Регулярные выражения

https://docs.python.org/3/library/index.html


s/^
^
*[
*(
)
$/
${1}
/mg

Назначение

Назначение: описание содержимого текстовой строки при помощи специальных правил

Синтаксис

1 способ

```
import  re
regex= re.compile
(“шаблон”)
```

2 способ

```
import  re
regex= re.compile
(r'шаблон')
```

Совпадение вначале -match

```
import   re
regex= re.compile ("privet")
str=input()
if (regex.match(str)):
        print(‘ДА’)
```

Совпадение вначале

privethtevirpghgh

Ghghghprivetghghghgh

Совпадение в любом месте строки -search

```
import   re
regex= re.compile ("privet")
str=input()
if (regex. search(str)):
        print(‘ДА’)
```

Совпадение в любом месте строки -search

Ghghghprivetghghghgh,
privethtevirpghgh

chfghprvjghkghkhk

[XYZ] - (или X, или Y, или Z)

```
import re
regex  =  re.compile ( "а[нкл]я" )
```

Ghghghаняghghghgh
спюрлоакяg576

Ghghghаннягhghghgh
спаярлоаакяg576

[^...] -любые символы, кроме

```
import re
regex  =  re.compile ( "[^bn]log" )
```

Flog,dlog,plog,
cfhglogcnvm

blog, cvncvnnlog.kjb

[...-...] - от… до

```
import re
regex = re.compile ( "[a-z]" )
```

a,b,c,d,….,x,y,z

A,B,C,D,….,X,Y,Z

[...-...] - от… до

```
import re
regex  =  re.compile ( "[a-z]kt" )
```

akt,bkt,ckt,…., kt,zkt

A,B,C,D,….,X,Y,Z

Инвертированные символьные классы

всё, кроме a, b, c:

```
[^abc]
```

^ как символ:

```
[abc^]
```

^ -начало

```
import  re
regex  =  re.compile ("^log")
```

loghtevirpghgh

fhkgjlog, dglogxgj

$ -конец

```
import re
regex = re.compile ("log$")
```

Dhdhlog**log**,virp**log**, **log**

fhkgj**log**5, dg**log**xgj

Полное совпадение строки -fullmatch

```
import   re
regex= re.compile ("privet")
str=input()
if (regex. fullmatch(str)):
        print(‘ДА’)
```

Полное совпадение

```
import  re
regex  =  re.compile ("^log$")
str=input()
if (regex. search(str)):
        print('ДА')
```

log

fhkgjlog, dglogxgj

Спецсимволы

регулярные выражения имеют спецсимволы, которые нужно экранировать.
Вот их список:

. ^ $ * + ? { } [ ] \ | ( )

\ -превращает специальный символ в обычный и наоборот

```
import re
regex = re.compile ( "rom\." )
```

rom.

rom\.

\ -превращает специальный символ в обычный и наоборот

```
import re
regex  =  re.compile ( "[xr]om\." )
```

```
import re
regex= re.compile ("^[^s]om\.txt")
```

. -любой символ, кроме перевода строки

```
import re
regex = re.compile ( "ro.ka" )
```

romka, ro\ka, ro.ka, выпгоыka

romk

. -любой символ, кроме перевода строки

```
import re
regex = re.compile ( "ro.ka$" )
```

```
import re
regex = re.compile (" \$ro.ka" )
```

* -повторение предыдущего символа ноль и более раз

```
import re
regex  =  re.compile ( "ro*ka" )
```

rooka, roooka,rka

* -повторение предыдущего символа ноль и более раз

```
import re
regex  =  re.compile (".*" )
```

«захватить всё»

```
import re
regex  =  re.compile ("\.*" )
```

+ -повторение предыдущего символа один и более раз

```
import re
regex  =  re.compile ("bo+" )
```

Xzfbo, bогпо

xcb, bzzzz

+ -повторение предыдущего символа один и более раз

```
import re
regex  =  re.compile ("^[ak]" )
```

Начинается с любого из указанных символов

```
import re
regex  =  re.compile ("[ak]+" )
```

Указанные символы встречаются один и более раз

? -повторение предыдущего символа ноль или один раз

```
import re
regex = re.compile (“^bo?$" )
```

b, bo

booooo

```
import   re
regex= re.compile ("colou?r")
str=“colour”
if (regex.search(str)):
        print(‘ДА’)
else:
    print(‘НЕТ’)
```

```
import   re
regex= re.compile ("colou?r")
str=“color”
if (regex.search(str)):
        print(‘ДА’)
else:
    print(‘НЕТ’)
```

Квантификатор определяет, сколько раз повторяться

? - 0 или 1 раз

* - 0 и более раз

+ - 1 и более раз

{2,5} - от 2 до 5 раз

{2,} - 2 и более раз

{5,} - от 0 до 5 раз

{2} - ровно 2 раза

Диапазоны повторов

`bo{7}`	точно 7
`bo{2,5}`	от 2 до 5, x < y
`bo{5,}`	5 или более
`b{,5}`	не работает!

| -ИЛИ

(^bo?)|(\.{5})$

….. ,bo, b

Буквы и цифры

\d ~ цифры от 0 до 9

\w ~ буквы, цифры и подчёркивание
работает для русских букв!

И наоборот:

\D ~ всё, кроме цифр

\W ~ всё, кроме букв и цифр

\s -пробельный символ

\S -непробельный символ

Использование ИТЕРАТОРА

re.finditer(“шаблон”,”строка”) – выдает все совпадения по шаблону

```
import  re
s=0
text=input()
regex=re.compile(“шаблон")
for i in re.finditer(regex,text):
    s+=1
print(s)
```

```
"^([01]?\\d\\d?|2[0-4]\\d|25[0-
5])\\.([01]?\\d\\d?|2[0-4]\\d|25
[0-5])\\.([01]?\\d\\d?|2[0-4]\\d
|25[0-5])\\.([01]?\\d\\d?|2[0-4]
\\d|25[0-5])$";
```