Федеральное государственное автономное образовательное

учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Направление подготовки: 09.03.04 – Системное и прикладное программное обеспечение

Дисциплина «Информатика»

Отчёт по лабораторной работе №3

Регулярные выражения

Вариант №408417

Выполнил: студентка группы P3115

Галак Екатерина Анатольевна

Проверил

Белокон Юлия Алексеевна

Санкт – Петербург, 2024

Оглавление

Задание:

Задание №1 (Задание на 60 баллов (Смайлики))

1) Реализуйте программный продукт на языке Python, используя регулярные выражения по варианту, представленному в таблице.

2) Для своей программы придумайте минимум 5 тестов. Каждый тест является отдельной сущностью, передаваемой регулярному выражению для обработки. Для каждого теста необходимо самостоятельно (без использования регулярных выражений) найти правильный ответ. После чего сравнить ответ, выданный программой, и полученный самостоятельно. Все 5 тестов необходимо показать при защите.

3) Программа должна считать число смайликов определённого вида (вид смайлика описан в таблице вариантов) в предложенном тексте. Все смайлики имеют такую структуру: [глаза][нос][рот]. Вариантом является различные наборы глаз, носов и ртов.

408417 % 6 = 3

408417 % 4 = 1

408417 % 8 = 1

Номер в ИСУ % 6	Глаза	Номер в ИСУ % 4	Нос	Номер в ИСУ % 8	Рот
3	:	1	<	1	)

Формат смайлика: :<)

Задание №2 (Необязательное задания для получения оценки «4» или «5» (позволяет набрать +18 баллов от максимального числа баллов БаРС за данную лабораторную))

1) Реализуйте программный продукт на языке Python, используя регулярные выражения по варианту, представленному в таблице.

2) Для своей программы придумайте минимум 5 тестов. Каждый тест является отдельной сущностью, передаваемой регулярному выражению для обработки. Для каждого теста необходимо самостоятельно (без использования регулярных выражений) найти правильный ответ. После чего сравнить ответ, выданный программой, и полученный самостоятельно. Все 5 тестов необходимо показать при защите. Пример тестов приведён в таблице.

3) Можно использовать циклы и условия, но основной частью решения должны быть регулярные выражения.

408417 % 6 = 3

<table>
<tr><th>Номер в ИСУ % 6</th><th>Задание</th></tr>
<tr><td>3</td><td>Дан текст. Требуется найти в тексте все фамилии, отсортировав их по алфавиту. Фамилией для простоты будем считать слово с заглавной буквой, после которого идут инициалы. Могут существовать двойные фамилии, которые тоже нужно учитывать.
<table>
<tr><th>Ввод</th><th>Вывод</th></tr>
<tr><td>Студент Вася вспомнил, что на своей лекции Балакшин П.В. упоминал про старшекурсников, которые будут ему помогать: Анищенко А.А., Машина Е.А. и Голованова-Иванова Д.В.</td><td>Анищенко
Балакшин
Голованова-Иванова
Машина</td></tr>
</table></td></tr>
</table>

Задание №3 (Необязательное задания для получения оценки «4» или «5» (позволяет набрать +22 балла от максимального числа баллов БаРС за данную лабораторную))

1) Реализуйте программный продукт на языке Python, используя регулярные выражения по варианту, представленному в таблице.

2) Для своей программы придумайте минимум 5 тестов. Все 5 тестов необходимо показать при защите.

3) Протестируйте свою программу на этих тестах.

4) Можно использовать циклы и условия, но основной частью решения должны быть регулярные выражения.

408417 % 8 = 1

Номер в ИСУ % 8	Задание
1	С помощью регулярного выражения найти в тексте слова, в которых встречается строго одна гласная буква (встречаться

<table>
<tr>
<td></td>
<td>она может несколько раз). Пример таких слов: окно, трава, молоко, etc. После чего данные слова требуется отсортировать сначала по увеличению длины слова, а затем лексикографически
<table>
<tr><th>Ввод</th><th>Вывод</th></tr>
<tr><td>Классное слово – обороноспособность, которое должно идти после слов: трава и молоко.</td><td>и
идти
слов
слово
трава
должно
молоко
обороноспособность</td></tr>
</table>
</td>
</tr>
</table>

Основные этапы вычисления:

Задание 1

Листинг программы

```
import re

def count_num_emoticons(search_str):
    return len(re.findall(r':<\)', search_str))
```

Тестирование реализовано с использованием unit-тестов

```
import unittest
from task1 import *
import os

class Test1TestCase(unittest.TestCase):
    # (answer = 0)
    def test_zero(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task1", "ZeroAnswer.txt")
        with open(file_path, "r") as file:
            text = file.read()
        res = count_num_emoticons(text)
        self.assertEqual(res, 0)

    # (small answer)
    def test_small_0(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task1", "SmallAnswer_0.txt")
        with open(file_path, "r") as file:
            text = file.read()
        res = count_num_emoticons(text)
        self.assertEqual(res, 6)

    def test_small_1(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task1", "SmallAnswer_1.txt")
        with open(file_path, "r") as file:
            text = file.read()
        res = count_num_emoticons(text)
        self.assertEqual(res, 9)

    # (big answer)
    def test_big_0(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task1", "BigAnswer_0.txt")
        with open(file_path, "r") as file:
            text = file.read()
        res = count_num_emoticons(text)
        self.assertEqual(res, 90145)

    def test_big_1(self):
```

```
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task1", "BigAnswer_1.txt")
        with open(file_path, "r") as file:
            text = file.read()
        res = count_num_emoticons(text)
        self.assertEqual(res, 1412311)
```

Файлы, в которых расположены тексты для тестов:

https://disk.yandex.ru/d/fG4eXjyt2dtwFw

Результат запуска тестов для первого задания:

```
D:\ITMO\sppo_2024\inf\lab3>python -m unittest tests/test_task1.py
.....
----------------------------------------------------------------------
Ran 5 tests in 1.154s

OK
```

Задание 2

Листинг программы

```
import re

def find_sort_last_names(search_text):
    pattern = r'([А-Я][а-я]+(?:-[А-Я][а-я]+)*)(?:\s[А-Я]\.[А-Я]\.)'
    last_names = sorted(set(re.findall(pattern, search_text)))

    return last_names
```

Тестирование реализовано с использованием unit-тестов

```
import unittest
from task2 import *
import os

class Test2TestCase(unittest.TestCase):
    # (answer = 0)
    def test_zero(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task2", "ZeroAnswer.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = find_sort_last_names(text)
        self.assertEqual(res, [])

    # (small answer)
    def test_small_0(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task2", "SmallAnswer_0.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = find_sort_last_names(text)
```

```
        my_res = ['Борисов-Зинченко-Малышев', 'Зайцев', 'Иванов', 'Козлов-
Алексеенко', 'Кузнецов', 'Петрова', 'Попова',
                  'Романов', 'Сидоров', 'Смирнова', 'Соколова', 'Федоров']
        self.assertEqual(res, my_res)

    def test_small_1(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task2", "SmallAnswer_1.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = find_sort_last_names(text)
        my_res = ['Андреев', 'Борисов', 'Васильев', 'Зайцев', 'Иванов',
'Козлов', 'Коновалов', 'Кузнецов', 'Николаев',
                  'Петров', 'Попова', 'Романов', 'Савельев', 'Сидоров',
'Смирнова', 'Соколов', 'Степанов', 'Федоров', 'Филиппов']
        self.assertEqual(res, my_res)

    def test_small_2(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task2", "SmallAnswer_2.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = find_sort_last_names(text)
        my_res = ['Игнатьева', 'Кузнецова-Борисова', 'Макарова', 'Попов',
'Семенов-Иванов', 'Сидорова-Ильина', 'Тихонова', 'Филиппов']
        self.assertEqual(res, my_res)

    # (big answer)
    def test_big_0(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task2", "BigAnswer_0.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = find_sort_last_names(text)
        my_res = ['Андреев', 'Борисов', 'Васильев', 'Васильева', 'Волкова',
'Голубев-Серов',
                  'Григорьев', 'Дмитриев', 'Евдокимов', 'Ефимов-Степанюк-
Бондарев', 'Жаров', 'Зайцев',
                  'Захаров', 'Иванов', 'Иванова', 'Казаков', 'Киселев',
'Климов',
                  'Ковалев', 'Ковалева', 'Козлов', 'Комаров', 'Коновалов',
'Копылов',
                  'Королев', 'Костров', 'Кузнецов', 'Кузьмина', 'Леонов',
'Литвинов-Грибов-Кулаков',
                  'Макаров', 'Малахов', 'Малышев', 'Марков', 'Мартынов',
'Медведев',
                  'Медведева', 'Михайлов', 'Михайлова', 'Моисеев', 'Морозов-
Семенов', 'Наумов',
                  'Нестеров', 'Никитин', 'Никитина', 'Николаев', 'Новиков',
'Новикова',
                  'Петров', 'Попова', 'Романов', 'Савельев', 'Сидоров',
'Смирнова',
                  'Соколов-Завьялов', 'Степанов', 'Федоров', 'Филиппов']
        self.assertEqual(res, my_res)
```

Файлы, в которых расположены тексты для тестов:

https://disk.yandex.ru/d/uTCrS_Tq69g6FQ

Результат запуска тестов для второго задания:

```
D:\ITMO\sppo_2024\inf\lab3>python -m unittest tests/test_task2.py
.....
----------------------------------------------------------------------
Ran 5 tests in 0.004s

OK
```

Задание 3

Листинг программы

```
import re

def has_one_vowel(word):
    list_vowels = ["Аа", "Ее", "Ёё", "Ии", "Оо", "Уу", "Ыы", "Ээ", "Юю",
"Яя"]
    cnt = 0
    for vowel in list_vowels:
        if bool(re.search(rf'[{vowel}]', word)) == False:
            continue
        cnt += 1
        word_without_one_vowel = re.sub(f'[{vowel}]', '', word)

        vowels_to_search = re.sub(f'[{vowel}]', '', 'АаЕеЁёИиОоУуЫыЭэЮюЯя')

        pattern = rf'[{vowels_to_search}]'

        if bool(re.search(pattern, word_without_one_vowel)) == True:
            return False
    if cnt > 0:
        return True
    return False

def my_key(a):
    return (len(a), a.lower())

def search_words_with_one_vowel(search_string):
    words = re.findall(r'\b\w+\b', search_string)

    one_vowel_words = [word for word in words if has_one_vowel(word)]

    return sorted(set(one_vowel_words), key=my_key)
```

Тестирование реализовано с использованием unit-тестов

```
import unittest
from task3 import *
import os

class Test3TestCase(unittest.TestCase):
    # (answer = 0)
    def test_zero(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task3", "ZeroAnswer.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = search_words_with_one_vowel(text)
```

```
        self.assertEqual(res, [])

    # (small answer)
    def test_small_0(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task3", "SmallAnswer_0.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = search_words_with_one_vowel(text)
        my_res = ['рог', 'град', 'стол', 'стул', 'фата', 'шкаф', 'город',
'молоток']
        self.assertEqual(res, my_res)

    def test_small_1(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task3", "SmallAnswer_1.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = search_words_with_one_vowel(text)
        my_res = ['кот', 'пот', 'Рот', 'Ток', 'борт', 'Горб', 'Куст', 'мозг',
'торт', 'Лампа', 'топот', 'Тромб']
        self.assertEqual(res, my_res)

    # (big answer)
    def test_big_0(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task3", "BigAnswer_0.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = search_words_with_one_vowel(text)
        my_res = ['ум', 'бок', 'Ком', 'лак', 'лом', 'пар', 'том', 'угл',
'бинт', 'день', 'клей',
                  'корм', 'корт', 'корь', 'курс', 'лень', 'Литр', 'мост',
'Парк', 'ромб', 'сель', 'сеть',
                  'тост', 'файл', 'фото', 'Ветер', 'вихрь', 'мороз', 'папка',
'спрей', 'Ткань', 'топор',
                  'Смерчь', 'квадрат', 'Лампада', 'Пингвин', 'карандаш',
'сложность']
        self.assertEqual(res, my_res)

    def test_big_1(self):
        cur_dir = os.getcwd()
        file_path = os.path.join(cur_dir,
"tests/files_with_text_for_tests/files_for_task3", "BigAnswer_1.txt")
        with open(file_path, "r", encoding="utf-8") as file:
            text = file.read()
        res = search_words_with_one_vowel(text)
        my_res = ['вал', 'вид', 'Газ', 'код', 'кол', 'нос', 'рис', 'сад',
'сук', 'шок', 'ямб',
                  'блок', 'боль', 'врач', 'даль', 'дань', 'дача', 'круг',
'лист', 'мышь',
                  'риск', 'Ритм', 'Рост', 'Соль', 'сорт', 'хлеб', 'цвет',
'брешь', 'брошь',
                  'весть', 'въезд', 'гость', 'Грамм', 'грипп', 'дробь',
'кварц', 'квест', 'Кисть',
                  'кость', 'кросс', 'лесть', 'месть', 'нефть', 'носок',
'около', 'пасть', 'плеть', 'Принц',
                  'ртуть', 'Сахар', 'скотч', 'Скрип', 'спрос', 'старт',
'ствол', 'Степь', 'стиль',
```

```
                  'страж', 'страх', 'стриж', 'Строй', 'тембр', 'тесть',
'тросс', 'фрукт', 'хруст',
                  'честь', 'шесть', 'взгляд', 'гвоздь', 'горсть', 'гроздь',
'грусть', 'доктор',
                  'скорбь', 'спектр', 'стресс', 'тормоз', 'шерсть',
'Четверг', 'абракадабра']

        self.assertEqual(res, my_res)
```

Файлы, в которых расположены тексты для тестов:

https://disk.yandex.ru/d/ZzG4MDbF3-xEVw

Результат запуска тестов для третьего задания:

```
D:\ITMO\sppo_2024\inf\lab3>python -m unittest tests/test_task3.py
.....
----------------------------------------------------------------------
Ran 5 tests in 0.014s

OK
```

Результат запуска тестов для трех заданий одной командой

```
D:\ITMO\sppo_2024\inf\lab3>python -m unittest
...............
----------------------------------------------------------------------
Ran 15 tests in 1.048s

OK
```