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. WikipediaAPI

Студент гр. 0382 _____ Санников В.А.

Преподаватель _____ Шевская Н.В.

Санкт-Петербург

2020

Цель работы.

Изучить основные управляющие конструкции языка Python и WikipediaAPI.

Задание.

Написать 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. Bno results"
случае, если язык есть, устанавливает его как язык запросов в текущей программе.

2. Ищет максимальное число слов в кратком содержании страниц и 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, вывести последнее.

3. Строит список-цепочку из страниц и выводит полученный список на экран. Элементы списка-цепочки — это страницы из входных данных, между которыми может быть одна промежуточная страница или не быть промежуточных страниц.

Основные теоретические положения.

Использованные конструкции Python:

◦print() — выводит принимаемые значения на консоль;

◦input() — считывает входные данные, возвращает строку;

◦len() — принимает строку или список, возвращает целочисленное значение – длину входного объекта;

◦range() — генерирует ряд чисел в заданном диапазоне с определённым шагом;

•Функции модуля Wikipedia:

◦page(title) – возвращает объект класса WikipediaPage, который представляет собой страничку сервиса Wikipedia, название которой -строка title;

◦languages() –возвращает словарь, ключами которого являются сокращенные названия языков сервиса, а значениями – полные названия;

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

•Операторы:

◦if: else:— если значение выражения после оператора if и перед двоеточием - true, выполняет блок кода с одинаковым уровнем отступа после if, если false – блок кода после else;

◦in— если объект перед оператором является подстрокой или элементом объекта после оператора – значение выражения – true, в противном случае – false;

◦notin— работает аналогично оператору in, но инвертирует значение;

◦break —прерывает выполнение цикла;

◦return — используется в функциях для возвращения каких-либо значений.

•Циклы:

◦for <переменная> in <итерируемый объект>:— для каждого значенияперемен переменной, находящегося в итерируемом объекте,выполняет блок кода с одинаковым уровнем отступа после двоеточия;

•Пользовательские функции:

◦def <переменная> in <итерируемый объект>:— для каждого значенияназвание тексте программы по названию функции выполняется блок кода, находящийся после двоеточия в определении функции, используя принимаемые параметры.

•Методы:

◦str.split()— метод класса str, принимает на вход разделитель - один или несколько символов (по умолчанию – пробел), разбивает строку, к которой применён, на подстроки по разделителю и возвращает список этих подстрок;

◦list.append() — добавляет в конец списка list элемент из круглых скобок.

•Обращения к полям:

◦page.summary— поле класса page модуля Wikipedia, возвращает многострочный литерал – краткое содержание страницы page;

◦page.title — поле класса page модуля Wikipedia, возвращает строку – название страницы page;

◦pagei.links—поле класса page модуля Wikipedia, возвращает список строк – названий страниц, ссылки на которые содержит страница page.

•Инструкция: ◦import <переменная> in <итерируемый объект>:— для каждого

Выполнение работы.

Ход работы:

В самой первой строке программы импортируем модуль wikipedia с помощью инструкции import.

Для считывания данных с клавиатуры используем переменную a, в которую с помощью функции input() и метода .split(‘, ‘).

Итак, начнем выполнение подзадач:

1) Данная подзадача выполняется с помощью функции lang_here(lg), которая на вход принимает строку и проверяет, есть ли данный ключ в

словаре, который возвращает метод wikipedia.languages(). Тут же устанавливается язык lg как язык запросов с помощью функции wikipedia.set_lang(lg) и возвращает True, если такой имеется, False, если отсутствует.

2) Вторую подзадачу у нас реализует функция max_st(st), получающая на вход список строк, являющимися названиями страниц. В начале функции объявляем несколько переменных: max_str = ‘ ‘ - название страницы с максимальным кратким содержанием, size = 0 — размер страницы, которому будет присваиваться соответствующее значение с некоторой итерации цикла for и numb_max = 0 — максимальный размер краткого содержания.

В цикле for, i поочерёдно принимает каждое значение из списка st. Заранее присваиваем переменной i_page значение страницы с Wikipedia через wikipedia.page(i) (для удобства). Далее присваиваем в переменную size длину краткого содержания страницы(i_size) с помощью len() и метода .split(), и если оно не меньше numb_max, то в numb_max кладем переменную size, а max_str принимает значение i-го. Функция возвращает список из двух элементов : numb_max и max_str.

3) Третья подзадача выполняется с помощью функции st_cep(st), которая на вход получает список строк, которые являются названиями страниц. В самом начале функции создаем список ans из одного элемента st[0] и присваиваем переменной count значение длинны списка st уменьшенную на единицу (в силу особенности перебора значений в цикле for).

В цикле for для каждой страницы, кроме последней создаём список ссылок этой страницы thefirst, и с помощью оператора in проверяем, содержится ли ссылка на следующую по списку страницу st[i+1]. Если есть, то к списку ans добавляется st[i+1]. Если же нет, то для каждой ссылки из thefirst проверяется, есть ли такая страница в wikipedia, и если да, то для этой страницы формируем список ссылок thesecond, и проверяется, входит ли в

этот список st[i+1]. Если входит, то к списку ans добавляется st[i+1] и цикл прерывается с помощью break.

Функция возвращает список res.

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']	Программа работает верно
2.	Айсберг,IBM, apomogite	No results	Программа работает верно

Выводы.

Была написана программа, которая считывает данные с помощью функции input() и метода .split() и выводит результат с помощью функции print().

Первая подзадача была решена с помощью функции lang_here(lg).

Вторая подзадача была решена с помощью функции max_st(st), в которой с помощью цикла for и проверки условия if находится название страницы с самым длинным описанием.

Третья подзадача была решена с помощью функции st_cep(st), которая с помощью циклов for и проверки условий if...else проверяла, есть ли на одной странице ссылки на следующую (или есть, но через промежуточную страницу) и создавала список-цепочку этих страниц с помощью метода .append().

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: papa.py

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

def lang_here(lg):
    if lg in wikipedia.languages():
        wikipedia.set_lang(lg)
        return True
    else:
        return False

def max_st(st):
    max_str = ''
    size = 0
    numb_max = 0
    for i in st:
        i_page = wikipedia.page(i)
        size = len((i_page.summary).split())
        if size >= numb_max:
            numb_max = size
            max_str = i_page.title
    return [numb_max, max_str]

def st_cep(st):
    ans = [st[0]]
    count = len(st)-1
    for i in range(count):
        thefirst = wikipedia.page(st[i]).links
        if st[i+1] in thefirst:
            ans.append(st[i+1])
        else:
            for j in thefirst:
                if is_page_valid(j):
                    thesecond=(wikipedia.page(j)).links
                    if st[i+1] in thesecond:
                        ans.append(j)
                        ans.append(st[i+1])
                        break
    return ans

a = input().split(', ')
if lang_here(a[len(a)-1]):
    result = max_st(a)
    print(result[0], result[1])
    print(st_cep(a))
else:
    print('no results')
```

```
    print(st_cep(a))
else:
    print('no results')
```