Chapter - 6 : Functions

re.compiler

Compile a regular expression pattern into a regular expression object, which can be used for matching using its match(), search() and other methods.

Syntax

```python
sPattern = re.compile(pattern)
result = sPattern.match(string)
```

re.MULTILINE

- When specified, the pattern character '^' matches at the beginning of the string and at the beginning of each line (immediately following each newline); and
- The pattern character '$' matches at the end of the string and at the end of each line (immediately preceding each newline).
- By default, '^' matches only at the beginning of the string, and '$' only at the end of the string and immediately before the newline (if any) at the end of the string.

Example

```python
import re

x1 = "UttarPradesh Lucknow Gorakhpur
HimachalPradesh Shimla
Bihar Patna Buxar Muzaffarpur"

srcPtr = re.compile("\w+")

s1 = srcPtr.findall(x1)
s2 = srcPtr.findall(x1, re.MULTILINE)

print (s1)
p
```n

Output

```
['UttarPradesh']
['UttarPradesh', 'HimachalPradesh', 'Bihar']
```
Using \texttt{r} prefix

- When \texttt{r} or \texttt{R} prefix is used before a regular expression, it means raw string. For example, 'n' is a new line whereas \texttt{r"n"} means two characters: a backslash \ followed by n.
- Backslash \ is used to escape various characters including all metacharacters. However, using \texttt{r} prefix makes \ treat as a normal character.

\textbf{Example}

```python
import re
x1="Hello\nWorld"
print (x1)
x2=r"Hello\nWorld"
print (x2)
```

\textbf{Output}

```
Hello
World
Hello\nWorld
```

\textbf{Example}

```python
import re
string = '\n and \r are escape sequences.'
result = re.findall(r'\[\n\r\]', string)
print(result)
```

\textbf{Output}

```
['\n', '\r']
```

\textbf{re.sub()}

It will returns a string where matched occurrences are replaced with the content of replace variable.

\textbf{Syntax}

`re.sub(pattern, replace, string)`

\textbf{Example}

```python
# Program to remove all whitespaces
import re
# multiline string
string = 'Name Ajay Age 12\nHindi 23 \n English 6'
# matches all whitespace characters
pattern = 's+'
# empty string
replace = ''
new_string = re.sub(pattern, replace, string)
print(new_string)
```

\textbf{Output}

```
Name Ajay Age 12 Hindi 23 English 6
```
re.split()
It will splits the string depending on match and returns a list of strings where the splits have occurred.

**Syntax**  `split( pattern , string , count of split[optional])`

**Example-1**
```python
import re

string = 'Twelve:12 Eighty nine:89'
pattern = \"\d+\"

result = re.split(pattern, string)
print(result)
```

**Output**  
[
'Twelve:', ' Eighty nine:', '']

**Example-2**
```python
import re

string = 'Twelve:12 Eighty nine:89'
pattern = \"\d+\"

result = re.split(pattern, string,1)
print(result)
```

**Output**  
[
'Twelve:', ' Eighty nine:89']