# Программирование 24 1 17

Предположим, нам нужно найти в тексте слово с умляутом, как, напр, Händel. Оно может быть записано разныи способами, например: Handel, Haendel. Как это сделать наиболее рационально?

Регулярные выражения.

Для них нужен особый модуль: **re**

H(ä|ae?)ndel

| or

? предыдущего символа может и не быть

() – группировка

. один любой символ

\* предыдущий символ (группа) может повторяться от 0 до бесконечного числа раз

+ предыдущий символ (группа) может повторяться положительное число раз [1,..]

Чтобы эти символы воспринимались как обычные, используется также \

##Match – часть строки, подходящая под регулярные выражения.

Примеры:

И(б|в)?ол?

При возникновении неоднозначности – «жадность»: положим, под паттерн попадает несколько пересекащихся элементов, то берется тот, который больше. + --квантификатор, жадный по умолчанию.

Ограничение жадности квантификатора: .+? Рассматриваемый пример: до.\*е+ (жадный) или до.\*е+? (нежадный, т.е. берет тот элемент, что меньше, первый попавшийся, по всей видимости).

Еще символы:

[…] –один из перечисленных символов, например: к[оа]рова (похоже на к(о|а)рова)

[…-…] – один символ из диапозона [а-я]. Если сделать так [-…] или так […-], то ничего не будет работать.

Пример запроса: [а-яА-ЯA-Zbq ] (никаких запятых не нужно)

В запросах на русский алфавит ё не входит, поэтому пишем так, чтобы ее включить: [а-яё]

Все формы глагола to have: Ha(s|d|v(e|ing))

Слова, имеющие хотя-бы два слога: .\*[аеёюяи<>э] #не дописал

Теперь использование конкретно в Питоне

Поиск слов, соответстующих регулярному выражению

Import re

M = re.search(regex(регулярное выражение – заданное заранее или прямо здесь написанное),s)

**На месте s – только строка**

Но там еще нет слова, это только проверяет, есть ли слово, подходящее под паттерн

S ‘abc’

If re.search(‘[ab]’,s):

Print(“Нашлось”)