МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)
КАФЕДРА МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Информатика»

Тема: Основные управляющие конструкции. Wikipedia API

Студент гр. 0382 _____ Афанасьев Н. С.

Преподаватель _____ Шевская А. И.

Санкт-Петербург

2020

Цель работы.

Освоение основных управляющих конструкций языка Python и модуля Wikipedia.

Задание.

Используя вышеописанные инструменты, напишите программу, которая принимает на вход строку вида:

название_страницы_1, название страницы_2, ... название_страницы_n, сокращенная_форма_языка

и делает следующее:

1. Проверяет, есть ли такой язык в возможных языках сервиса, если нет, выводит строку *"no results"* и завершает выполнение программы. В случае, если язык есть, устанавливает его как язык запросов в текущей программе.

2. Ищет максимальное число слов в кратком содержании страниц *"название_страницы_1", "название страницы_2", ... "название_страницы_n",* выводит на экран это максимальное количество и название страницы (т.е. её title), у которой оно обнаружилось. Считается, что слова разделены пробельными символами.

Если максимальных значений несколько, выведите последнее.

3. Строит список-цепочку из страниц и выводит полученный список на экран.

Элементы списка-цепочки — это страницы *"название_страницы_1", "название страницы_2", ... "название_страницы_n"*, между которыми может быть одна промежуточная страница или не быть промежуточных страниц.

Гарантируется, что существует или одна промежуточная страница или ноль: т. е. в числе ссылок первой страницы можно обнаружить вторую.

Цепочка должна быть кратчайшей, т. е. если существуют две цепочки, одна из которых содержит промежуточную страницу, а вторая нет, стройте цепочку без промежуточного элемента.

Первая строка содержит решение подзадачи №2, вторая - №3.

Выполнение работы.

Для начала, программа получает на вход строку с данными и делит её в массив *arr* с помощью метода *split()*. Последний элемент массива, содержащий сокращённую форму языка, записывается в переменную lang.

Для выполнения первой подзадачи вызывается функция ***setLang(lang),*** которая, с помощью метода словаря *get(),* проверяет наличие этого языка в системе (*wikipedia.languages()*). Если такой язык присутствует, то он устанавливается в качестве языка запроса через метод *wikipedia.set_lang(lang),* в противном случае выводится строка *"no results"* и работа программы прекращается.

Для выполнения второй подзадачи вызывается функция ***maxSummary(arr),*** которая ищет страницу с максимальным количеством слов в кратком содержании. В теле цикла for перебираются все страницы в массиве arr и, если в текущей странице, полученной через метод *wikipedia.page(),* количество слов (*count*) в кратком содержании (*page.summary()*) наибольшее, то это число записывается в переменную *maxWords*, а её заголовок в *maxWTitle*. Программа выводит эти два значения через пробел.

Для выполнения третьей подзадачи вызывается функция ***makeChain(arr),*** которая строит список-цепочку *chain* из введённых страниц arr, между которыми может быть одна или ноль промежуточных страниц. Для каждой страницы проверяется, есть ли в её ссылках, полученных из поля *wikipedia.page().links*, ссылка на следующую страницу введённого массива arr. Если условие выполняется, то в цепочку добавляются эта страница. Если условие не выполняется, то программа просматривает каждую страницу *nextPage* из массива ссылок и ищет ту, в которой есть ссылка на следующую страницу из введённого массива arr. В этом случае в цепочку добавляется страница *nextPage*, являющаяся промежуточной, и страница из массива *arr*. Каждая страница *nextPage* получается в результате работы функции ***tryPage(title),*** которая возвращает страницу с помощью *wikipedia.page(),* либо

None, если при поиске страницы произошла ошибка. Полученная цепочка выводится на экран, после чего выполнение программы завершается.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	Айсберг, IBM, ru	115 IBM ['Айсберг', 'Буран', 'IBM']	Верно
2.	C++, JavaScript, Google Maps, en	394 Google Maps ['C++', 'C (programming language)', 'JavaScript', 'Ajax (programming)', 'Google Maps']	Верно
3.	冷蔵庫, 氷山, 中国, ja	3 中国 ['冷蔵庫', '水', '氷山', '中国']	Верно, хотя считает не кол-во слов, а абзацев (в японском нет пробелов)
4	page1, page2, page3, page4, lang	no results	Верно, нет языка с сокращением lang

Выводы.

Были изучены основные конструкции языка Python (ввод и вывод данных, списки, словари, условные операторы *if*, *elif* и *else*, циклы *for* и *while*, функции), модуль Wikipedia. Для устранения дублирования кода и повышения читаемости кода были использованы функции.

Разработана программа, принимающая на вход строку, включающую в себя названия страниц и сокращённую форму языка. Программа проверяет наличие этого языка в сервисе и устанавливает его, если он существует; ищет максимальное количество слов в кратком содержании страниц; строит список-цепочку из введённых страниц, между которыми может быть одна или ноль промежуточных страниц. Все результаты программа выводит на экран.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
import wikipedia

def setLang(lang):
    if wikipedia.languages().get(lang):
        wikipedia.set_lang(lang)
        return True
    else: return False

def maxSummary(arr):
    maxWords, maxWTitle = 0, ''
    for title in arr:
        page = wikipedia.page(title)
        count = len(page.summary.split())
        if count >= maxWords:
            maxWords, maxWTitle = count, page.title
    return str(maxWords), maxWTitle

def makeChain(arr):
    chain = [arr.pop(0)]
    for title in arr:
        page = wikipedia.page(chain[-1])
        if title in page.links: chain.append(title)
        else:
            for nextTitle in page.links:
                nextPage = tryPage(nextTitle)
                if not nextPage: continue
                if title in nextPage.links:
                    chain.extend([nextTitle, title])
                    break
    return chain

def tryPage(title):
    try:
        return wikipedia.page(title)
    except Exception:
        return None

arr = input().split(', ')
lang = arr.pop(-1)
if setLang(lang):
    print(' '.join(maxSummary(arr)))
    print(makeChain(arr))
else: print("no results")
```