Министерство Образования и Науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

ЛАБОРАТОРНАЯ РАБОТА № 3

«Регулярные выражения»

по дисциплине

«Информатика»

Выполнил:

Студент группы P3130

Гуренков Максим Сергеевич

Проверил:

Балакшин Павел Валерьевич

Санкт-Петербург  
2023

Содержание

[Задание 3](#_Toc136003821)

[Основные этапы вычисления 4](#_Toc136003822)

[Вывод 7](#_Toc136003823)

# Задание

1. Основное 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.

3) Программа должна считать количество 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.

Вариант: 368069%6 = 5, 368069%4 = 1, 368069%7 = 2 – “[<O”

1. Дополнительное задание 1

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. Пример тестов приведён в таблице.

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

Вариант: 368069%6 = 5

С помощью регулярного выражения найти в тексте все слова, в которых две гласные стоят подряд, а после этого слова идёт слово, в котором не больше 3 согласных.

1. Дополнительное задание 2

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

2) Для своей программы придумайте минимум 5 тестов.

3) Протестируйте свою программу на этих тестах.

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

Вариант: 368069%5 = 4

Необходимо выбрать три любых буквы и расстояние между ними. С помощью регулярного выражения нужно найти все слова (последовательность символов ограниченная пробелами), в которых встречаются эти буквы в заданной последовательности и расстояние (например, через один друг от друга).

# Основные этапы вычисления

1. Основное задание

Программа считает количество смайликов вида “[<O” в предложенном тексте.

Было разработано две функции: одна, использующая регулярные выражения (см. рисунок 1), другая – нет (см. рисунок 2).

Чтобы продемонстрировать преимущества регулярных выражений в выполнении этой задачи, был реализован дополнительный функционал: теперь функции имеют два режима: если в качестве аргумента функциям передается True, то учитываются только те смайлики, которые ограждены с двух сторон пробелами (те, что расположены вплотную к слову или непосредственно в нём, не учитываются), если False, то учитываются все смайлики, как бы они не были расположены в тексте. Тогда можно заметить, что для реализации вышесказанного, для функции, использующей регулярное выражение, необходимо было добавить условный оператор и убрать несколько символов в выражении, когда для второй функции пришлось осуществлять абсолютно новую логику.

Разработанные тестовые значения и результат работы программы представлены ниже, на рисунках 3 и 4 соответственно.

Для реализации вычислений с помощью регулярных выражений была использована библиотека “re”.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1 – функция с регулярным выражением

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 2 – функция без регулярного выражения

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3 – тестовые значения

Изображение выглядит как текст, снимок экрана, меню

Автоматически созданное описание

Рисунок 4 – результат выполнения

1. Дополнительное задание 1

Программа ищет слова, в которых стоят две гласные подряд и после которых находится слово, содержащее не более чем 3 согласных, в предложенном тексте.

Также было разработано две функции: одна, использующая регулярные выражения (см. рисунок 5), другая – нет (см. рисунок 6).

Разница в объеме кода в обоих функциях выглядит намного существеннее, чем в основном задании, ввиду относительной сложности задачи: функция, использующая регулярные выражения занимает две строчки, учитывая перенос кода, вторая функция – двадцать одну.

Разработанные тестовые значения и результат работы программы представлены ниже, на рисунках 7 и 8 соответственно.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 5 – функция с регулярным выражением

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6 – функция без регулярного выражения

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 7 – тестовые значения

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 8 – результат работы программы

1. Дополнительное задание 2

Программа ищет слова, содержащее определенные 3 буквы в определенной последовательности с определенным расстоянием между этими буквами (значения динамические – задаются в самой программе)

Было разработано две функции: одна, использующая регулярные выражения (см. рисунок 9), другая – нет (см. рисунок 10).

Разница в объеме кода в обоих функциях также значительная: функция, использующая регулярные выражения занимает три строчки, учитывая перенос кода, вторая функция – шестнадцать, также учитывая перенос кода. Они обе используют ещё одну вспомогательную функцию addUpperCase(), предназначенную для того, чтобы поиск слов был регистронезависимым (чтобы соответствие искалось с буквами как верхнего, так и нижнего регистра). Её код представлен на рисунке 11.

В качестве тестовых значений приведены 5 текстов, соответствующие им буквы и значения дистанций между ними. Соответствие проводится по порядковому номеру внутри списка (первый текст из списка с текстами соответствует первому набору букв из списка с наборами букв и первому значению из списка дистанций). Тестовые значения и результат работы программы представлены ниже, на рисунках 12 и 13.1–13.2 соответственно.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 9 – функция с регулярным выражением

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 10 – функция без регулярного выражения

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Рисунок 11 – функция addUpperCase()

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 12 – тестовые значения

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 13.1 – результат работы программы

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 13.2 – результат работы программы

Полный код программы доступен по ссылке: https://github.com/mgurenk0v/Informatics/tree/main/Lab3/regular\_expressions

# Вывод

В ходе выполнения лабораторной работы №3 я узнал, что такое регулярные выражения и научился использовать их для поиска строк в тексте.

Список литературы

1. Регулярные выражения. Всё проще, чем кажется: [Электронный ресурс]. URL: https://habr.com/ru/articles/567106/ (Дата обращения: 30.05.2023)
2. Шпаргалка по регулярным выражениям: [Электронный ресурс]. URL: http://website-lab.ru/article/regexp/shpargalka\_po\_regulyarnyim\_vyirajeniyam/?ysclid=lic6nwu8a9222893682 (Дата обращения: 30.05.2023)