 26  
74 mr - 25  
75 were - 25  
76 they - 25  
77 all - 25  
78 him - 24  
79 didnt - 19  
80 have - 19  
81 been - 19  
82 mrs - 18  
83 them - 18  
84 very - 17  
85 be - 16  
86 there - 16  
87 looked - 16  
88 back - 15  
89 people - 14  
90
```

Алгоритм роботи програми:

1. Зчитати дані з текстового файлу в масив рядків
2. Пробігаючись по кожному слову, перевіряти, чи є воно стоп словом, якщо є – перейти до наступного
3. Якщо слово не є стоп словом перевірити чи є воно в масиві унікальних, якщо так, збільшити кількість в масиві чисел, де відповідна цифра відповідає відповідному слову, якщо ж ще не має, то додати його в масив унікальних і в масиві чисел поставити кількість 1
4. Відсортувати за спаданням слова по кількості їх входжень
5. Вивести на екран перші 25 слів і кількість їх входжень (або всі, якщо їх менше 25)

## 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.

### Вхідний файл

Результат роботи програми:

```
Program.cs task2.lang_with_go
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
93 %
Вывод
Показать выходные
```

able - 3, 5  
acting - 5  
adventure - 5  
affect - 3  
afternoon - 2  
afterward - 6  
again - 1, 6  
againthe - 3  
age - 4  
ago - 4  
air - 3, 4  
albus - 3, 3  
all - 1, 1, 2, 2, 2, 2, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6  
allowed - 4, 6  
almost - 2, 2, 2, 4, 4, 5  
already - 5  
also - 1, 2  
although - 1  
always - 1, 2, 4  
amount - 1  
angelharry - 5  
angrily - 2  
angry - 5, 6  
animals - 6  
another - 1, 2, 4, 5, 6  
anything - 1, 2, 3, 3, 3, 5, 5, 5, 6  
any - 1, 2  
anyone - 1, 3, 5  
anywhere - 1  
apart - 6  
apologized - 6  
appearance - 4  
appeared - 3, 3  
approve - 2  
arms - 4  
arm - 4  
arrived - 1, 1, 3, 5

Алгоритм роботи програми:

1. Зчитати дані з текстового файлу в масив рядків
2. Пробігаючись по кожному слову, перевіряти, чи є воно стоп словом, якщо є – перейти до наступного
3. Якщо слово не є стоп словом перевірити чи є воно в масиві унікальних, якщо так, додати номер сторінки в масив рядків (вважаємо, що одна сторінка – 45 рядків), де відповідний рядок с номерами сторінок відповідає відповідному слову, якщо ж ще не має, то додати його в масив унікальних і в масиві рядків з сторінками додати сторінку, на якій відповідне слово розташоване
4. Відсортувати слова по алфавіту
5. Вивести на екран результат роботи у вигляді [Слово – номер сторінки, номер сторінки]

Для виконання даних завдань були використані: вбудована функція `string.ToCharArray` і `string.Split` для приведення типу `string` до типу `char[]` і розділення рядка на слова по пробілу відповідно, функція `int.ToString` для приведення типу `int` в тип `string`, а також клас `StreamReader` для зчитування інформації з текстового файлу.