

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

Кондратов Ю.А.

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*;
 - *page.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 *inp*, 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 *set_lang(lg)*, принимающей *inp[-1]* в качестве параметра *lg*, в которой с помощью операторов *if* и *not in* проверяем входит ли сокращённое название языка, записанное в *lg* в список ключей словаря языков сервиса, полученного с помощью функции *wikipedia.languages()*. Если не входит – выводим строку “*no results*” и возвращаем значение *False*.

Если же введённый язык является одним из языков сервиса – с помощью функции `wikipedia.set_lang(inp[-1])` он устанавливается в качестве языка запросов в текущей программе и возвращается значение `True`.

Если значение `set_lang()` - `False`, то функцией `print()` выводится строка «*no results*».

Если значение `set_lang()` - `True`, то выполнение программы продолжается и из списка входных данных при помощи метода `inp.pop(-1)` удаляется элемент с названием языка.

2. Выполнение второй подзадачи.

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

- `max_words` – предназначена для хранения целого числа – количества слов в самом длинном кратком содержании страницы;
- `longest_page` – предназначена для хранения строки – названия страницы с максимальным 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 `max_words`, происходит запись этого количества в переменную `max_words`. Функция возвращает кортеж из двух элементов: `max_words`, `longest_page`. Таким образом выполняется вторая подзадача программы и функцией `print(max_summary[0], max_summary[1])` выводится результат.

3. Выполнение третьей подзадачи.

Для решения этой подзадачи реализована пользовательская функция `make_chain()`, принимающая список названий введённых страниц, в которой создаётся список `ans`, 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 *ans* 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 *inp* входит в список *sub_links*, значит текущая страница из списка *links* является промежуточной, поэтому в конец списка *ans* сначала добавляется она, а потом — следующая страница из списка *inp*, 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 *make_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, гу	115 IBM ['Айсберг', 'Буран', 'IBM']	Программа работает правильно
2.	Айсберг, IBM, hh	no results	Программа работает правильно

Выводы.

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

Разработана программа, считывающая с помощью функции *input()* и метода *split()* входные данные.

Первая подзадача программы реализована с помощью функции *set_lang()*.

Вторая подзадача реализована в функции *max_summary()* с помощью алгоритма поиска максимума в цикле *for*, результат подзадачи выводится встроенной функцией *print()*.

Третья подзадача реализована функцией *make_chain()*, в ней при помощи списка *ans* и двух циклов *for*, в которых производится проверка вхождения названия следующей страницы в список *links* ссылок текущей или один из списков *sub_links* ссылок страниц из списка *links*, и добавления следующей и, если требуется, промежуточной страниц в список *ans*. Вывод данных производится функцией *print()*.

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

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

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

inp = input().split(' ', ')')
if not set_lang(inp[-1]):
    print("no results")
else:
    inp.pop(-1)
    print(max_summary(inp)[0], max_summary(inp)[1])
    print(make_chain(inp))
```