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

Студентка гр. 0382 _____ Здобнова К.Д.

Преподаватель _____ Шевская Н.В.

Санкт-Петербург

2020

Цель работы.

Научиться работать с основными управляющими конструкциями Python, познакомиться с модулем *wikipedia* и научиться работать с wiki-страничками сервиса 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.

Предположим, нам на вход поступила строка:

Айсберг, IBM, ru

В числе ссылок страницы с названием "Айсберг", есть страница с названием , которая содержит ссылку на страницу с названием "Буран", у которой есть ссылка на страницу с названием "IBM" -- это и есть цепочка с промежуточным звеном в виде страницы "Буран".

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.

Пример входных данных:

Айсберг, IBM, ru

Пример вывода:

115 IBM

['Айсберг', 'Буран', 'IBM']

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

Основные теоретические положения.

Используемые функции модуля Wikipedia в лабораторной работе:

page(title) – ищет страницу с названием title;

wikipedia.languages() – словарь, ключами которого являются скоращения

set_lang(lang) – устанавливает язык lang, как язык запросов в текущей программе;

page.summary – строка краткого содержания страницы page;

page.title – название страницы page;

page.links – список названий страниц, ссылки на которые содержит страница page.

Выполнение работы.

В лабораторной работе использовалась функция *is_page_valid(page_name)*, которая возвращает *True*, если wiki-страница с таким названием существует и *False* в ином случае.

Вводные данные записываются в список *input_list*.

Значение языка запросов хранится в последнем элементе списка, если такого языка не существует в словаре wikipedia, то программа печатает *“no results”* и завершает исполнение. Если такой язык существует, то обрабатываются подзадачи №2 и №3.

Функция *max_words_page(input_list)* обрабатывает входной список и находит вики-страницу с максимальным числом слов в кратком содержании, возвращает такую страницу.

В переменной *max_words* хранится максимальное число слов в кратком содержании страниц "название_страницы_1", "название страницы_2", ... "название_страницы_n". В переменной *position* хранится индекс страницы в словаре, с максимальным числом слов в кратком содержании.

Функция *make_chain(input_list)* выполняет 3 подзадачу – составляет минимально возможную цепочку страниц, возвращает список.

В переменную *length* записывается размер списка *input_list* без сокращенной формы языка.

Переменная *processed* используется в цикле обработки страниц для составления кратчайших цепочек. В переменную *temp* записываются промежуточные страницы wikipedia. Результат для 3 подзадачи записывается в список *input_list*.

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']	Страница «Айсберг» не содержит ссылки на страницу «IBM», кратчайший путь через страницу «Буран»
2.	Россия, Япония, Вена, ru	341 Россия ['Россия', 'Япония', '11 февраля', 'Вена']	Страница «Япония» не содержит прямой ссылки на страницу «Вена», кратчайший путь через страницу «11 февраля»
3.	Cats, Cats in ancient Egypt, New Kingdom of Egypt, Pharaoh, en	344 Cat ['Cats', 'Cats in ancient Egypt', 'New Kingdom of Egypt', 'Pharaoh']	Ссылка каждой последующей страницы содержится в предыдущей странице
4	Cats, Composer, Cantata	no results	Язык «Cantata» не существует в словаре Wikipedia

Выводы.

Были изучены основные управляющие конструкции языка Python: условные операторы, операторы ветвления. Был исследован модуль *wikipedia* для работы с wiki-страничками сервиса Wikipedia.

Разработана программа, выполняющая считывание с клавиатуры исходных данных с помощью функции *input()*. Для обработки входных данных и выполнения подзадач были использованы функции модуля wikipedia, такие как: *page(title)*, *wikipedia.languages()*, *set_lang(lang)*; и поля класса: *page.summary*, *page.title* и *page.links*.

Также использовались условные операторы *if-else* и цикл *for*. Для отлавливания исключительных ситуаций был использован блок *try-except*.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb1.py

```
import wikipedia

def is_page_valid(page):
    try:
        wikipedia.page(page)
    except Exception:
        return False
    return True

def max_words_page(input_list):
    max_words = 0
    position = 0
    for i in range (len(input_list) - 1):
        if is_page_valid(input_list[i]):
            if len(wikipedia.page(input_list[i]).summary.split()) >
max_words:
                max_words                                          =
len(wikipedia.page(input_list[i]).summary.split())
                position = i
    return wikipedia.page(input_list[position]).title

def make_chain(input_list):
    input_list.pop(len(input_list) - 1)
    length = len(input_list) - 1
    processed = 0
    for pos in range (length):
        if     input_list[processed     +     1]     not     in
wikipedia.page(input_list[processed]).links:
            for                          j                          in
range(len(wikipedia.page(input_list[processed]).links)):
                temp                                               =
wikipedia.page(wikipedia.page(input_list[processed]).links[j])
                if input_list[processed + 1] in temp.links:
                    input_list.insert(processed        +        1,
wikipedia.page(input_list[processed]).links[j])
                    processed += 1
                    break
        processed += 1
    return input_list

input_list = input().split(', ')
if     input_list[len(input_list)     -     1]     not     in
wikipedia.languages().keys():
    print('no results')
else:
    wikipedia.set_lang(input_list[len(input_list) - 1])
    max_page = max_words_page(input_list)
    print(len(wikipedia.page(max_page).summary.split()), max_page)
    input_list = make_chain(input_list)
    print(input_list)
```