МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Математического Обеспечения и Применения ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «информатика»

Тема: Основные управляющие конструкции. Wikipedia API

Студент гр. 0382 _____ Тюленев Т.В.

Преподаватель _____ Шевская Н.В.

Санкт-Петербург

2020

Цель работы.

Изучение базовых конструкций языка Python и модуля Wikipedia.

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. В 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.

В данной работе были использованы такие конструкции языка Python как:

- Встроенные функции:
  - *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*;
  - *not in* — 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 значения переменной, находящегося в итерируемом объекте, выполняет блок кода с одинаковым уровнем отступа после двоеточия;
- Пользовательские функции:
  - *def <название функции>(<принимаемые параметры>):* — при вызове в 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*.

Выполнение работы.

В самом начале программы необходимо импортировать модуль wikipedia строкой *import wikipedia.*

Для решения поставленных задач необходимо сначала считать входные данные. Для этого используется переменная *S,* в которую при помощи функции *input()* и метода *split(', ')* записывает список, состоящий из подстрок входной строки, разделённой по запятой с пробелом.

1. Выполнение первой подзадачи.

Производится при помощи пользовательской функции *is_lang_valid(lang),* принимающей *lang* в качестве параметра *lang,* в которой с помощью операторов *if* и *in* проверяем входит ли сокращённое название языка, записанное в *lang* в список ключей словаря языков сервиса, полученного с помощью функции *wikipedia.languages().* Если не входит – возвращаем значение False.

Если же введённый язык является одним из языков сервиса – с помощью функции *wikipedia.set_lang(lang)* он устанавливается в качестве языка запросов в текущей программе и возвращается значение True.

2. Выполнение второй подзадачи.

Далее для реализации второй подзадачи используется пользовательская функция *max_sum(),* принимающая список введённых названий страниц, в которой существуют переменные:

- *summ* – предназначена для хранения целого числа – количества слов в самом длинном кратком содержании страницы;
- *title* – предназначена для хранения сроки – названия страницы с максимальный количеством слов в кратком содержании.

В цикле *for*, количество итераций которого равно количеству введённых названий страниц, в каждой итерации оператором *if* с помощью функции *len* проверяется количество слов в кратком содержании очередной страницы, если оно больше текущего значения переменной *summ*, происходит запись этого количества в переменную *summ*. Функция возвращает кортеж из двух элементов: *summ, title.*

3. Выполнение третьей подзадачи.

Для решения этой подзадачи реализована пользовательская функция list_chain*(),* принимающая список названий введённых страниц, в которой создаётся список *way*, нулевым элементом которого является название первой страницы.

С помощью цикла *for* для каждого названия страницы кроме последнего создаётся переменная *links*, хранящая список ссылок этой страницы. Далее с помощью оператора *in* проверяется название следующей страницы на вхождения в список *links,* если название входит в список, значит между страницами нет промежуточной, поэтому в конец списка *way* добавляется название следующей страницы.

Если же название следующей страницы не входит в список *links,* то с помощью цикла *for* для каждой страницы из *links* пользовательской функцией *is_page_valid()* проверяется факт существования этой страницы, если страница существует, то создаётся список *sub_links* ссылок этой страницы и, если следующая страница списка *ss* входит в список *sub_links,* значит текущая страница из списка *links* является промежуточной, поэтому в конец списка *way* сначала добавляется она, а потом — следующая страница из списка *ss,* после чего производится выход из цикла оператором *break.* .

Таким образом создаётся список страниц, в который если требуется, включены промежуточные страницы. Этот список и возвращает функция *list_chain()*. Результат выводится функцией *print(make_chain(inp))*.

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, 11	no results	Программа работает правильно

Выводы.

В ходе работы были изучены основные управляющие конструкции языка Python и модуль wikipedia.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

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

def is_lang_valid(lang):
     if lang in wikipedia.languages():
          wikipedia.set_lang(lang)
          return True
     else:
          return False

def max_sum(pages):
    summ = 0
    title = ''
    for i in range(len(pages)):
        if len(wikipedia.page(pages[i]).summary.split()) >= summ:
            summ = len(wikipedia.page(pages[i]).summary.split())
            title = wikipedia.page(pages[i]).title
    return summ, title

def list_chain(ss):
    way = [ss[0]]
    for i in range(len(ss) - 1):
        links = wikipedia.page(ss[i]).links
        if ss[i + 1] in links:
            way.append((ss[i + 1]))
        else:
            for j in range(len(links)):
                if is_page_valid(links[j]):
                    sub_links = wikipedia.page(links[j]).links
                    if ss[i + 1] in sub_links:
                        way.append(links[j])
                        way.append(ss[i + 1])
                        break
     return way

s = list(input().split(','))
s = [i.strip() for i in s]
lang = s[len(s)-1]
s.pop()

if is_lang_valid(lang):
     x = max_sum(s)
     print (x[0], x[1], sep = ' ')
     print (list_chain(s))
else:
     print ("no results")
```