Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені ІгоряСікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 1 з дисципліни
«Мультипарадигменне програмування»

„Проектування і аналіз алгоритмів для вирішення NP-складних задач ч.1”

Виконав(ла) *ІП-01 Семененко А. В.* _____
(шифр, прізвище, ім'я, по батькові)

Перевірив *Очеретяний О. К.* _____
(прізвище, ім'я, по батькові)

Київ 2021

1 ЗАВДАННЯ

1.1 Перше 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. Щоб все було просто, ми не піклуємося про порядок слів з однаковою частотою повторень.

1.2 Друге завдання:

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 (1800символів).

2 ВИКОНАННЯ

Для даної роботи було використано мову програмування Python. Ця мова програмування мені дуже подобається через чистоту коду і прозорість виконання, що дуже часто схоже на псевдокод.

Як відомо на Python немає goto, але я знайшов бібліотеку, що дає можливість її використовувати: «goto-statement». Завантажити цю бібліотеку дуже просто: pip install goto-statement.

2.1 Псевдокод алгоритму першого завдання:

1) Визначаємо необхідні змінні, в тому числі стоп-слова
2) Зчитуємо файл в вигляді однієї строки
3) Кожну букву в тексті перетворюємо в маленьку
4) Зчитуємо слова, не беручи до уваги стоп-слова та рахуємо повтори.
5) Сортуємо масив слів за спаданням кількості повторів
6) Виводимо n перших слів масиву в консоль

2.2 Псевдокод алгоритму другого завдання:

1) Визначаємо необхідні змінні
2) Зчитуємо файл в вигляді однієї строки
3) Кожну букву в тексті перетворюємо в маленьку
4) Зчитуємо слова рахуємо повтори з вказанням сторінок
5) Прибираємо всі слова, що зустрічаються більш ніж 100 разів
6) Сортуємо масив слів за спаданням кількості повторів
7) Виводимо масив слів в консоль

Програмна реалізація алгоритму розв’язання першого завдання (Python)

```
from goto import with_goto

@with_goto
def program():
    ignoredWords = ["i", "me", "my", "myself", "we", "our", "ours",
"ourselves",
                    "you", "your", "yours", "yourself", "yourselves",
"he", "him", "his", "himself",
                    "she", "her", "hers", "herself", "it", "its",
"itself", "they", "them", "their", "theirs",
                    "themselves",
                    "what", "which", "who", "whom", "this", "that",
"these", "those",
                    "am", "is", "are", "was", "were", "be", "been",
"being", "have", "has", "had", "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", "should", "now"]
    upper = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    lower = "abcdefghijklmnopqrstuvwxyz"

    with open("input1.txt", "r") as file:
        text = file.read()
    text += "$"

    str = ""
    i = 0
    label.edit
    if text[i] in lower:
        str += text[i]
    elif text[i] in upper:
        j = 0
        label.lowering
        if text[i] != upper[j]:
            j += 1
            goto.lowering
        str += lower[j]
    elif text[i] == "\n" or text[i] == " ":
        str += " "
    elif text[i] == "$":
        text = str + "$"
        goto.editEnd
    i += 1
    goto.edit
    label.editEnd

    word = ""
    result = [[None, 0]]
    i = 0
    label.count
    if text[i] == " ":
        if word in ignoredWords:
            word = ""
```

```
            i += 1
            goto.count
        j = 0
        label.wordInserter
        if word == result[j][0]:
            result[j][1] += 1
            word = ""
            i += 1
            goto.count
        elif result[j][0] == None:
            result[j][0] = word
            result[j][1] = 1
            result += [[None, 0]]
            word = ""
            i += 1
            goto.count
        else:
            j += 1
            goto.wordInserter
    elif text[i] == "$":
        result = result[:-1]
        wordCount = j
        goto.countEnd
    else:
        word += text[i]
    i += 1
    goto.count
    label.countEnd

    i = 0
    label.sorter
    maxId = i
    j = i + 1

    label.sorterInner
    if result[maxId][1] < result[j][1]:
        maxId = j
    if j >= wordCount:
        goto.sorterInnerEnd
    j += 1
    goto.sorterInner

    label.sorterInnerEnd
    result[i], result[maxId] = result[maxId], result[i]
    i += 1
    if i >= wordCount:
        goto.sorterEnd
    goto.sorter

    label.sorterEnd
    i = 0
    label.printer
    print(f"{result[i][0]} - {result[i][1]}")
    if i < wordCount and i < 4 - 1:
        i += 1
        goto.printer
    return

program()
```

Програмна реалізація алгоритму розв’язання другого завдання (Python)

```
from goto import with_goto

@with_goto
def program():
    upper = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    lower = "abcdefghijklmnopqrstuvwxyz"
    maxWords = 100
    wordsPerPage = 235
    lettersPerPage = 1800

    with open("input2.txt", "r", encoding="utf8") as file:
        content = file.read()
    content += "$"

    str = ""
    i = 0
    label.edit
    if content[i] in lower:
        str += content[i]
    elif content[i] in upper:
        j = 0
        label.lowering
        if content[i] != upper[j]:
            j += 1
            goto.lowering
        str += lower[j]
    elif content[i] == "\n" or content[i] == " ":
        str += " "
    elif content[i] == "$":
        content = str + "$"
        goto.editEnd
    i += 1
    goto.edit
    label.editEnd

    word = ""
    currentPage = 1
    letterCounter = 0
    wordCounter = 0
    wordsInResult = -1
    result = [[None, 0, []]]
    i = 0
    label.counter
    if content[i] == " ":
        j = 0
        wordCounter += 1
        if wordCounter >= wordsPerPage or letterCounter >= lettersPerPage:
            currentPage += 1
            letterCounter -= lettersPerPage
            wordCounter = 0
        label.wordInserter
        if word == result[j][0]:
            result[j][1] += 1
            if currentPage not in result[j][2]:
                result[j][2] += [currentPage]
            word = ""
            i += 1
            goto.counter
        elif result[j][0] == None:
            result[j][0] = word
            result[j][1] = 1
            result[j][2] += [currentPage]
            result += [[None, 0, []]]
            wordsInResult += 1
            word = ""
```

```
            i += 1
            goto.counter
        else:
            j += 1
            goto.wordInserter
    elif content[i] == "$":
        result = result[:-1]
        goto.counterEnd
    else:
        word += content[i]
    i += 1
    letterCounter += 1
    goto.counter
    label.counterEnd

    i = 0
    label.wThrower
    if result[i][1] > maxWords or result[i][0] == "":
        wordsInResult -= 1
        del result[i]
    else:
        i += 1
    if i >= wordsInResult:
        goto.wThrowerEnd
    goto.wThrower
    label.wThrowerEnd

    i = 0
    label.sorter
    maxId = i
    j = i + 1

    label.sorterInner
    if result[maxId][0] > result[j][0]:
        maxId = j
    if j >= wordsInResult:
        goto.sorterInnerEnd
    j += 1
    goto.sorterInner

    label.sorterInnerEnd
    result[i], result[maxId] = result[maxId], result[i]
    i += 1
    if i >= wordsInResult:
        goto.sorterEnd
    goto.sorter
    label.sorterEnd

    i = 0
    label.printer
    print(f"{result[i][0]} - {result[i][2]}")
    if i < wordsInResult:
        i += 1
        goto.printer
    return

program()
```

4 ВИСНОВОК

3.1 Приклад роботи програми 1:

```
C:\Anaconda3\python.exe C:\Anaconda3\task1.py
live - 2
mostly - 2
white - 1
tigers - 1
```

3.2 Приклад роботи програми 2:

```
C:\Anaconda3\python.exe C:/Anaconda3/task2.py
about - [1, 7, 9, 10, 11, 16, 20, 21]
above - [10]
absolutely - [21]
abuse - [3]
accept - [8, 9]
accidental - [18]
accomplished - [13]
account - [3]
acknowledged - [1, 19]
acquaintance - [6, 7, 22]
acquainted - [6, 10, 12, 19]
acquired - [17]
```