Шаблоны, соответствующие одному символу

Шаблон	Описание	Пример	Применяем к тексту
`.`	Один любой символ, кроме новой строки `\n`.	`м.л.ко`	молоко, малако, Им0л0коИхлеб
`\d`	Любая цифра	`СУ\d\d`	СУ35, СУ111, АЛСУ14
`\D`	Любой символ, кроме цифры	`926\D123`	926)123, 1926-1234
`\s`	Любой пробельный символ (пробел, табуляция, конец строки и т.п.)	`бор\sода`	бор ода, бор ода, борода
`\S`	Любой непробельный символ	`\S123`	X123, я123, !123456, 1+123456
`\w`	Любая буква (то, что может быть частью слова), а также цифры и _	`\w\w\w`	Год, f_3, qwert
`\W`	Любая не-буква, не-цифра и не подчёркивание	`сом\W`	сом!, сом?
`[..]`	Один из символов в скобках, а также любой символ из диапазона `a-b`	`[0-9][0-9A-Fa-f]`	12, 1F, 4B

Шаблон	Описание	Пример	Применяем к тексту
`[^..]`	Любой символ, кроме перечисленных	`<[^>]>`	<1>, <a>, <>
`\d≈[0-9]`, `\D≈[^0-9]`, `\w≈[0-9a-zA-Z a-яA-ЯёЁ]`, `\s≈[ \f\n\r\t\v]`	Буква “ё” не включается в общий диапазон букв! Вообще говоря, в `\d` включается всё, что в юникоде помечено как «цифра», а в `\w` — как буква. Ещё много всего!		
`[abc-]`, `[-1]`	если нужен минус, его нужно указать последним или первым		
`[*[(+\\\]\t]`	внутри скобок нужно экранировать только `]` и `\`		
`\b`	Начало или конец слова (слева пусто или не-буква, справа буква и наоборот). В отличие от предыдущих соответствует позиции, а не символу	`\bвал`	вал, перевал, Перевалка
`\B`	Не граница слова: либо и слева, и справа буквы, либо и слева, и справа НЕ буквы	`\Bвал`	перевал, вал, Перевалка
		`\Bвал\B`	перевал, вал, Перевалка

Квантификаторы (указание количества повторений)

Шаблон	Описание	Пример	Применяем к тексту
`{n}`	Ровно n повторений	`\d{4}`	1, 12, 123, 1234, 12345
`{m,n}`	От m до n повторений включительно	`\d{2,4}`	1, 12, 123, 1234, 12345
`{m,}`	Не менее m повторений	`\d{3,}`	1, 12, 123, 1234, 12345
`{,n}`	Не более n повторений	`\d{,2}`	1, 12, 123
`?`	Ноль или одно вхождение, синоним `{0,1}`	`валы?`	вал, валы, валов
`*`	Ноль или более, синоним `{0,}`	`СУ\d*`	СУ, СУ1, СУ12, ...
`+`	Одно или более, синоним `{1,}`	`a\)+`	a), a)), a))), ba)])

`*?`
`+?`
`??`
`{m,n}?`
`{,n}?`
`{m,}?`
	По умолчанию квантификаторы *жадные* — захватывают максимально возможное число символов. Добавление `?` делает их *ленивыми*, они захватывают минимально возможное число символов	`\(.*\)`
		`\(.*?\)`
			(a + b) * (c + d) * (e + f)
			(a + b) * (c + d) * (e + f)

Регулярки в питоне

Функция	Её смысл
`re.search(pattern, string)`	Найти в строке `string` первую строчку, подходящую под шаблон `pattern`;
`re.fullmatch(pattern, string)`	Проверить, подходит ли строка `string` под шаблон `pattern`;
`re.split(pattern, string, maxsplit=0)`	Аналог `str.split()`, только разделение происходит по подстрокам, подходящим под шаблон `pattern`;
`re.findall(pattern, string)`	Найти в строке `string` все непересекающиеся шаблоны `pattern`;
`re.finditer(pattern, string)`	Итератор по всем непересекающимся шаблонам `pattern` в строке `string` (выдаются `match`-объекты);
`re.sub(pattern, repl, string, count=0)`	Заменить в строке `string` все непересекающиеся шаблоны `pattern` на `repl`;

Пример использования всех основных функций

```
import re

match = re.search(r'\d\d\D\d\d', r'Телефон 123-12-12')
print(match[0] if match else 'Not found')
# -> 23-12
match = re.search(r'\d\d\D\d\d', r'Телефон 1231212')
print(match[0] if match else 'Not found')
# -> Not found

match = re.fullmatch(r'\d\d\D\d\d', r'12-12')
print('YES' if match else 'NO')
# -> YES
match = re.fullmatch(r'\d\d\D\d\d', r'T. 12-12')
print('YES' if match else 'NO')
# -> NO

print(re.split(r'\W+', 'Где, скажите мне, мои очки??!'))
# -> ['Где', 'скажите', 'мне', 'мои', 'очки', '']

print(re.findall(r'\d\d\.\d\d\.\d{4}',
                 r'Эта строка написана 19.01.2018, а могла бы и 01.09.2017'))
# -> ['19.01.2018', '01.09.2017']

for m in re.finditer(r'\d\d\.\d\d\.\d{4}', r'Эта строка написана 19.01.2018,
а могла бы и 01.09.2017'):
    print('Дата', m[0], 'начинается с позиции', m.start())
# -> Дата 19.01.2018 начинается с позиции 20
# -> Дата 01.09.2017 начинается с позиции 45

print(re.sub(r'\d\d\.\d\d\.\d{4}',
             r'DD.MM.YYYY',
             r'Эта строка написана 19.01.2018, а могла бы и 01.09.2017'))
# -> Эта строка написана DD.MM.YYYY, а могла бы и DD.MM.YYYY
```

Использование дополнительных флагов в питоне

Константа	Её смысл
`re.ASCII`	По умолчанию `\w`, `\W`, `\b`, `\B`, `\d`, `\D`, `\s`, `\S` соответствуют все юникодные символы с соответствующим качеством. Например, `\d` соответствуют не только арабские цифры, но и вот такие: ٠١٢٣٤٥٦٧٨٩. `re.ASCII` ускоряет работу, если все соответствия лежат внутри ASCII.
`re.IGNORECASE`	Не различать заглавные и маленькие буквы. Работает медленнее, но иногда удобно
`re.MULTILINE`	Специальные символы `^` и `$` соответствуют началу и концу каждой строки
`re.DOTALL`	По умолчанию символ `\n` конца строки не подходит под точку. С этим флагом точка — вообще любой символ

Простые шаблоны, соответствующие позиции

Шаблон	Описание	Пример	Применяем к тексту
`^`	Начало всего текста или начало строчки текста, если `flag=re.MULTILINE`	`^Привет`	
`$`	Конец всего текста или конец строчки текста, если `flag=re.MULTILINE`	`Будь здоров!$`	
`\A`	Строго начало всего текста		
`\Z`	Строго конец всего текста		
`\b`	Начало или конец слова (слева пусто или не-буква, справа буква и наоборот)	`\bвал`	вал, перевал, Перевалка
`\B`	Не граница слова: либо и слева, и справа буквы, либо и слева, и справа НЕ буквы	`\Bвал`	перевал, вал, Перевалка
		`\Bвал\B`	перевал, вал, Перевалка

Шаблон	Применяем к тексту
`(?:\w\w\d\d)+`	Есть миг29а, ту154б. Некоторые делают даже миг29ту154ил86.
`(?:\w+\d+)+`	Есть миг29а, ту154б. Некоторые делают даже миг29ту154ил86.
`(?:\+7|8)(?:-\d{2,3}){4}`	+7-926-123-12-12, 8-926-123-12-12
`(?:[Xx][аоеи]+)+`	Муха — хахахехо, ну хаааховоохе, да хахахехохиииии! Хам трамвайный.
`\b(?:[Xx][аоеи]+)+\b`	Муха — хахахехо, ну хаааховоохе, да хахахехохиииии! Хам трамвайный.

Сложные шаблоны, соответствующие позиции (*lookaround* и Co)

Шаблон	Описание	Пример	Применяем к тексту
`(?=...)`	*lookahead assertion*, соответствует каждой позиции, сразу после которой начинается соответствие шаблону ...	`Isaac (?=Asimov)`	Isaac Asimov, Isaac other
`(?!...)`	*negative lookahead assertion*, соответствует каждой позиции, сразу после которой НЕ может начинаться шаблон ...	`Isaac (?!Asimov)`	Isaac Asimov, Isaac other
`(?<=...)`	*positive lookbehind assertion*, соответствует каждой позиции, которой может заканчиваться шаблон ... Длина шаблона должна быть фиксированной, то есть `abc` и `a|b` — это ОК, а `a*` и `a{2,3}` — нет.	`(?<=abc)def`	abcdef, bcdef
`(?<!...)`	*negative lookbehind assertion*, соответствует каждой позиции, которой НЕ может заканчиваться шаблон ...	`(?<!abc)def`	abcdef, bcdef

lookaround на примере королей и императоров Франции

`Людовик(?=VI)` — Людовик, за которым идёт VI
КарлIV, КарлIX, КарлV, КарлVI, КарлVII, КарлVIII,
ЛюдовикIX, ЛюдовикVI, ЛюдовикVII, ЛюдовикVIII, ЛюдовикX, ..., ЛюдовикXVIII,
ФилиппI, ФилиппII, ФилиппIII, ФилиппIV, ФилиппV, ФилиппVI

`Людовик(?!VI)` — Людовик, за которым идёт не VI

КарлIV, КарлIX, КарлV, КарлVI, КарлVII, КарлVIII,
ЛюдовикIX, ЛюдовикVI, ЛюдовикVII, ЛюдовикVIII, ЛюдовикX, ..., ЛюдовикXVIII,
ФилиппI, ФилиппII, ФилиппIII, ФилиппIV, ФилиппV, ФилиппVI

`(?<=Людовик)VI` — «шестой», но только если Людовик

КарлIV, КарлIX, КарлV, КарлVI, КарлVII, КарлVIII,
ЛюдовикIX, ЛюдовикVI, ЛюдовикVII, ЛюдовикVIII, ЛюдовикX, ..., ЛюдовикXVIII,
ФилиппI, ФилиппII, ФилиппIII, ФилиппIV, ФилиппV, ФилиппVI

`(?<!Людовик)VI` — «шестой», но только если не Людовик

КарлIV, КарлIX, КарлV, КарлVI, КарлVII, КарлVIII,
ЛюдовикIX, ЛюдовикVI, ЛюдовикVII, ЛюдовикVIII, ЛюдовикX, ..., ЛюдовикXVIII,
ФилиппI, ФилиппII, ФилиппIII, ФилиппIV, ФилиппV, ФилиппVI

Шаблон	Комментарий	Применяем к тексту
`(?<!\d)\d(?!\d)`	Цифра, окружённая не-цифрами	Text ABC 123 A1B2C3!
`(?<=#START#).*?(?=#END#)`	Текст от #START# до #END#	text from #START# till #END#
`\d+(?=_(?!_))`	Цифра, после которой идёт ровно одно подчёркивание	12_34__56
`^(?:(?!boo).)*?$`	Строка, в которой нет boo (то есть нет такого символа, перед которым есть boo)	a foo and boo and zoo and others

Шаблон	Комментарий	Применяем к тексту
`^(?:(?!boo)(?!foo).)*?$`	Строка, в которой нет ни boo, ни foo	a foo and boo and zoo and others