

Что это и зачем?

Регулярные выражения (*regular expressions*, *RegExp*) — это формальный язык для операций с подстроками.

Чаще всего регулярные выражения используются для:

- поиска в строке;
- разбиения строки на подстроки;
- замены части строки;
- валидации (проверки).

Синтаксис

Оператор	Описание
.	Один любой символ, кроме новой строки \n.
?	0 или 1 вхождение шаблона слева
+	1 и более вхождений шаблона слева
*	0 и более вхождений шаблона слева
\w	Любая цифра или буква (\w — все, кроме буквы или цифры)
\d	Любая цифра [0-9] (\d — все, кроме цифры)
\s	Любой пробельный символ (\s — любой непробельный символ)
\b	Граница слова
[...]	Один из символов в скобках ([^ . .] — любой символ, кроме тех, что в скобках)
\	Экранирование специальных символов (\ . означает точку или \+ — знак «плюс»)
^ и \$	Начало и конец строки соответственно
{n,m}	От n до m вхождений ({ ,m} — от 0 до m)
a b	Соответствует a или b
()	Группирует выражение и возвращает найденный текст
\t, \n, \r	Символ табуляции, новой строки и возврата каретки соответственно

Упражнения

- 1) Смотрим, как работают операторы, на regexr.com (вставляем любой текст и ищем в нем подстроки). Особые случаи:
 - * vs *?
 - + vs +?
 - \\\
- 2) Выполняем все упражнения на regexone.com.
Дополнительное условие: в match должна подсветиться вся строка.

Модуль re

Документация:

<https://docs.python.org/2/library/re.html>

Методы:

- `re.match()` — поиск совпадения в начале строки
- `re.search()` — поиск первого совпадения
- `re.findall()` — поиск всех совпадений (возвр. список)
- `re.split()` — разбиение строки
- `re.sub()` — замена
- **`re.compile()` — компиляция**
- ...