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 _____ Сергеев Д.А.

Преподаватель _____ Шевская Н.В.

Санкт-Петербург

2020

Цель работы.

Изучить основные управляющие конструкции языка Python и научиться работать с модулем Wikipedia API.

Задание.

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.

Основные теоретические положения.

В данной работе были использованы такие конструкции языка Python как:

- Встроенные функции:
  - input() – считывает входные данные, возвращает строку
  - print() – выводит аргумент на консоль
  - len() – принимает на вход строку или список, возвращает целочисленную длину данного объекта

    - range() – создаёт список , содержащий арифметическую прогрессию с определенным шагом
- Функции модуля Wikipedia API:
    - languages() – возвращает словарь, ключами которого являются сокращенные названия языков, а значениями – названия
    - page(title) – возвращает объект класса WikipediaPage, который представляет собой страничку сервиса Wikipedia, название которой - строка title
    - set_lang(lang) - устанавливает язык lang, как язык запросов в текущей программе
- Операторы:
    - if: else: - если значение выражения после оператора if - true, то выполняется блок кода в одинаковой табуляцией после if, в случае false выполняется блок кода после else:
    - not – инвертирует значение выражения
    - in – если объект перед оператором является подстрокой или элементом объекта после оператора, то значение выражения – true, в противном случае – false
    - break – прерывает цикл
    - continue – цикл переходит на следующую итерацию
    - return – в функции возвращает значение
- Пользовательские функции:
    - def is_page_valid(page) - возвращает True, если wiki-страница с таким названием существует и False в ином случае.
- Методы:
    - str.split(sep) – метод класса str, принимает на вход разделитель(по умолчанию – пробел) и разбивает строку, к которой применён, на подстроки по разделителю и возвращает список этих подстрок.
    - list.append() – добавляет аргумент в конец списка list

- Обращения к полям:
    - page.summary – поле класса page модуля Wikipedia, которое возвращает строку, содержащую краткое содержание страницы page
    - page.title – поле класса page модуля Wikipedia, которое возвращает строку, содержащую краткое содержание страницы page
    - page.links – поле класса page модуля Wikipedia, которое возвращает список названий страниц, ссылка на которые содержит страница page

Выполнение работы.

Разработанный программный код см. в приложении А.

В самом начале программы с помощью инструкции *import wikipedia* подключается модуль *wikipedia*.

Далее в переменную *a* с помощью функции *input()* записывается строка, введённая пользователем. После этого с помощью строкового метода *split(‘, ‘)* в переменную *a* помещается список, состоящий из подстрок входящих в введённую строку.

Подзадачи:

1). В оператор *if* поступает значение функции *(set_language(a[-1]))*. Функция *set_language(lang)* получает на вход строку *a[-1]*. И, если строка *a[-1]* является ключом для словаря *wikipedia.languages()*, то язык *a[-1]* устанавливается, как язык запросов, а функция возвращает значение *True*, в противном случае возвращается значение *False*. Если вернулось значение *False*, то на терминал с помощью функции *print()* печатается строка: *“no results”* и программа завершается. А если вернулось значение *True*, то программа переходит к блоку кода, записанному ниже оператора *if*.

2). На терминал с помощью функции *print()* выводится сначала первый, а потом нулевой элементы кортежа, возвращаемого функцией *max_word_title(list)*, в которую поступает список *a*. В функции *max_word_title(list)* создаётся локальная переменная *max=0*, созданная для хранения максимального числа слов в кратком содержании, и пустая строка *title_max*, в которую будет записан

заголовок страницы, чьё краткое содержание содержит наибольшее количество слов. Далее с помощью цикла *for i in range(len(list)-1)* удаётся пройтись по всем нужным элементам списка *a*. И в случае, если *len(wikipedia.page(a[i]).summary.split())>max*, то переменной max присваивается значение len(wikipedia.page(a[i]).summary.split()), а строке *title_max* - *wikipedia.page(a[i]).title*. В логическом выражении сначала, создаётся объект класса *WikipediaPage* с помощью функции *wikipedia.page()*, принимающей элемент списка *a* на вход, далее применяется поле класса *WikipediaPage summary*, которое возвращает строку, содержащую краткое содержание страницы на *Wikipedia*, после этого применяется строковый метод *split()*, который создаёт список подстрок строки с кратким содержанием, с разделителем равным пробелу. И наконец, значение переменной max сравнивается со значением функции *len()*, примененной к полученному списку. Функция возвращает переменные *title_max* и *max*.

3). На терминал с помощью функции *print()* выводится список *k*, возвращаемый функцией *shortest_chain(list)*, на вход которой поступает список a. В функции *shortest_chain(list)* создаётся пустой лист *k*, и переменную *key=0*, после этого с помощью метода *append()* в лист k добавляется нулевой элемент списка *list*. С помощью цикла *for i in range(len(list) - 1)* удаётся пройтись по всем интересующим нас элементам списка *list*, в переменную *l1* помещается объект класса *WikipediaPage* с помощью функции *wikipedia.page(list[i])*. А значение переменной *key* присваивается ноль. Далее если *i* не равен *len(list)-1*, так как к тому времени когда программа дойдет до этого индекса, *list[len(list)-1]* уже будет добавлен в список *k*, проверяется, если следующий элемент*(i+1)* списка *list* входит в список *l1.links*(список ссылок, находящихся на странице *list[i]*), то в список *k* добавляется элемент списка *list* с индексом *i+1*. Если же следующий элемент*(i+1)* списка *list* не входит в список *l1.links*, то мы ищем промежуточную страницу. С помощью цикла *for j in l1.links*, мы проходим по каждой ссылке к странице *l1*. Возможна такая ситуация, при которой две или более промежуточные страницы, содержат ссылку на страницу из списка *list*, чтобы

добавилась только одна промежуточная страница, мы контролируем это с помощью переменной *key*, которая принимает значение 1, если промежуточная страница включена в цепочку и 0, если не включена. Поэтому вначале цикла тела цикла *for* проверяется чему равна переменная *k*, если она равна 1, то с помощью оператора *break* цикл завершается, если же *k=0*, то с помощью пользовательской функции *is_page_valid(page)* проверяется, доступна ли страница если нет, то итерация завершается, а если да то, после этого создаётся переменная *l2=wikipedia.page(j)* и если *list[i+1]* входит в список *l2.links*(список ссылок страницы, на которую ссылается страница *list[i+1]*), то как промежуточная страница *j*, так и страница из списка *list* добавляются в список *k*, а переменной *key* присваивается значение 1. Функция возвращает список *k*.

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 правильно
2.	Айсберг, IBM, privet	no results	Программа работает правильно

Выводы.

В ходе выполнения работы были изучены основные управляющие конструкции языка Python и получен опыт работы с модулем Wikipedia API.

Разработана программа, выполняющая считывание с клавиатуры исходных данных с помощью функции *input()* и метода *split()*.

Первая подзадача выполнена с помощью функции *set_language()* и логических операторов *if: else:* и *in*, и функции *wikipedia.set_lang()*.

Вторая подзадача была выполнена с помощью функции *max_word_title()* и цикла *for* с логическим оператором *if: else:* . Результат подзадачи выводится с помощью встроенной функции *print()*.

Третья подзадача была выполнена с помощью функции *shortest_chain()* и цикла for с логическими операторами *if*: *else*:, *in*, *break*, *continue*. Результат выводится функцией *print()*.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: cs_lab1.py

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

def set_language(lang):
    if lang in wikipedia.languages():
        wikipedia.set_lang(a[len(a)-1])
        return True
    else:
        return False

def max_word_title(list):
    max=0
    title_max=''
    for i in range (len(list)-1):
        if len(wikipedia.page(a[i]).summary.split())>max:
            max = len(wikipedia.page(a[i]).summary.split())
            title_max=wikipedia.page(a[i]).title
    return title_max,max

def shortest_chain(list):
    k = []
    key = 0
    k.append(list[0])
    for i in range(len(list) - 1):
        l1 = wikipedia.page(list[i])
        key = 0
        if i != len(list) - 1:
            if (list[i + 1] in l1.links):
                k.append(list[i + 1])
            else:
                for j in l1.links:
                    if (key == 1):
                        break
                    else:
                        if is_page_valid(j) == True:
                            l2 = wikipedia.page(j)
                            if list[i + 1] in l2.links:
                                k.append(j)
                                k.append(list[i + 1])
                                key = 1
                                continue
                        else:
                            break
    return k

a=input().split(', ')
if (set_language(a[-1])):
    print (max_word_title(a)[1],max_word_title(a)[0])
    print(shortest_chain(a))
else:
print('no results')
```