

Regular Expressions



Regular Expressions

Introduction

- **RE** is a string that contains special symbols and characters to find and extract the information
- Operations:
 - ✓ Search
 - ✓ Match
 - ✓ Find
 - ✓ Split
- Also called as *regex*
- Module: **re**
 - This module contains the methods like
 - `compile()`
 - `search()`
 - `match()`
 - `findall()`
 - `split()...`
- **import re**

Regular Expressions

Steps

- Step-1: Compile the RE

```
prog = re.compile(r'm\w\w')
```

- Step-2: Search the strings

```
str = "cat mat bat rat"  
result = prog.search(str)
```

- Step-3: Display the result

```
print(result.group())
```

Regular Expressions

Example-1: search()

```
import re
str = 'man sun mop run'
result = re.search(r'm\w\w', str)
if result: #if result is not None
    print(result.group())
```



```
import re
str = 'man sun mop run'
prog = re.compile(r'm\w\w')
result = prog.search(str)
if result: #if result is not None
    print(result.group())
```

search(): Combination of compile and run

- Point: Returns only the first string matching the RE

Regular Expressions

Example-2: findall()

```
import re
str = 'man sun mop run'
result = re.findall(r'm\w\w', str)
print(result)
```

findall()

- Returns all the matching strings
- Returns in the form of the list

Regular Expressions

Example-3: match()

```
import re
str = 'man sun mop run'
result = re.match(r'm\w\w', str)
print(result.group())
```

match()

- Returns the string only if it is found in the beginning of the string
- Returns None, if the string is not found

Regular Expressions

Example-4: match()

```
import re
str = 'sun man mop run'
result = re.match(r'm\w\w', str)
print(result)
```

`match()`

- Returns None, since the string is not found

Regular Expressions

Example-5: split()



```
import re
str = 'This; is the: "Core" Python\'s Lecturer'
result = re.split(r'\w+', str)
print(result)
```

- split() - splits the RE
 - W : Split at non-alphanumeric character
 - + : Match 1 or more occurrences of characters

split()

- splits the string into pieces according to the given RE

Regular Expressions

Example-6: Find & Replace: sub()

```
import re
str = 'Kumbhmela will be conducted at Ahmedabad in India.'
res = re.sub(r'Ahmedabad', 'Allahabad', str)
print(res)
```

Syntax:

```
sub(RE, new, old)
```

RE: Sequence Characters

RE: sequence characters

- Match only one character in the string

Character	Description
<code>\d</code>	Represents any digit (0 - 9)
<code>\D</code>	Represents any non-digit
<code>\s</code>	Represents white space Ex: <code>\t\n\r\f\v</code>
<code>\S</code>	Represents non-white space character
<code>\w</code>	Represents any alphanumeric (A-Z, a-z, 0-9)
<code>\W</code>	Represents non-alphanumeric
<code>\b</code>	Represents a space around words
<code>\A</code>	Matches only at start of the string
<code>\Z</code>	Matches only at end of the string

RE: sequence characters

Example-1:



To match all words starting with 'a'

```
import re
str = 'an apple a day keeps the doctor away'
result = re.findall(r'a[\w]*', str)

# findall() returns a list, retrieve the elements from list
for word in result:
    print(word)
```

To match all words starting with 'a', not sub-words then RE will look like this

```
import re
str = 'an apple a day keeps the doctor away'
result = re.findall(r'\ba[\w]*\b', str)

# findall() returns a list, retrieve the elements from list
for word in result:
    print(word)
```

* Matches with 0 or more occurrences of the character

RE: sequence characters

Example-2:



To match all words starting with numeric digits

```
import re
str = 'The meeting will be conducted on 1st and 21st of every month'
result = re.findall(r'\d[\w]*', str)
#for word in result:
print(word)
```

* Matches with 0 or more occurrences of the character

RE: sequence characters

Example-3:



To retrieve all words having 5 characters

```
import re  
  
str = 'one two three four five six seven 8 9 10'  
  
result = re.findall(r'\b\w{5}\b', str)  
  
print(result)
```

character	Description
\b	Matches only one space
\w	Matches any alpha numeric character
{5}	Repetition character

RE: sequence characters

Example-4: search()



To retrieve all words having 5 characters using search()

```
# search() will give the first matching word only.  
import re  
str = 'one two three four five six seven 8 9 10'  
result = re.search(r'\b\w{5}', str)
```

character	Description
\b	Matches only one space
\w	Matches any alpha numeric character
{5}	Repetition character

RE: sequence characters

Example-5: findall()



To retrieve all words having 4 and above characters using findall()

```
import re

str = 'one two three four five six seven 8 9 10'

result = re.findall(r'\b\w{4,}\b', str)

print(result)
```

character	Description
\b	Matches only one space
\w	Matches any alpha numeric character
{4, }	Retrieve 4 or more characters

RE: sequence characters

Example-6: findall()



To retrieve all words having 3, 4, 5 characters using findall()

```
import re

str = 'one two three four five six seven 8 9 10'

result = re.findall(r'\b\w{3, 5}\b', str)

print(result)
```

character	Description
\b	Matches only one space
\w	Matches any alpha numeric character
{3, 5}	Retrieve 3, 4, 5 characters

RE: sequence characters

Example-7: findall()



To retrieve only single digit using findall()

```
import re

str = 'one two three four five six seven 8 9 10'

result = re.findall(r'\b\d\b', str)

print(result)
```

character	Description
\b	Matches only one space
\d	Matches only digit

RE: sequence characters

Example-7: findall()



To retrieve all words starts with 't' from the end of the string

```
import re

str = 'one two three one two three'

result = re.findall(r't{\w}*\z', str)

print(result)
```

character	Description
\z	Matches from end of the string
\w	Matches any alpha numeric character
t	Starting character is 't'

RE: Quantifiers

RE: Quantifiers

- Characters which represents more than 1 character to be matched in the string

Character	Description
<code>+</code>	1 or more repetitions of the preceding regexp
<code>*</code>	0 or more repetitions of the preceding regexp
<code>?</code>	0 or 1 repetitions of the preceding regexp
<code>{m}</code>	Exactly m occurrences
<code>{m, n}</code>	From m to n. m defaults to 0 n defaults to infinity

RE: Quantifiers

Example-1:



To retrieve phone number of a person

```
import re  
str = 'Tomy: 9706612345'  
res = re.search(r'\d+', str)  
print(res.group())
```

character	Description
\d	Matches from any digit
+	1 or more repetitions of the preceding regexp

RE: Quantifiers

Example-2:



To retrieve only name

```
import re  
str = 'Tomy: 9706612345'  
res = re.search(r'\D+', str)  
print(res.group())
```

character	Description
\D	Matches from any non-digit
+	1 or more repetitions of the preceding regexp

RE: Quantifiers

Example-3:



To retrieve all words starting with “an” or “ak”

```
import re  
str = 'anil akhil anant arun arati arundhati abhijit ankur'  
res = re.findall(r'a[nk][\w]*', str)  
print(res)
```


RE: Quantifiers

Example-4:



To retrieve DoB from a string

```
import re
str = 'Vijay 20 1-5-2001, Rohit 21 22-10-1990, Sita 22 15-09-2000'
res = re.findall(r'\d{2}-\d{2}-\d{4}', str)
print(res)
```

RE	Description
<code>\d{2}-\d{2}-\d{4}</code>	Retrieves only numeric digits in the format of 2digits-2digits-4digits

RE: Special Character

RE: Special Characters

Character	Description
<code>\</code>	Escape special character nature
<code>.</code>	Matches any character except new line
<code>^</code>	Matches begining of the string
<code>\$</code>	Matches ending of a string
<code>[...]</code>	Denotes a set of possible characters Ex: <code>[6b-d]</code> matches any characters 6, b, c, d
<code>[^...]</code>	Matches every character except the ones inside brackets Ex: <code>[^a-c6]</code> matches any character except a, b, c or 6
<code>(...)</code>	Matches the RE inside the parentheses and the result can be captured
<code>R S</code>	matches either regex R or regex S

RE: Special Characters

Example-1:



To search whether a given string is starting with 'He' or not

```
import re  
  
str = "Hello World"  
  
res = re.search(r"^He", str)  
  
if res:  
    print("String starts with 'He'")  
else  
  
    print("String does not start with 'He'")
```

RE	Description
"^He"	Search from the beginning

RE: Special Characters

Example-2:



To search whether a given string is starting with 'He' or not from the end

```
import re
str = "Hello World"
res = re.search(r"World$", str)
if res:
    print("String ends with 'World'")
else
    print("String does not end with 'World'")
```

RE	Description
"World\$"	Search from the end

RE: Special Characters

Example-3:



To search whether a given string is starting with 'World' or not from the end by ignoring the case

```
import re

str = "Hello World"

res = re.search(r"world$", str, re.IGNORECASE)

if res:
    print("String ends with 'world'")
else:
    print("String does not end with 'world'")
```

RE	Description
"World\$"	Search from the end
re.IGNORECASE	Ignore the case

re.IGNORECASE

RE: Special Characters

Example-4:

To retrieve the timings am or pm

```
import re  
str = 'The meeting may be at 8am or 9am or 4pm or 5pm.'  
res = re.findall(r'\dam|\dpm', str)  
print(res)
```

RE: On Files

RE: On Files

Example-1:



To retrieve the emails from the file

```
import re

# open file for reading
f = open('mails.txt', 'r')

# repeat for each line of the file
for line in f:
    res = re.findall(r'\s+@\S+', line)

# display if there are some elements in result
if len(res)>0:
    print(res)

# close the file
f.close()
```

RE: On Files

Example-2:



To retrieve the data and write to another file

```
# Open the files
f1 = open('salaries.txt', 'r')
f2 = open('newfile.txt', 'w')

# repeat for each line of the file f1
for line in f1:
    res1 = re.search(r'\d{4}', line) # extract id no from f1
    res2 = re.search(r'\d{4},\.\d{2}', line) # extract salary from f1
    print(res1.group(), res2.group()) # display them
    f2.write(res1.group()+"\t") # write id no into f2
    f2.write(res2.group()+"\n") # write salary into f2

# close the files
f1.close()
f2.close()
```

RE: On HTML Files

RE: On HTML Files

Example-1:



To retrieve info from the HTML file

Step-1:

<code>import urllib.request</code>	Import this module
------------------------------------	--------------------

<code>f = urllib.request.urlopen(r'file:///path)</code>

Ex:

<code>f = urllib.request.urlopen(r'file:///~ Python\sample.html')</code>
--

<code>urllib.request</code>	Module name
-----------------------------	-------------

<code>urlopen</code>	To open the html files
----------------------	------------------------

<code>file:///</code>	Protocol to open the local files
-----------------------	----------------------------------

<code>~ Python\sample.html</code>	Under home DIR, under Python sub-DIR the sample.html file is present
-----------------------------------	--

RE: On HTML Files

Example-1:

Step-2: read and decode

<code>text = f.read()</code>	To read the file content
<code>str = text.decode()</code>	Since the HTML file contains the information in the byte strings

Step-3: Apply RE

```
r'<td>\w+</td>\s<td>(\w+)<td>\s<td>(\d\d.\d\d)<td>'
```

