12.09.2023

Регулярные выражения в реальной задаче

Ошуркова Татьяна,
Росбанк. Старший ИТ-менеджер

Flow настоящие возможности

росбанк 30 лет

О себе

Работаю в банковской сфере

Росбанк, Кредит Урал Банк (группа Газпромбанка)

Разработчик баз данных

А также web-разработчик, аналитик

Преподаватель программирования

Преподавала курсы для детей и взрослых

О чем будем говорить

Что такое регулярные выражения

Наиболее частые области
применения регулярных выражений

Синтаксис регулярных выражений

Примеры использования
регулярных выражений

Регулярные выражения

Регулярные выражения - это механизм для поиска и замены текста

Задачи регулярок

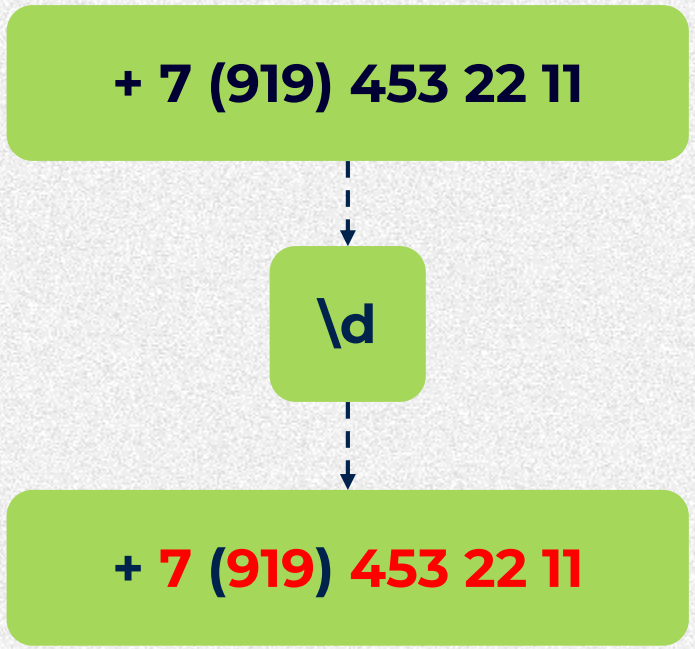
- Определить наличие последовательности символов
- Найти последовательность символов
- Заменить последовательность символов

Примеры реальных задач регулярных выражений

- Выполнить валидацию данных
- Проверить тип файла
- Найти данные заданного формата
- Удалить некорректные символы

Как составлять регулярки

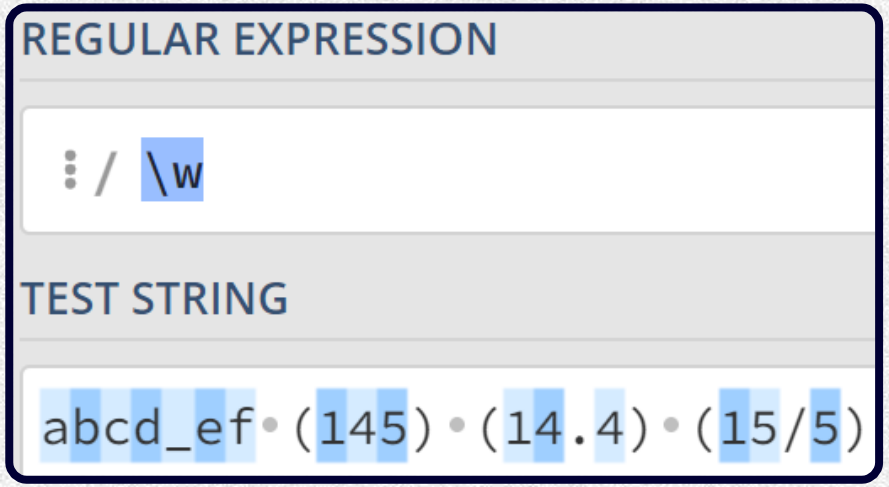
Символьные классы (character classes)

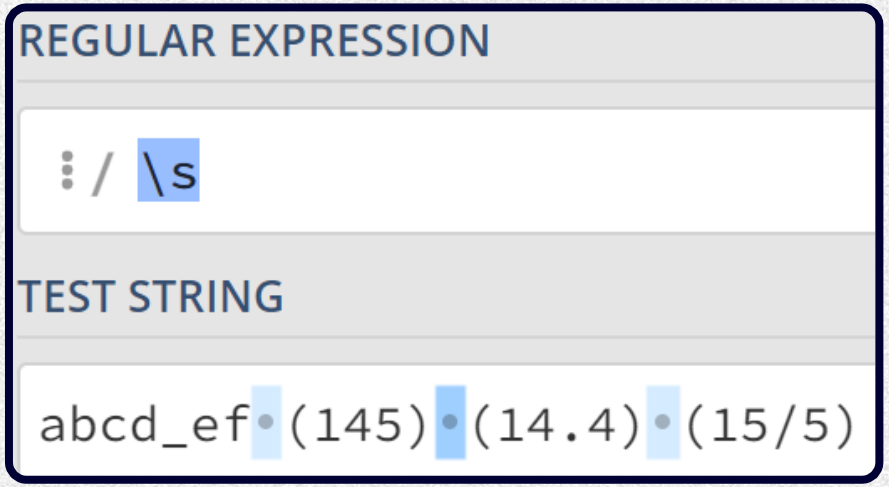


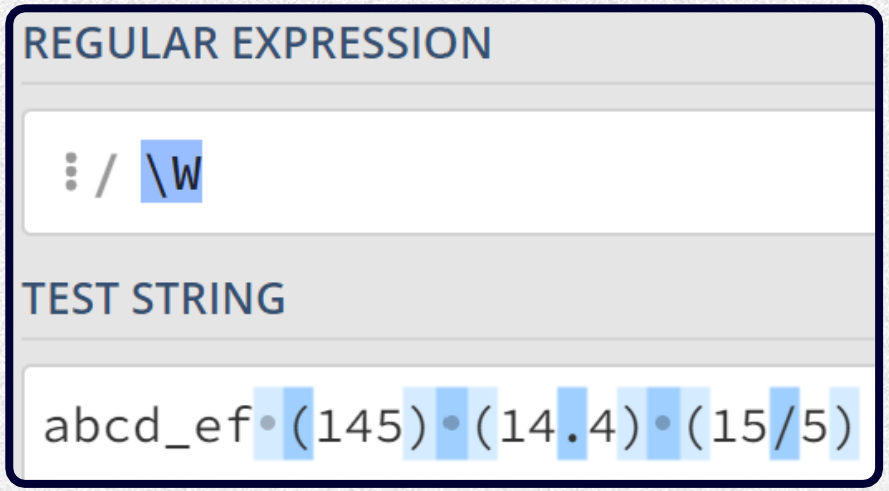
\d	digit - цифра
\s	space - пробел
\w	word - слово
\D	не цифра (все кроме \d)
\S	не пробел (все кроме \s)
\W	не латиница, не знак подчёркивания и не цифра
.	любой символ кроме новой строки

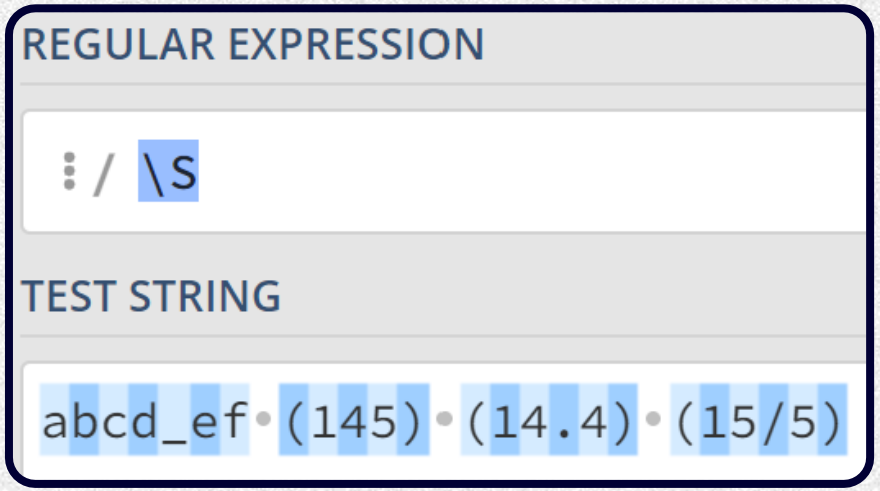
Символьные классы

Примеры









Как составлять регулярки

Начало и конец строки

Начало строки	Конец строки	Совпадение строки
test request bg request start end request ↓ ^request ↓ test request bg request start end request	end of req test end start end bg ↓ end$ ↓ end of req test end start end bg	11:12 max call 16:45 ↓ ^\d\d:\d\d$ ↓ 11:12 max call 16:45

Как составлять регулярки

Наборы (sets) и диапазоны (ranges)

Набор	Диапазон	Исключающий диапазон
srer strat seer	req1 reqt req0 req_	req1 reqt req0 req_
s[are]er	req[a-z1-9_]	req[^1-3a-c]
srer strat seer	req1 reqt req0 req_	req1 reqt req0 req_

Немного практики

Кредитные договоры хранятся в формате трёхзначный идентификатор-год. Какие кредитные договоры **подходят под написанное регулярное выражение?**

1. 25c-2022
2. 31f-2022
3. 32b-2021
4. 72g-2020
5. 55a-2024

\d[2,5][a-e]-202[0-2]

Немного практики

Кредитные договоры хранятся в формате трёхзначный идентификатор-год. Какие кредитные договоры **подходят под написанное регулярное выражение?**

1. 25c-2022
2. 31f-2022
3. 32b-2021
4. 72g-2020
5. 55a-2024

\d[2,5][a-e]-202[0-2]

1. **25c-2022**
2. 31f-2022
3. **32b-2021**
4. 72g-2020
5. 55a-2024

Как составлять регулярки

Квантификаторы (quantifiers)

Количество	Обозначения	
start 123wrw 12tyws	?	{0,1}
\d{3,5}\w{3,}	+	{1,}
start 123wrw 12tyws	*	{0,}

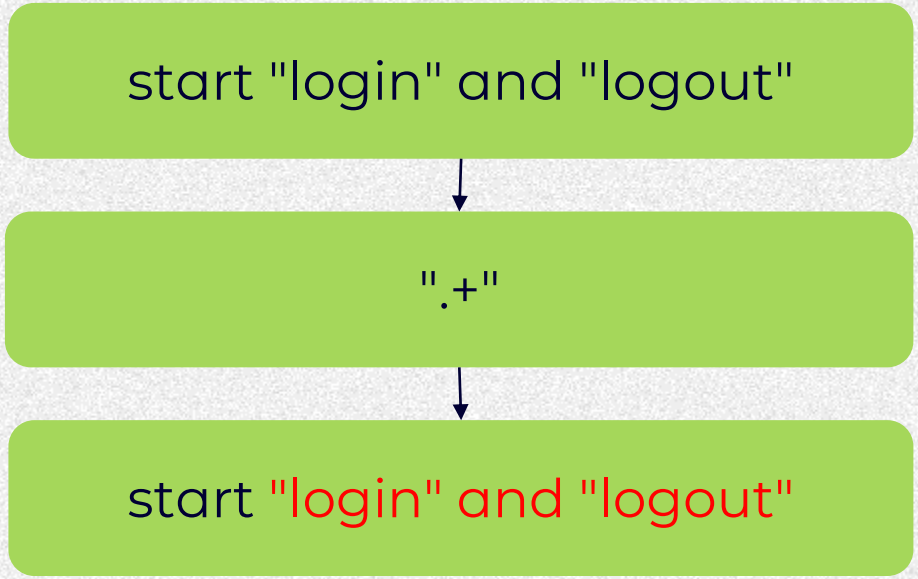
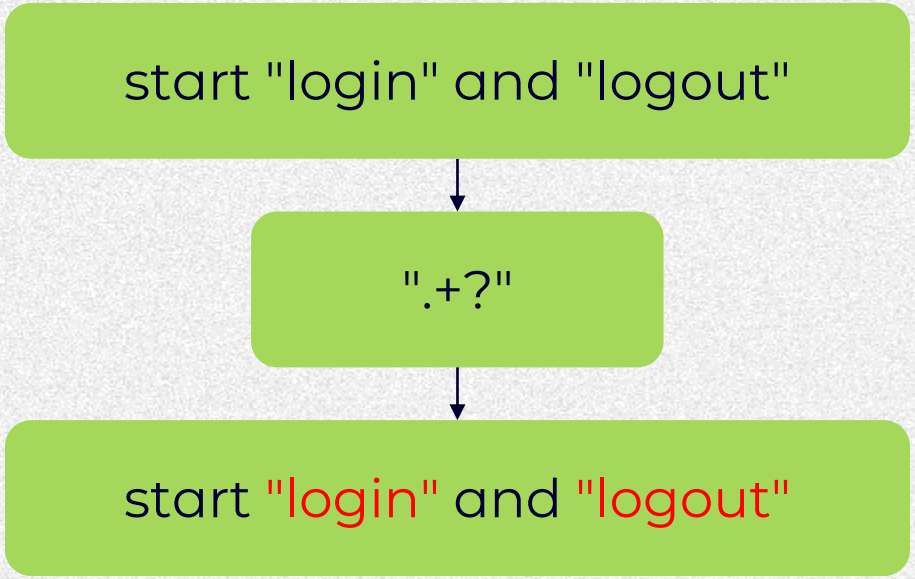
100 15 10.24 45.782 145.3 78.675

[\s^]\d+\.\d{1,2}[\s$]

100 15 10.24 45.782 145.3 78.675

Как составлять регулярки

Жадная (greedy) и ленивая (lazy) квантификация

Жадная	Ленивая
	

Как составлять регулярки

Группы (groups)

my@mail.com
my?@mail
his@site.com.uk
Mike.Tyson@yandex

↓

[-.\w]+@([\w-]+\.)+\w+

↓

my@mail.com
my?@mail
his@site.com.uk
Mike.Tyson@yandex

20.02.2022
2023-11-22
10-10-2012

↓

(\d{1,2})([-./])(\d{1,2})([-./])(\d{4})

↓

20.02.2022
2023-11-22
10-10-2012

Как составлять регулярки

Группы с захватом (capturing) и без (non-capturing)

20.02.2022
2023-11-22
10-10-2012
10-10.2012

(\d{1,2})([-./])(\d{1,2})\2(\d{4})$

20.02.2022
2023-11-22
10-10-2012
10-10.2012

20.02.2022
2023-11-22
10-10-2012
10-10.2012

(?:\d{1,2})([-./])(\d{1,2})\1(\d{4})$

20.02.2022
2023-11-22
10-10-2012
10-10.2012

Немного практики

Необходимо выполнить соответствие логина, который создает пользователь, определенным требованиям. **Какие из логинов корректны в соответствии с написанным для валидации регулярным выражением?**

1. kate23@mail
2.nikita.8.9
3.maria_petrova
4.evgenia1995
5.oleg2-23
6.Dima01

[a-z0-9_.]{3,11}

Немного практики

Необходимо выполнить соответствие логина, который создает пользователь, определенным требованиям. **Какие из логинов корректны в соответствии с написанным для валидации регулярным выражением?**

1. kate23@mail
2.nikita.8.9
3.maria_petrova
4.evgenia1995
5.oleg2-23
6.Dima01

[a-z0-9_.]{3,11}

1. kate23@mail
2.nikita.8.9
3.maria_petrova
4.evgenia1995
5.oleg2-23
6.Dima01

Какие задачи мы рассмотрим

Валидация данных и определение
направления платежа в платежном поручении

Разбор логов. Найти нужный запрос,
достать строку с ключевыми словами

Задача с платежным поручением

Определение правил в реквизитах

Формализация средствами регулярных выражений

Создание sql запроса

Определение правил в реквизитах

Корреспондентский счет

- Начало счета
- Соответствие счету получателя

Счет получателя платежа

- Начало счета (план)
- Сочетание цифр в счете

Формализация правил

Корр. счет	Счёта получателя
(*)	^475(*)
(*)	^40[392]09.....4(*)
(*)	^60323(*)
^40102(*)	^3100(*)
^40102(*)	^3212(*)
^40102(*)	^3222(*)
^40102(*)	^41[892]07.....5(*)
^40102(*)	^03242(*)

Начало строки

Список совпадения

Любые символы и количество

Соответствие

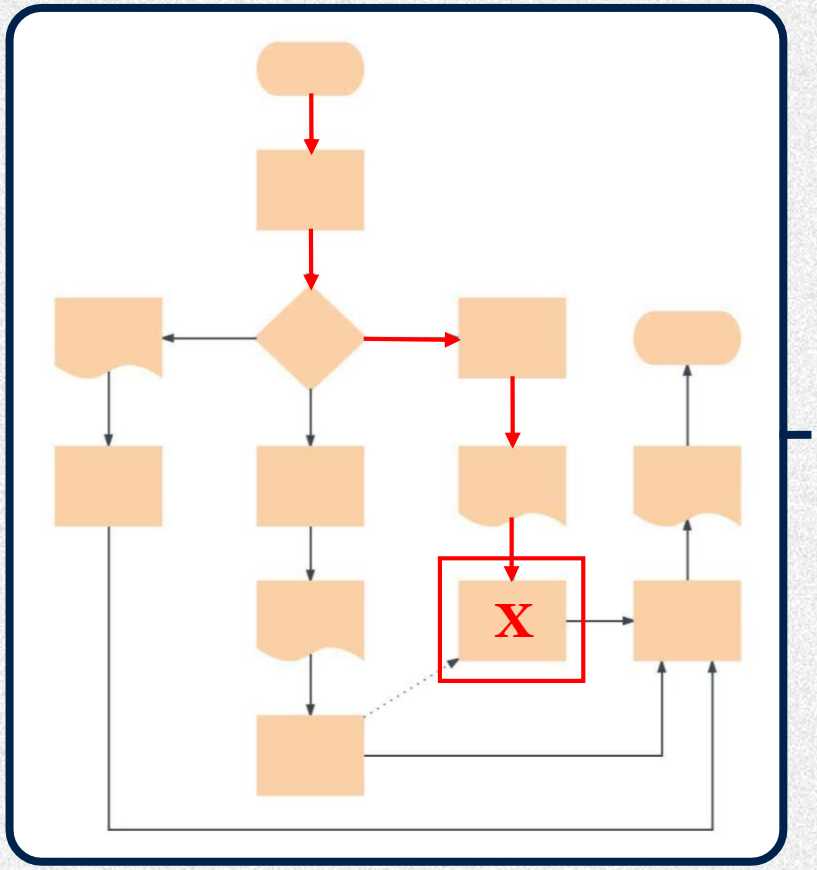
Регулярные выражения в запросе

Таблица payment_rules

BIC	CORR_ACCOUNT	PAY_ACCOUNT	KBK	DIRECTION
(*)	(*)	^68097(*)	(*)	Перевод внутри банка
(*)	^40307(*)	^3541(*)	(*)	Бюджетный платеж
(*)	^40307(*)	^3921(*)	(*)	Бюджетный платеж

```
select pr.DIRECTION
from payment_rules pr
where REGEXP_LIKE(v_Bank_Bik, pr.BIC)
and REGEXP_LIKE(v_Corr_Account, pr.CORR_ACCOUNT)
and REGEXP_LIKE(v_Pay_Account, pr.PAY_ACCOUNT)
and REGEXP_LIKE(v_KBK, pr.KBK)
```

Поиск ошибки



BIC	CORR_ACCOUNT	PAY_ACCOUNT	KBK	DIRECTION
(*)	(*)	^68097(*)	(*)	Перевод внутри банка
(*)	^40307(*)	^3541(*)	(*)	Бюджетный платеж
(*)	^40307(*)	^3921(*)	(*)	Бюджетный платеж

```
select pr.DIRECTION
from payment_rules pr
where REGEXP_LIKE(v_Bank_Bik, pr.BIC)
and REGEXP_LIKE(v_Corr_Account, pr.CORR_ACCOUNT)
and REGEXP_LIKE(v_Pay_Account, pr.PAY_ACCOUNT)
and REGEXP_LIKE(v_KBK, pr.KBK)
```

Пример поиска ошибки

40307891425361	35217829103243 7	Перевод внутри банка

BIC	CORR_ACCOUNT	PAY_ACCOUNT	KBK	DIRECTION
(*)	^40307(*)	^3541(*)	(*)	Бюджетный платеж
(*)	^40307(*)	^3521(*)	(*)	Перевод внутри банка

Настройки с использованием регулярных выражений

BIC	CORR_ACCOUNT	PAY_ACCOUNT	KBK	DIRECTION
(*)	(*)	^68097(*)	(*)	Перевод внутри банка
(*)	^40307(*)	^40[392]09.....4(*)	(*)	Бюджетный платеж
(*)	^40308(*)	^40[392]07.....5(*)	(*)	Платеж в другой банк

Задача с логами

Файл лога

```
03-29-13:17:38.600-14828_14828:I open_connection-...some_text.
03-29-13:17:38.651-14828-14828:D MainModule:
03-29-13:17:40.655_14828-14828:D InputProcessor:8
03-29-13:17:40.657-14828-14828:D MainModule:восемь
03-29_13:17:40.658-14828_15420:D InputCommand:8
...some_text...
...some_text...
...some_text...
03-29_13:17:41.239-14828-15420:D WmsApi:Request:Login_request('03-429-
2022_13:17:40','arg_1','arg_2','arg_31','arg_A','arg_51,'arg_6','arg_7','arg_8','arg_9',
'arg_101,'arg_11','arg_12','[CDATA[O......]]')
...some_text...
...some_text...
...some_text...
```

Большой объем

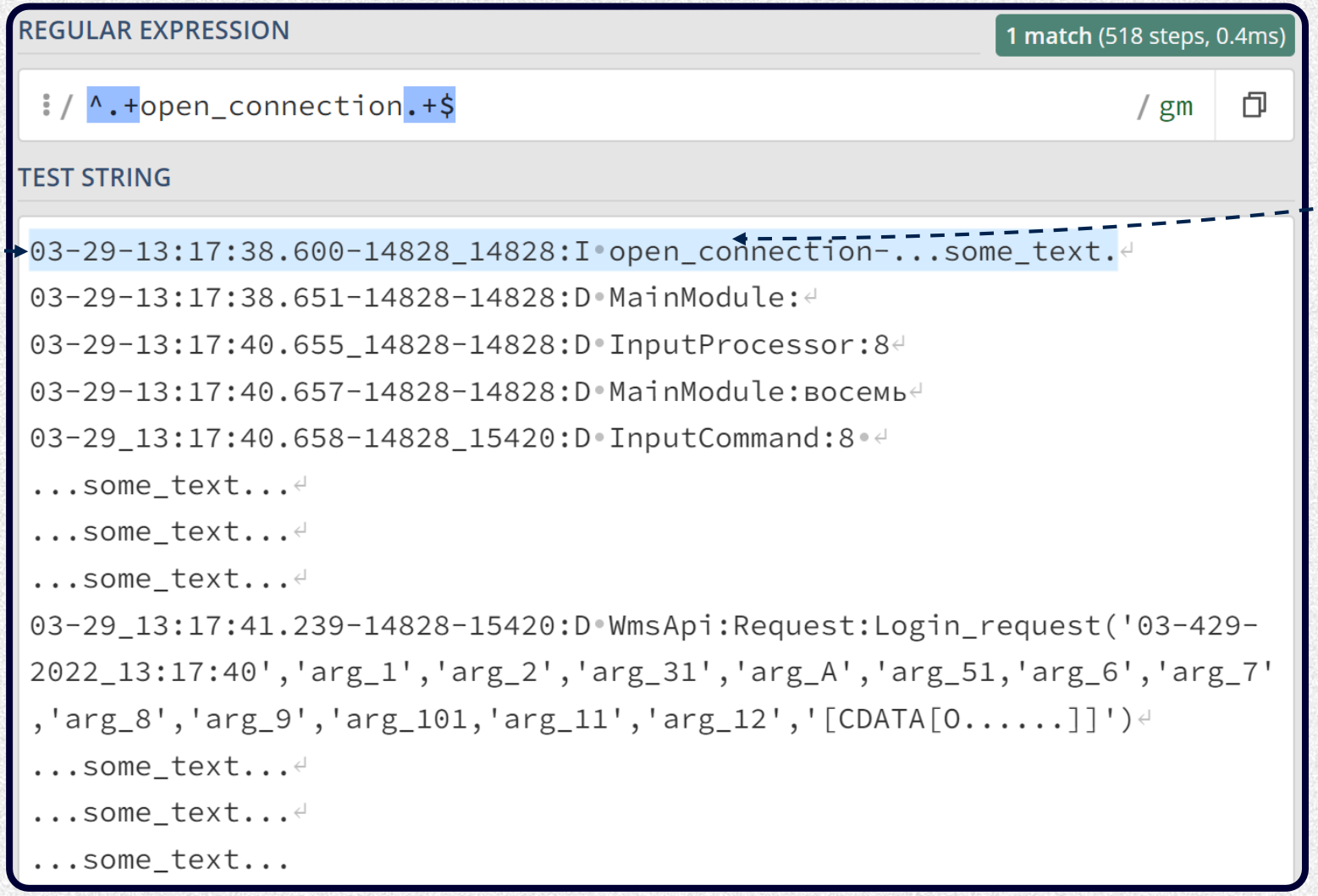
Нерелевантные данные

Сложные правила отбора

Определение формата

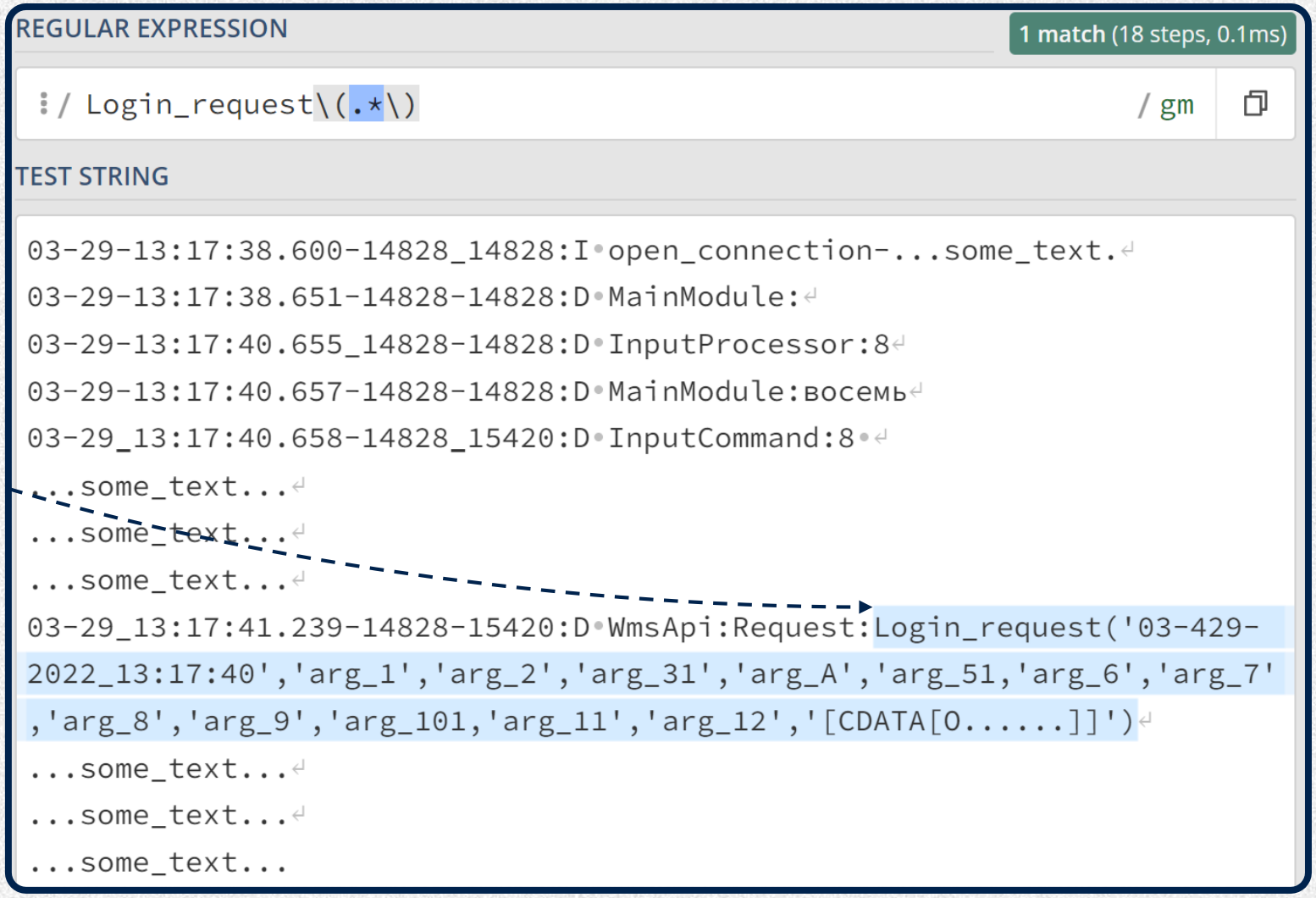
Цифровые данные

Ключевые слова

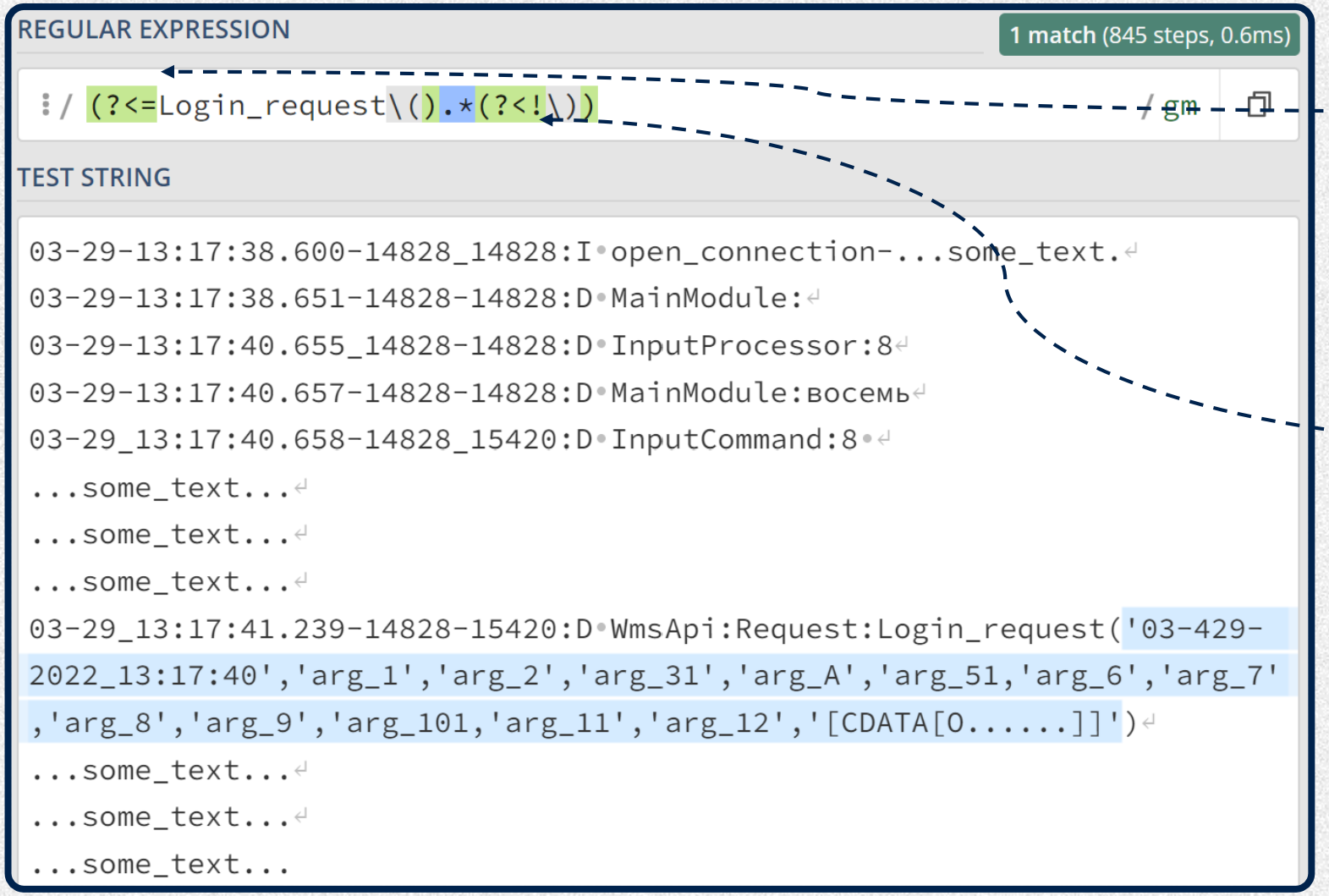


Выделение ключевых слов/символов

Нужны данные
из Login_request



Требования к результату



(?<=Y)X
ищет совпадение с X,
если перед ним есть Y

(?<!Y)X
ищет совпадение с X
при условии, что перед
ним нет Y

Опережающие
и ретроспективные проверки
(lookahead and lookbehind)

Проверка регулярного выражения

REGULAR EXPRESSION

```
/ (?<=test)\.txt
```

TEST STRING

```
test.txt
readme.txt
mytest.txt
main_file.txt
```

SUBSTITUTION

```
.log
```

```
test.log
readme.txt
mytest.log
main_file.txt
```

```python
import re

regex = r"(?<=test)\.txt"

test_str = ("test.txt\n"
            "readme.txt\n"
            "my_test.txt\n"
            "main_file.txt")

subst = ".log"

result = re.sub(regex, subst, test_str)

print(result)

#   test.log
#   readme.txt
#   my_test.log
#   main_file.txt
```

Ограничения использования регулярных выражений

- Ограниченность функционала
- Зависимость от применяемой технологии
- Сложность прочтения/понимания
- Сложная адаптация одной реализации к разным задачам

Итоги. Плюсы использования регулярных выражений

- Возможность решения широкого класса задач
- Необходимый инструмент при работе с текстовыми документами
- Кроссплатформенность
- Быстрый поиск и анализ текстовых данных

Будет полезно

regex101.com

regexcrossword.com

справка на GitHub