Міністерство освіти і науки України
Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Програмування інтелектуальних інформаційних систем

ЗВІТ

до лабораторної роботи №1
з дисципліни “Мультипарадигменне програмування”

Виконав студент ІТ-01 Задніпрянець Артур Артурович _____
(№ групи, прізвище, ім’я, по батькові)

Прийняв ас. Очеретяний О. К. _____
(посада, 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 вирішення задачі застосовано мову програмування Python та IDE PyCharm. Підключено бібліотеку goto та використано декоратор @with_goto

3. Опис програмного коду

Завдання 1:

```python
from goto import with_goto


@with_goto
def main():
    n = 25
    i = 0
    temp = 0
    max_val = 0
    d = dict()
    stop_words = ["i", "me", "my", "myself", "we", "our", "ours", "ourselves", "you", "you're", "you've", "you'll",
                  "you'd", "wouldn't",
                  "your", "yours", "yourself", "yourselves", "he", "him", "his", "himself", "she", "she's", "her",
                  "hers", "herself",
                  "it", "it's", "its", "itself", "they", "them", "their", "theirs", "themselves", "what", "which",
                  "who", "whom", "this",
                  "that", "that'll", "these", "those", "am", "is", "are", "was", "were", "be", "been", "being", "have",
                  "has", "had", "doesn't",
                  "having", "do", "does", "did", "doing", "a", "an", "the", "and", "but", "if", "or", "because", "as",
                  "until", "while",
                  "of", "at", "by", "for", "with", "about", "against", "between", "into", "through", "during", "before",
                  "after", "above",
                  "below", "to", "from", "up", "down", "in", "out", "on", "off", "over", "under", "again", "further",
                  "then", "once", "here",
                  "there", "when", "where", "why", "how", "all", "any", "both", "each", "few", "more", "most", "other",
                  "some", "such", "no",
                  "nor", "not", "only", "own", "same", "so", "than", "too", "very", "s", "t", "can", "will", "just",
                  "don", "don't", "should",
                  "should've", "now", "d", "ll", "m", "o", "re", "ve", "y", "ain", "aren", "aren't",
                  "couldn't", "didn't",
                  "doesn't", "hadn't", "hasn't", "haven't", "isn't",
                  "mightn't",
                  "mustn't", "needn't", "shan't", "shouldn", "shouldn't", "wasn't",
                  "weren't", "won't"]
    symbols = [',', '.', '/', '!', '?', ':', ';', '-', '"', "'"]

    # Read the file
    with open('text.txt', 'r') as f:
        text = f.read() + "$#"
```

```
    # Delete symbols from text
    label .drop_symbols_start
    if text[i] in symbols:
        text = text[:i] + text[i + 1:]
    if text[i] == "$":
        goto.drop_symbols_end
    else:
        i += 1
        goto.drop_symbols_start
    label.drop_symbols_end

    # Convert all characters to lower case
    i = 0
    label .to_lower
    if 64 < ord(text[i]) < 91:
        text = text[:i] + chr(ord(text[i]) + 32) + text[i+1:]

    i += 1
    if text[i] == "$":
        goto.to_lower_end
    goto .to_lower
    label .to_lower_end
    i = 0

    # Creating the dictionary with words and their number
    label.first
    if text[i] == " " or text[i] == "$" or text[i] == "\n":
        if text[temp:i] not in stop_words:
            if d.get(text[temp:i]):
                d[text[temp:i]] += 1
            else:
                d[text[temp:i]] = 1
        temp = i + 1
    i += 1
    if text[i] == "#":
        goto.end
    goto.first
    label.end
```

```
79
80      # Searching for max value
81      i = len(d)
82      values = list(d.values())
83      label .max_start
84      if max_val < values[i - 1]:
85          max_val = list(d.values())[i - 1]
86      i -= 1
87      if i == 0:
88          goto .max_end
89      goto .max_start
90      label .max_end
91
92      # Printing most used words
93      keys = list(d.keys())
94      i = 0
95      j = max_val
96      length = len(d)
97      label .print_start
98      if n == 0:
99          goto .print_end
100     if i < length:
101         if d[keys[i]] == j:
102             print(f"{keys[i]} - {d[keys[i]]}")
103             n -= 1
104         i += 1
105         goto .print_start
106     else:
107         if j == 0:
108             goto.print_end
109         j -= 1
110         i = 0
111         goto.print_start
112     label .print_end
113
114
115 if __name__ == "__main__":
116     main()
```

Завдання 2:

```python
from goto import with_goto


@with_goto
def main():
    i = 0
    temp = 0
    row_counter = 0
    page_counter = 1  # It is more common to use line numbering that begins with 1
    d = dict()
    res = dict()
    symbols = [',', '.', '/', '!', '?', ':', ';', '-', '"', "'"]

    with open('text1.txt', 'r') as f:  # Reading the file 'text1.txt'
        text = f.read() + "$#"

    # Convert all characters to lower case
    label .to_lower
    if 64 < ord(text[i]) < 91:
        text = text[:i] + chr(ord(text[i]) + 32) + text[i+1:]
    i += 1
    if text[i] == "$":
        goto.to_lower_end
    goto .to_lower
    label .to_lower_end

    # Delete symbols from text
    i = 0
    label .drop_symbols_start
    if text[i] in symbols:
        text = text[:i] + text[i + 1:]
        i -= 1
    if text[i] == "$":
        goto.drop_symbols_end
    else:
        i += 1
        goto.drop_symbols_start
    label.drop_symbols_end
```

```
    # Fill the dictionary with values in the format:
    # {"word" : [number of words, page_1, page_2, ..., page_n ], ...}
    i = 0
    label .format_start
    if text[i] == " " or text[i] == "$" or text[i] == "\n":
        if text[i] == "\n":
            row_counter += 1
            page_counter = row_counter // 45 + 1
        if d.get(text[temp:i]):
            d[text[temp:i]][0] += 1
            if d[text[temp:i]][-1] != page_counter:
                d[text[temp:i]].append(page_counter)
        else:
            d[text[temp:i]] = [1, page_counter]
        temp = i + 1
    i += 1
    if text[i] == "#":
        goto.format_end
    goto.format_start
    label .format_end

    # Delete the words which number > 100
    i = 0
    keys = list(d.keys())
    values = list(d.values())
    length = len(d)
    label .strict_start
    if d[keys[i]][0] < 101:
        res[keys[i]] = d[keys[i]][1:]
    i += 1
    if i == length:
        goto.strict_end
    goto.strict_start
    label.strict_end
```

```
74
75      # Sort the 'keys' list
76      keys = list(res.keys())
77      values = list(res.values())
78      length = len(res)
79      i = 0
80      label .sort_start
81      if i == length - 1:
82          goto.sort_end
83      j = 0
84      # ------------------Start of inner cycle------------------
85      label .inner_start
86      if j == length - 1 - i:
87          goto.inner_end
88      if keys[j] > keys[j + 1]:
89          keys[j], keys[j + 1] = keys[j + 1], keys[j]
90      j += 1
91      goto.inner_start
92      label .inner_end
93      # ------------------End of inner cycle------------------
94      i += 1
95      goto.sort_start
96      label .sort_end
97
98      # Print the result dictionary
99      i = 0
100     label .print_start
101     print(f"{keys[i]} - {str(res[keys[i]])[1:-1]}")
102     if i == length - 1:
103         goto.print_end
104     else:
105         i += 1
106         goto.print_start
107     label .print_end
108
109
110 if __name__ == "__main__":
111     main()
```

Скріншоти роботи програмного застосунку
Завдання 1:

```
D:\Projects\PythonProject\venv\Scripts\python.exe D:\Projects\PythonProject\lab1_1.py
twain - 4
american - 3
story - 3
humorist - 2
called - 2
adventures - 2
tom - 2
sawyer - 2
huckleberry - 2
finn - 2
great - 2
 - 2
later - 2
typesetter - 2
orion - 2
angels - 2
time - 2
even - 2
earned - 2
bankruptcy - 2
financial - 2
comet - 2
known - 1
pen - 1
name - 1

Process finished with exit code 0
```

Завдання 2:

```
D:\Projects\PythonProject\venv\Scripts\python.exe D:/Projects/PythonProject/lab1_2.py
44 - 2
abbreviation - 1
able - 2, 4
about - 1, 2, 3, 4
above - 5
abroad - 3
absolute - 3
absolutely - 5
absurd - 2, 4
accent - 2, 7
accident - 5
according - 2
account - 3
accustomed - 1
actually - 3, 4
add - 2
adding - 6
addition - 1
admittedly - 1
adonis - 2
adroitly - 2
advice - 6
affairs - 4
affected - 1
affirmed - 2
afraid - 3
after - 1, 3, 5
afterwards - 2
again - 1, 3, 6
against - 3
ago - 3
agree - 2, 3
agreed - 4
ah - 2
aha - 6
```