Program Code: J620-002-4:2020

Program Name: FRONT-END SOFTWARE DEVELOPMENT ¶

Title : Regular Expressions - Part 1

Name: Ooi Caaron

IC Number: 990701-07-5837

Date : 3/7/23

Introduction : Lean to use Regular Expressions to get the data by using RE format

Conclusion : Still need to practice more

P12 - Regular Expressions

References:

https://docs.python.org/3/library/re.html (https://docs.python.org/3/library/re.html)

RegEx https://pypi.org/project/regex/ (https://pypi.org/project/regex/)

Before the start this lesson. Try to write a code to extract only the digits from this String. (Spend not more than 10 minutes on this)

str = "abc**00123**xyz**456_0**"

answer = ['00123', '456', '0']

In [17]:

```
# Answer
import re
str = "abc00123xyz456_0"
result = re.findall(r'[0-9]+',str)
result
```

Out[17]:

```
['00123', '456', '0']
```

What is a Regular Expression?

A regular expression in a programming language is a special text string used for describing a search pattern.

It is extremely useful for extracting information from text such as code, files, log, spreadsheets or even documents.

```
    \     Used to drop the special meaning of character following it

    \\    Represent a character class

    ^     Matches the beginning

    $     Matches the end

    .     Matches any character except newline

    ?     Matches zero or one occurrence.

    |     Means OR (Matches with any of the characters separated by it.

    *     Any number of occurrences (including 0 occurrences)

    +     One ore more occurrences

    {}    Indicate number of occurrences of a preceding RE to match.

    ()    Enclose a group of REs
```

Methods

re.match(), re.search() and re.findall() are methods of the Python module re.

The re.match() method

The re.match() method finds match if it occurs at start of the string. For example, calling match() on the string ‘FS Forward School FS’ and looking for a pattern ‘FS’ will match.

In [9]:

```python
import re

result = re.match(r'FS', 'FS Forward School FS')

print(result.group(0))
```

```
FS
```

The re.search() method

The re.search() method is similar to re.match() but it doesn’t limit us to find matches at the beginning of the string only.

In [11]:

```python
import re

result = re.search(r'FS', 'FS Forward School FS')

print(result)
print(result.group(0))
```

```
<re.Match object; span=(0, 2), match='FS'>
FS
```

The re.findall() method

The re.findall() helps to get a list of all matching patterns. It searches from start or end of the given string. If we use method findall to search for a pattern in a given string it will return all occurrences of the pattern. While searching a pattern, it is recommended to use re.findall() always, it works like re.search() and re.match() both.

In [7]:

```python
import re

result = re.findall(r'FS', 'FS Forward School FS')

print(result)
```

```
['FS', 'FS']
```

In [8]:

```
import re   # Need module 're' for regular expression

# Try find: re.findall(regexStr, inStr) -> matchedSubstringsList

# r'...' denotes raw strings which ignore escape code, i.e., r'\n' is '\'+'n'

re.findall(r'[0-9]+', 'abc123xyz')

# Return a list of matched substrings
```

Out[8]:

```
['123']
```

In [15]:

```
re.findall(r'[0-9]+', 'abcxyz')
[]

# where are the numbers?
```

Out[15]:

```
[]
```

In [16]:

```
import re

re.findall(r'[0-9]+', 'abc00123xyz456_0')
```

Out[16]:

```
['00123', '456', '0']
```

In [23]:

```
import re

re.findall(r'\d+', 'abc00123xyz456_0')
```

Out[23]:

```
['00123', '456', '0']
```

In [12]:

```
# Try substitute: re.sub(regexStr, replacementStr, inStr) -> outStr
import re

re.sub(r'[0-9]+', r'*', 'abc00123xyz456_0')
```

Out[12]:

```
'abc*xyz*_*'
```

In [19]:

```
# Try substitute with count: re.subn(regexStr, replacementStr, inStr) -> (outStr, count)
import re

re.subn(r'[0-9]+', r'*', 'abc00123xyz456_0')

# Return a tuple of output string and count
```

Out[19]:

```
('abc*xyz*_*', 3)
```

Texts

In [28]:

```
#  matching text that contain two keywords between characters.
import re

s1 = "I love Namewee"

regex=r"(I[a-zA-Z_0-9 ]+Namewee)"

match = re.findall(regex, s1)
print(match)
```

```
['I love Namewee']
```

In [29]:

```
#  matching text that contain two keywords between characters.
import re

s2 = "I hate Namewee"

regex=r"(I[a-zA-Z_0-9 ]*Namewee)"

match = re.findall(regex, s2)
print(match)
```

```
['I hate Namewee']
```

In [34]:

```
#  matching text that contain two keywords between characters.
import re

s3 = "I love Namewee, Namewee, I Dunno him, I Hate Namewee"

regex=r"(I[a-zA-Z_0-9 ]*Namewee)"

match = re.findall(regex, s3)
print(match)
```

```
['I love Namewee', 'I Hate Namewee']
```

In [38]:

```
#  matching text that contain two keywords between characters.
import re

s4 = "I heard that somebody said Namewee that he heard that A was informed that B though

regex=r"(I[a-zA-Z_0-9 ]*Namewee)"

match = re.findall(regex, s4)
print(match)
```

```
['I heard that somebody said Namewee that he heard that A was informed tha
t B thought that C misunderstood Namewee']
```

In [31]:

```
#  matching text that contain two keywords between characters.
import re

s5 = "I heard that somebody said Namewee, that he heard that A was informed that B thoug

regex=r"(I[a-zA-Z_0-9 ]*Namewee)"

match = re.findall(regex, s5)
print(match)

#observe the comma
```

```
['I heard that somebody said Namewee']
```

In [39]:

```
import re

s = "Loo Keen Ngin"

regex=r"(Forward School)"

match = re.findall(regex, s)

print(match)
```

```
[]
```

In [40]:

```
import re

s = "Loo Keen Ngin Forward School"

regex=r"(Forward School)"

match = re.findall(regex, s)

print(match)
```

```
['Forward School']
```

In [51]:

```
# re.match - If zero or more characters at the beginning of string match the regular exp
# return a corresponding match object. Return None if the string does not match the patt
# note that this is different from a zero-length match.

import re

s = "Loo Keen Ngin Forward School"

regex=r"(Forward School)"

match = re.match(regex, s)

print(match)
```

```
<re.Match object; span=(0, 4), match='Loo '>
```

In [50]:

```python
import re

s = "Loo Keen Ngin Forward School"

regex=r"(Loo Keen Ngin)"

match = re.match(regex, s)

print(match)
print(match.group(0))
```

```
<re.Match object; span=(0, 13), match='Loo Keen Ngin'>
Loo Keen Ngin
```

Here are a few more common ranges that you may want to match:

```
    000..255: ([01][0-9][0-9]|2[0-4][0-9]|25[0-5])
    0 or 000..255: ([01]?[0-9]?[0-9]|2[0-4][0-9]|25[0-5])
    0 or 000..127: (0?[0-9]?[0-9]|1[01][0-9]|12[0-7])
    0..999: ([0-9]|[1-9][0-9]|[1-9][0-9][0-9])
    000..999: [0-9]{3}
    0 or 000..999: [0-9]{1,3}
    1..999: ([1-9]|[1-9][0-9]|[1-9][0-9][0-9])
    001..999: (00[1-9]|0[1-9][0-9]|[1-9][0-9][0-9])
    1 or 001..999: (0{0,2}[1-9]|0?[1-9][0-9]|[1-9][0-9][0-9])
    0 or 00..59: [0-5]?[0-9]
    0 or 000..366: ([012]?[0-9]?[0-9]|3[0-5][0-9]|36[0-6])
```

** Try them out **

In [36]:

```python
#example

import re

s = "000..255"

regex=r"([01][0-9][0-9]|2[0-4][0-9]|25[0-5])"

match = re.findall(regex, s)

print(match)
```

```
['000', '255']
```

In [52]:

```python
# A Python program to demonstrate working of re.match().
import re

# Lets use a regular expression to match a date string
# in the form of Month name followed by day number
regex = r"([a-zA-Z]+) (\d+)"

match = re.search(regex, "I was born on June 24")

print(match)

if match != None:

    # We reach here when the expression "([a-zA-Z]+) (\d+)"
    # matches the date string.

    # This will print [14, 21), since it matches at index 14
    # and ends at 21.
    print("Match at index % s, % s" % (match.start(), match.end()))

    # We us group() method to get all the matches and
    # captured groups. The groups contain the matched values.
    # In particular:
    # match.group(0) always returns the fully matched string
    # match.group(1) match.group(2), ... return the capture
    # groups in order from left to right in the input string
    # match.group() is equivalent to match.group(0)

    # So this will print "June 24"
    print("Full match: % s" % (match.group(0)))

    # So this will print "June"
    print("Month: % s" % (match.group(1)))

    # So this will print "24"
    print("Day: % s" % (match.group(2)))

else:
    print("The regex pattern does not match.")
```

```
<re.Match object; span=(14, 21), match='June 24'>
Match at index 14, 21
Full match: June 24
Month: June
Day: 24
```

Resources

https://regex101.com/ (https://regex101.com/) https://www.debuggex.com/ (https://www.debuggex.com/)