Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Факультет программной инженерии и компьютерных технологий

Информатика

Лабораторная работа №3

Регулярные выражения

Вариант 467802

Выполнила студент

Умарова Амина

Группа №Р3118

Преподаватель: Малышева Татьяна

Алексеевна

г. Санкт-Петербург

2024

Оглавление

[Задание 3](#_Toc184301427)

[Дополнительное задание 3](#_Toc184301428)

[Основные этапы решения: 4](#_Toc184301429)

[Решение доп.задания. 4](#_Toc184301430)

[Заключение: 5](#_Toc184301431)

[Список использованных источников: 5](#_Toc184301432)

# Задание

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].

## Дополнительное задание

* 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.

Вариант 0:

Хайку – жанр традиционной японской лирической поэзии века, известный с XIV века. Оригинальное японское хайку состоит из 17 слогов, составляющих один столбец иероглифов. Особыми разделительными словами – кирэдзи – текст хайку делится на части из 5, 7 и снова 5 слогов. При переводе хайку на западные языки традиционно вместо разделительного слова использую разрыв строки и, таким образом, хайку записываются как трёхстишия. Перед вами трёхстишия, которые претендуют на то, чтобы быть хайку. В качестве разделителя строк используются символы «/». Если разделители делят текст на строки, в которых 5/7/5 слогов, то выведите «Хайку!». Если число строк не равно 3, то выведите строку «Не хайку. Должно быть 3 строки.». Иначе выведите строку вида «Не хайку.» Для простоты будем считать, что слогов ровно столько же, сколько гласных, не задумываясь о тонкостях.

Пример:

|  |  |
| --- | --- |
| Ввод | Вывод |
| Вечер за окном. / Еще один день прожит. / Жизнь скоротечна… | Хайку! |
| Просто текст | Не хайку. Должно быть 3 строки. |
| Как вишня расцвела! / Она с коня согнала / И князя-гордеца. | Не хайку. |

* 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.

Вариант 2:

Вливаемся в роль фрилансера! Нужно найти (распарсить) цену валюты Bitcoin

используя регулярные выражения для частного сайта заказчика.

Пример ввода:

<meta name="daily\_volume" content="В суточным объемом торгов

₽2,835,029,974,960.63 RUB."/> <meta name="daily\_price" content="Мы обновляем

нашу цену BTC к RUB в режиме реального времени."/> <meta name="daily\_price"

content=" Цена Bitcoin в реальном времени сегодня составляет ₽5,797,806.88

RUB."/><meta name="daily\_price" content="Ethereum стоит на данный момент

₽229,590,78 RUB."/>

Пример вывода:

5,797,806.88

Основные этапы решения:

*Вариант 8-{P*

**import** re

regexp = r"8-{P"

**def** test(value):

match = re.findall(regexp, value)

**print**(**len**(match))

test("")

test("lj9i9989898-{Pkkkk")

test("8-{P 8-{ P8 -{P")

test("8-{P8-{P8-{P")

test("\8-\\{P \\8-{\P\} ")

## Решение доп.задания.

**import** re

**def** is\_haiku(text):

lines = re.split(r'\s\*/\s\*/\s', text.strip())

**if** **len**(lines) != 3:

**return** "Не хайку. Должно быть 3 строки."

vowels = r"[аеёиоуыэюя]"

syllables = [**len**(re.findall(vowels, line.lower())) **for** line **in** lines]

**if** syllables == [5, 7, 5]:

**return** "Хайку!"

**else**:

**return** "Не хайку."

test\_cases = [

("Она меня ждет, / Лолита - свет и тоска. / Сердце вновь болит.", "Хайку!"),

("Просто текст", "Не хайку. Должно быть 3 строки."),

("Как вишня расцвела! / Она с коня согнала / ", "Не хайку."),

("Ветер унесет. / Где цветы не умирают. / Там есть мой покой.", "Хайку!"),

("Снег идет опять. / Сколько лет пройдет без нас? / Тихо ночь лежит.", "Хайку!"),

]

**for** i, (text, expected) **in** **enumerate**(test\_cases, 1):

result = is\_haiku(text)

**print**(f"Тест {i}: {'Успех' **if** result == expected else 'Провал'} — Ожидалось: {expected}, Получено: {result}")

#3

**def** extract\_bitcoin\_price(text):

match = re.findall(r'Цена Bitcoin.\*?[$₽]([\d,]+\.\d{2})', text)

**return** match[-1] **if** match **else** None

tests = {

"Тест 1": ('Какая-то информация... Цена Bitcoin на сегодня: ₽2,000,000.00 RUB.', "2,000,000.00"),

"Тест 2": ('Цена Bitcoin составляет ₽123.45 RUB.Цена Bitcoin составляет ₽123.50 RUB.', "123.50"),

"Тест 3": ('Ethereum стоит ₽500,123.00 RUB.', None),

"Тест 4": ('Цена Litecoin сейчас ₽250,000.00 RUB.', None),

"Тест 5": ('Цена Bitcoin равна ₽', None),

"Тест 6": ('Просто текст без цены.', None),

}

**for** name, (input\_text, expected) **in** tests.items():

result = extract\_bitcoin\_price(input\_text)

**print**(f"{name}: Ожидалось: {expected}, Получено: {result}")

# Заключение:

В ходе лабораторной работы я научился эффективно использовать регулярные выражения для извлечения данных из текста в Python. Я понял, как создавать паттерны для поиска конкретных фрагментов текста и как использовать функции, такие как re.search() и re.findall(), для обработки строк и извлечения нужной информации.

# Список использованных источников:

1. Иванов И. Что такое регулярные выражения и как они работают? // Habr. – 2017. – URL: <https://habr.com/ru/articles/349860/> (дата обращения: 05.12.2024).
2. Тейлор, Д., Терри, Б. **Сценарии командной оболочки Linux, OS X, Unix**: 2-е изд. — М.: Изд-во "Вильямс", 2017. — 464 с.