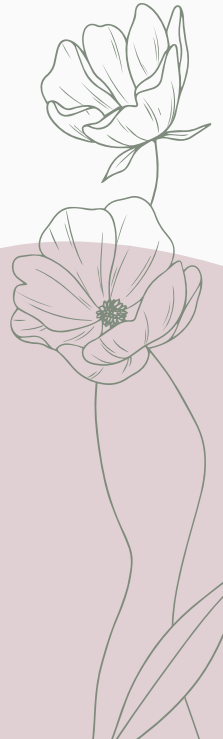
Программирование в лингвистике

Модуль re



Модуль re

re – предустановленный модуль, позволяющий работать с регулярными выражениями в Python. Его не нужно устанавливать, но нужно импортировать.

Re имеет 2 основных класса и 7 основных функций, с помощью которых можно составлять свои РВ, искать в тексте совпадения, заменять совпадения на другую строку, разбивать строку на части по совпадению и т.д.

```
import re

string = "123c3727a12t"
string = re.sub(r'\d+', '', string)
```

```
'cat'
```

re.Pattern

Pattern – это скомпилированное в экземпляр класса регулярное выражение. Регулярное выражение – строка Python, которая становится основным инструментом re после компиляции. Функции re сами компилируют строки в объект Pattern, но можно сделать это заранее:

```
pattern = re.compile(r'\d+')
```

Тогда функции из модуля re будут доступны в качестве методов Pattern:

```
new = pattern.sub('', string)
new = re.sub(r'\d+', '', string) # То же самое
```

Обратите внимание, что рекомендуется всегда ставить префикс **r (raw)** перед строкой с РВ, потому что там есть специальные последовательности, которые могут начать конфликтовать со стандартными последовательностями Python.

re.Match

Match – это экземпляр класса найденного результата.
Он содержит несколько вещей:

1. Само найденное выражение + отдельно содержимое групп РВ, если они там были.
2. Индексы найденного выражения в исходной строке.

Чтобы получить найденное выражение целиком, следует воспользоваться методом group:

```
m = pattern.search(string)
m.group()
```

Также можно получить содержимое каждой из групп, если передать в этот метод ее номер (счет с единицы).

re.Match

Также у Match есть три метода, позволяющих получить индексы:

Match.start() – возвращает индекс исходной строки, с которого вхождение начинается;
Match.end() – то же самое для конца вхождения (не включая правую границу, как в срезах);
Match.span() – возвращает кортеж из этих индексов.

```
string = "123c3727a12t"
pattern = re.compile(r'\d+')
m = pattern.search(string)
print(m.start(), m.end(), m.span())
```

```
0 3 (0, 3)
```

Функции re

Поиск	
re.search(pattern, text, flags)	Ищет до первого вхождения, полностью аналогично find, только с РВ.
re.findall(pattern, text, flags)	Ищет все вхождения. Возвращает списком строк. Если в РВ были группы, вернет список кортежей с содержимым групп.
re.finditer(pattern, text, flags)	То же, что findall, только возвращает итератор из объектов класса Match. Можно превратить в список, а можно сразу итерироваться.
Проверка	
re.match(pattern, text, flags)	Проверяет, что строчка text начинается с pattern. Возвращает Match.
re.fullmatch(pattern, text, flags)	Проверяет, что строчка text полностью подходит под pattern.
Замена	
re.sub(pattern, sub, text, flags)	Заменяет все вхождения pattern на строку sub (вместо строки можно передать функцию, которая принимает объект класса Match и возвращает строку).

Флаги

flags в каждой из этих функций - необязательный параметр.
Три полезных флага:

re.DOTALL	= re.S	Точка обозначает и \n тоже
re.MULTILINE	= re.M	^ и $ срабатывают после и до \n
re.IGNORECASE	= re.I	Игнорируется регистр

```
string = "cat\ndog\nhamster\n"
re.findall(r'(?m)^\w+', string)
```

```
['cat', 'dog', 'hamster']
```

В том числе они могут употреблятся внутри РВ: (?s), (?i), (?m)