Python Programming

RGM College of Engineering & Technology (Autonomous)

Department of Computer Science & Engineering

Academic Year : 2020-2021

REGULAR EXPRESSIONS - 3

Guido Van Rossum

Learning Mantra

If you really strong in the basics, then remaining things will become so easy.

Agenda

1. Important functions of 're' module

6. Important functions of 're' module

1. match()
2. fullmatch()
3. search()
4. findall()
5. finditer()
6. sub()
7. subn()
8. split()
9. compile()

1. match():

- We can use match function to check the given pattern at beginning of target string or not.
- If the match is available then we will get Match object, otherwise we will get None.

Eg:

```
import re
s=input("Enter pattern to check: ")
m=re.match(s,"abcabdefg") # match() function
if m!= None:
    print("Match is available at the beginning of the String")
    print("Start Index:",m.start(), "and End Index:",m.end())
else:
    print("Match is not available at the beginning of the String")
```

```
Enter pattern to check: abc
Match is available at the beginning of the String
Start Index: 0 and End Index: 3
```

Eg:

```
import re
s=input("Enter pattern to check: ")
m=re.match(s,"abcabdefg")              # match() function
if m!= None:
    print("Match is available at the beginning of the String")
    print("Start Index:",m.start(), "and End Index:",m.end())
else:
    print("Match is not available at the beginning of the String")
```

```
Enter pattern to check: rgm
Match is not available at the beginning of the String
```

2. fullmatch():

- We can use fullmatch() function to match a pattern to all of target string. i.e., complete string should be matched according to given pattern.
- If complete string matched then this function returns Match object otherwise it returns None.

Eg:

```
import re
s=input("Enter pattern to check: ")
m=re.fullmatch(s,"abababab")
if m!= None:
    print("Full String Matched")
else:
    print("Full String not Matched")
```

```
Enter pattern to check: ab
Full String not Matched
```

Eg:

```
import re
s=input("Enter pattern to check: ")
m=re.fullmatch(s,"ababab")
if m!= None:
    print("Full String Matched")
else:
    print("Full String not Matched")
```

```
Enter pattern to check: abababa
Full String not Matched
```

Eg:

```
import re
s=input("Enter pattern to check: ")
m=re.fullmatch(s,"abababab")
if m!= None:
    print("Full String Matched")
else:
    print("Full String not Matched")
```

```
Enter pattern to check: abababab
Full String Matched
```

3. search():

- We can use search() function to search the given pattern in the target string.
- If the match is available then it returns the Match object which represents first occurrence of the match.
- If the match is not available then it returns None.

Eg:

```
import re
s=input("Enter pattern to check: ")
m=re.search(s,"abaabaaab")
if m!= None:
    print("Match is available")
    print("First Occurrence of match with start index:",m.start(),"and end index:",m.end())
else:
    print("Match is not available")
```

```
Enter pattern to check: aa
Match is available
First Occurrence of match with start index: 2 and end index: 4
```

Eg:

```
import re
s=input("Enter pattern to check: ")
m=re.search(s,"abaabaaab")
if m!= None:
    print("Match is available")
    print("First Occurrence of match with start index:",m.start(),"and end index:",m.end())
else:
    print("Match is not available")
```

```
Enter pattern to check: bb
Match is not available
```

4. findall():

- This function is used to find all occurrences of the match.
- This function returns a list object which contains all occurrences.

Eg:

```
import re
l=re.findall("[0-9]","a7b9c5kz")
print(l)
```

```
['7', '9', '5']
```

5. finditer():

- It returns the iterator yielding a match object for each match.
- On each match object we can call start(), end() and group() functions.

Eg:

```
import re
itr=re.finditer("[a-z]","a7b9c5k8z")
for m in itr:
    print(m.start(),"...",m.end(),"...",m.group())
```

```
0 ... 1 ... a
2 ... 3 ... b
4 ... 5 ... c
6 ... 7 ... k
8 ... 9 ... z
```

Eg:

```
import re
itr=re.finditer("\d","a7b9c5k8z")
for m in itr:
    print(type(m))
    print(m.start(),"...",m.end(),"...",m.group())
```

```
<class 're.Match'>
1 ... 2 ... 7
<class 're.Match'>
3 ... 4 ... 9
<class 're.Match'>
5 ... 6 ... 5
<class 're.Match'>
7 ... 8 ... 8
```

6. sub():

- sub means substitution or replacement.

re.sub (regex, replacement, target_string)

- In the target string every matched pattern will be replaced with provided replacement.

Eg:

```
import re
s=re.sub("[a-z]","#","a7b9c5k8z")
print(s)
```

```
#7#9#5#8#
```

Note: Here, Every alphabet symbol is replaced with # symbol.

Eg:

```
import re
s=re.sub("\d","#","a7b9c5k8z")
print(s)
```

```
a#b#c#k#z
```

Note: Here, Every digit is replaced with # symbol.

7. subn():

- It is exactly same as sub except it can also returns the number of replacements.
- This function returns a tuple where first element is result string and second element is number of replacements.

(result_string, number of replacements)

Eg:

```
import re
t=re.subn("[a-z]","#","a7b9c5k8z")
print(t)
print("The Result String:",t[0])
print("The number of replacements:",t[1])
```

```
('#7#9#5#8#', 5)
The Result String: #7#9#5#8#
The number of replacements: 5
```

8. split():

- If we want to split the given target string according to a particular pattern then we should go for split() function.
- This function returns list of all tokens.

Eg:

```
import re
l=re.split(",","sunny,bunny,chinny,vinny,pinny")
print(l)
for t in l:
    print(t)
```

```
['sunny', 'bunny', 'chinny', 'vinny', 'pinny']
sunny
bunny
chinny
vinny
pinny
```

Eg:

```
import re
l=re.split("\.","www.rgmcet.edu.in")
for t in l:
    print(t)
```

```
www
rgmcet
edu
in
```

Eg:

```
import re
l=re.split("[.]","www.rgmcet.edu.in")
for t in l:
    print(t)
```

```
www
rgmcet
edu
in
```

Two special symbols used in Regular Expressions

1. ^ symbol:

We can use ^ symbol to check whether the given target string starts with our provided pattern or not.

Eg:

```
res=re.search("^Learn",s)
```

If the target string starts with 'Learn' then it will return Match object, otherwise returns None.

Eg:

```
import re
s="Learning Python is Very Easy"
res=re.search("^Learn",s)
if res != None:
    print("Target String starts with Learn")
else:
    print("Target String Not starts with Learn")
```

```
Target String starts with Learn
```

Eg:

```
import re
s="Learning Python is Very Easy"
res=re.search("^Learns",s)
if res != None:
    print("Target String starts with Learn")
else:
    print("Target String Not starts with Learn")
```

```
Target String Not starts with Learn
```

2. '$' symbol:

We can use $ symbol to check whether the given target string ends with our provided pattern or not.

Eg:

```
res=re.search("Easy$",s)
```

If the target string ends with 'Easy' then it will return Match object, otherwise returns None.

Eg:

```
import re
s="Learning Python is Very Easy"
res=re.search("Easy$",s)
if res != None:
    print("Target String ends with Easy")
else:
    print("Target String Not ends with Easy")
```

```
Target String ends with Easy
```

Eg:

```
import re
s="Learning Python is Very Easy"
res=re.search("easy$",s)
if res != None:
    print("Target String ends with Easy")
else:
    print("Target String Not ends with Easy")
```

```
Target String Not ends with Easy
```

Note:

If we want to ignore case then we have to pass 3rd argument re.IGNORECASE for search() function.

Eg:

res = re.search("easy$",s,re.IGNORECASE)

```
import re
s="Learning Python is Very Easy"
res=re.search("easy$",s,re.IGNORECASE)
if res != None:
    print("Target String ends with Easy")
else:
    print("Target String Not ends with Easy")
```

```
Target String ends with Easy
```

Eg:

```
import re
s="Learning Python is Very Easy"
res=re.search("Easys$",s)
if res != None:
    print("Target String ends with Easy")
else:
    print("Target String Not ends with Easy")
```

```
Target String Not ends with Easy
```

Any question?

If you try to practice programs yourself, then you will learn many things automatically

Spend few minutes and then enjoy the study

Thank You