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

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Введение в информационные технологии»

Тема: Основные управляющие конструкции языка Python

Студентка гр. 9304 _____ Каменская Е.К.

Преподаватель _____ Размочаева Н.В.

Санкт-Петербург

2019

Цель работы.

Изучить работу основных управляющих конструкций языка Python и использование подключаемых модулей и написать свою программу, используя их.

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.

Ваша программа должна располагаться в main.py.

Основные теоретические положения.

Модуль wikipedia:

- Функция page(title) — осуществляет поиск страницы и возвращает объект класса WikipediaPage, который представляет собой страничку сервиса Wikipedia, название которой - строка title.
- Функция languages() — осуществляет поиск всех возможных языков сервиса и возвращает словарь, ключами которого являются сокращенные названия языков, а значениями - названия.
- Функция set_lang(lang) — устанавливает язык lang, как язык запросов в текущей программе, возвращаемое значение отсутствует.

Атрибуты класса WikipediaPage (страницы сервиса Wikipedia):

- page.summary – краткое содержание страницы page;
- page.title – название страницы page;
- page.links – список названий страниц, ссылки на которые содержит страница page.

Выполнение работы.

Строка, введенная пользователем, разделяется при помощи функции *split()*. Полученный список хранится в переменной *user_in.*

Функции:

- *is_page_valid(page)* – получает на вход название страницы и проверяет, существует ли такая страница, путем попытки ее поиска функцией *page()*. Если возникает ошибка (страница не существует), функция возвращает *False*, иначе – *True*.
- *check_language(lang)* – получает на вход строку с сокращенным названием языка и проверяет с помощью оператора *if* и функции *languages()*, есть ли такой язык в языках сервиса Wikipedia. Если да, устанавливает язык *lang* как язык запросов в текущей программе (*wikipedia.set_lang(lang)*) и возвращает *True*, в противном случае возвращает *False.*
- *max_summary(pages)* – получает на вход массив имен страниц *pages*, итерируется по нему с помощью цикла *for*. На каждой итерации создает объект класса WikipediaPage *tab* с названием из входного списка и получает длину *tab_sum* краткого содержания найденной страницы функцией *len*.

  С помощью условного оператора *if* сравнивает новую *tab_sum* с переменной *max_sum*, перед циклом инициализированной нулем. Если *tab_sum* оказывается больше или равна *max_sum*, значение *tab_sum* присваивается переменной *max_sum*, а переменной *max_tab*, инициализированной пустой строкой перед циклом, присваивается строка с названием (*title*) *tab*.

  На выход подается строка, состоящая из *max_sum*, сконвертированной в строку с помощью функции *str()*, и *max_tab*.
- *make_chain(pages)* – получает на вход массив имен страниц *pages*, строит список-цепочку *chain* из этих страниц и, если нужно, промежуточных звеньев и возвращает его.

  Проходя входной массив циклом *for*, функция создает объект WikipediaPage *tab* и с помощью оператора *if* проверяет, есть ли в списке ссылок этого объекта (*tab.links*) следующее по индексу в

массиве имя страницы. Если да, имя следующей страницы добавляется в выходной список *chain*, иначе начинается цикл *for* для элементов в списке ссылок *tab*.

Каждый элемент проверяется функцией *is_page_valid()* и, если страница с таким именем существует, создается объект *elem*. Дальнейший алгоритм аналогичен алгоритму действий с объектом *tab*. Если такой страницы не найдено, цикл переходит к следующей итерации с помощью команды *continue*.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	Айсберг, IBM, ru	115 IBM ['Айсберг', 'Буран (космический корабль)', 'IBM']
2.	Чуумпу далай, Хоту Америка, Континент, sah	77 Чуумпу далай ['Чуумпу далай', 'Америкалар', 'Хоту Америка', 'Континент']

Выводы.

Были изучены и применены на практике основные управляющие конструкции языка Python, реализованы собственные функции, использован подключаемый модуль wikipedia. Разработана программа, считывающая с клавиатуры строку, обрабатывающая ее в соответствии с заданием и печатающая две строки, содержащие ответ. Для проверки существования страниц использован блок *try-except*, для взаимодействия с сервисом Wikipedia были использованы функции *page(), languages()* и *set_lang()* из импортированного модуля. Для получения данных о конкретном объекте класса WikipediaPage задействованы атрибуты *summary*, *title* и *links*.

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 check_language(lang):
    if lang in wikipedia.languages().keys():
        wikipedia.set_lang(lang)
        return True
    else:
        return False

def max_summary(pages):
    max_sum = 0
    max_tab = ''
    for name in pages:
        tab = wikipedia.page(name)
        tab_sum = len(tab.summary.split())
        if tab_sum >= max_sum:
            max_sum = tab_sum
            max_tab = tab.title
    return str(max_sum) + ' ' + max_tab

def make_chain(pages):
    chain = [pages[0]]
    for i in range(len(pages)-1):
        tab = wikipedia.page(pages[i])
     if pages[i+1] in tab.links:
            chain.append(pages[i+1])
        else:
            for name in tab.links:
                if not is_page_valid(name):
                    continue
                elem = wikipedia.page(name)
                if pages[i+1] in elem.links:
                    chain.append(name)
                    chain.append(pages[i+1])
                    break
    return chain
```

```
user_in = input().split(', ')
if check_language(user_in[-1]):
    print(max_summary(user_in[0:-1]))
    print(make_chain(user_in[0:-1]))
else: print('no results')
```