

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: 3](#_Toc180179305)

[Основные этапы вычисления: 6](#_Toc180179306)

[Задание 1 6](#_Toc180179307)

[Задание 2 7](#_Toc180179308)

[Задание 3 9](#_Toc180179309)

[Результат запуска тестов для трех заданий одной командой 11](#_Toc180179310)

# 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

|  |  |
| --- | --- |
| Номер в ИСУ % 6 | Задание |
| 3 | 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.   |  |  | | --- | --- | | Ввод | Вывод | | 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. | Анищенко  Балакшин  Голованова-Иванова  Машина | |

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 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   |  |  | | --- | --- | | Ввод | Вывод | | Классное слово – обороноспособность, которое должно идти после слов: трава и молоко. | и  идти  слов  слово  трава  должно  молоко  обороноспособность | |

# 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 **1.154**s

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 **0.004**s

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 **0.014**s

OK

## Результат запуска тестов для трех заданий одной командой

D:\ITMO\sppo\_2024\inf\lab3>python *-m* unittest

...............

----------------------------------------------------------------------

Ran **15** tests in **1.048**s

OK