WORKING WITH PYTHON REGULAR EXPRESSIONS
WHY REGULAR EXPRESSIONS?

```
#Example1
str1="My Name is Raju"
print(str1.replace("Raju","KSRaju"))

#Example2:
str2="Main Street is broad road"
print(str2.replace("road","rd"))
```

NOTE:
The above example converting all "road" patterns into rd, this is illegal, that time we are counting characters as follows...!!

```
#Example3: We can replace with the help of index
print(str2[0:17]+str2[17:].replace("road","rd"))
```

NOTE:
The above example display result as per programmer expectation, but counting characters every time is big challange, that time we are implementing PYTHON regexp or regex or re.

Define re?
Regular Expressions are powerful standardized way of searching, replacing, and parsing text with complex patterns of characters.

Syntax

```
import re
```

Regular Expression Patterns
You can escape a control character by preceding it with a backslash.

Pattern	Description
^	Matches beginning of line.
$	Matches end of line.
.	Matches any single character except newline.
re*	Matches 0 or more occurrences of preceding expression.
re+	Matches 1 or more occurrence of preceding expression.
re?	Matches 0 or 1 occurrence of preceding expression.

Special Character Classes

Example	Description
\d	Match a digit: [0-9]
\D	Match a nondigit: [^0-9]
\s	Match a whitespace character: [\t\r\n\f]
\S	Match nonwhitespace: [^ \t\r\n\f]
\w	Match a single word character: [A-Za-z0-9_]
\W	Match a nonword character: [^A-Za-z0-9_]

Literal characters

Example	Description
[Pp]ython	Match "Python" or "python"
rub[ye]	Match "ruby" or "rube"
[0-9]	Match any digit; same as[0123456789]
[^0-9]	Match anything other than a digit
[a-z]	Match any lowercase
[A-Z]	Match any uppercase

```
[a-zA-Z0-9]     Match any of the above

Repetition Cases
Example Description
ruby?   Match "rub" or "ruby": the y is optional
ruby*   Match "rub" plus 0 or more y's
ruby+   Match "rub" plus 1 or more y's
\d{3}   Match exactly 3 digits
\d{3,}  Match 3 or more digits
\d{3,5} Match 3, 4, or 5 digits

The most common uses of Regular Expressions are:
1 Search a string (match & search)
2 Finding a string (findall)
3 Break string into a sub strings (split)
4 Replace part of a string (sub)

Various methods of RE?
The 're' package provides multiple methods to perform queries on an
input string.
1 re.match()    2 re.search()
3 re.findall()  4 re.split()            5 re.sub()

The match Method
It finds and match, if pattern occurs at start of the string.

Syntax:
re.match(pattern, string)

Example:
import re
line="pet:cat I love cats"
mat=re.match(r"pet:\w\w\w",line)
print(mat)

NOTE:
It shows that pattern match has been found. To print the matching
string use method group, It helps to return the matching string.

import re
line="pet:cat I love cats"
mat=re.match(r"pet:\w\w\w",line)
print(mat.group(0))

Example:Using start and end methods
import re
line="Pet:Cat I like Pets Pet:Cow I love Cows"
mat=re.match(r"Pet:\w\w\w",line)
print(mat.group(0)) #Pet:Cat
print(mat.start())#0
print(mat.end())#7

NOTE: r always indicates PYTHON raw string..!!
NOTE: match method only matches the patterns in the starting..!!

Example:
from re import *
```

```
StrPatt = '^s...a$'
PyStr = 'subba'
PyResult = match(StrPatt, PyStr)
if(PyResult):
  print("Striing Pattern Matched")
else:
  print("String Pattern Not Matched")
```

The search Method
It searches for first occurrence of RE pattern within string.

Syntax:
```
re.search(pattern, string)
```

Example:
```
import re
line="pet:cat I love cats"
mat=re.search(r"pet:\w\w\w",line)
print(mat.group(0))
```

Example:
```
import re
line="I love cats pet:cat"
mat=re.match(r"pet:\w\w\w",line)
print(mat)
```

Example:
```
import re
line="I love cats pet:cat"
mat=re.search(r"pet:\w\w\w",line)
print(mat.group(0))
```

Example:
```
import re
line=" pet:cat I love cats pet:cow I love cow"
mat=re.search(r"pet:\w\w\w",line)
print(mat.group(0))
```

Example: For more than 1 time.
```
import re
PyResult = re.search('m*', 'Programming')
if(PyResult):
  print("Pattern Found")
else:
  print("Pattern Not Found")
```

Example:For Exactly 1 time.
```
from re import *
PyResult = search('!?', 'Hello World!')
if(PyResult):
  print("YES Found")
else:
  print("NOT Found")
```

NOTE:
search method returns only first occurance of the pattern in the string or line. If we need all matching patterns we should use

findall().

re.findall Method:
It helps to get a list of all matching patterns. It has no constraints of searching from start or end. If we will use method findall to search cat in given string it will return both occurrence of cow.

Syntax:
re.findall (pattern, string)

Example:
```
import re
line=" pet:cat I love cats pet:cow I love cow"
mat=re.findall(r"pet:\w\w\w",line)
print(mat)
```

Example:
```
import re
result=re.findall(r'@\w+.\w+','abc.test@gmail.com, xyz@test.in,
test.first@analyticsvidhya.com, first.test@rest.biz')
print(result)
```

Example:
```
import re
result=re.findall(r'@\w+.(\w+)','abc.test@gmail.com, xyz@test.in,
test.first@analyticsvidhya.com, first.test@rest.biz')
print(result)
```

re.split Method:
It helps to split string by the occurrences of given pattern.

Syntax:
re.split(pattern,string)

Example:
```
import re
line="I love cats pet:cat I love cows, pet:cow thank U"
mat=re.split(r"pet:\w\w\w",line)
print(mat)
```

Example:
```
import re
print(re.split(r'\s','Naresh i Technologies Ameerpet'))
print(re.split(r'\s','Leader in IT Training'))
```

Example:
```
import re
result=re.split(r' ','NareshIT HYDERABAD')
print(result)
```

Syntax2:
re.split(pattern, string, [maxsplit=0])

Example:
```
import re
result=re.split(r'a','NareshITHYDERABAD',maxsplit=1)
print(result)
```

re.sub():
It helps to search a pattern and replace with a new sub string. If the pattern is not found, string is returned unchanged.

Syntax:
re.sub(pattern, repl, string):

Example:
```
import re
str1="raju@abc.com and ksr@pqr.com and vara@vvv.in"
mat=re.sub(r"@\w+","@gmail",str1)
print(mat)
```

Example:
```
import re
text = "Python for beginners is a cool Scripting"
pattern = re.sub("cool", "good", text)
print(pattern)
```
****
Example:
```
import re
result = re.search(r'BigData', 'Data Science Based on BigData')
print (result.group(0))
#group(0) didn't return the entire match
```

Example:
```
import re
fi = open('Hai.txt')
for line in fi:
       if re.search('^F', line) :
         print(line)
```

rstrip()
It returns a copy of the string in which all chars have been stripped from the end of the string.

Syntax
str.rstrip([chars])

Example:
```
import re
fi = open('Hai.txt')
for line in fi:
       if re.search('From:', line) :
         line = line.rstrip()
         print(line)
```

Search and Replace
One of the most important re methods that use regular expressions is sub and search.

Syntax
re.search(pattern, repl)

Example:

```
import re
s = 'NareshITOn Ameerpet MAIN ROAD..!!'
Rep=s.replace('NareshIT', 'NiT:')
print(Rep)
```

re.findall Method:
It helps to get a list of all matching patterns. It has no constraints of searching from start or end. If we will use method findall to search NiT in given string it will return both occurrence of nit.

Syntax:
```
re.findall (pattern, string)
```

Example: Extract each character (using "\w")
```
import re
result=re.findall(r'.','Naresh i Technologies')
print(result)
result1=re.findall(r'\w','Naresh i Technologies')
print(result1)
```

Extract each word (using "*" or "+")
Example:
```
import re
result=re.findall(r'\w*','Naresh i Technologies Leader in IT
Training')
print (result)
```

Example:
```
import re
result=re.findall(r'\w+','Naresh i Technologies Leader in IT
Training')
print (result)
```

Extract each word (using "^")
Example:
```
import re
result=re.findall(r'^\w+','NiT  Naresh i Technologies')
print (result)
```

Example:
```
import re
result=re.findall(r'\w+$','NiT  Naresh i Technologies')
print (result)
```

Return the first three character of each word (\w)
Example:
```
import re
result=re.findall(r'\w\w\w','NiT Naresh i Technologies')
print (result)
```

Extract consecutive two characters those available at start of word boundary (using "\b")
Example:
```
import re
result=re.findall(r'\b\w.','NiT Naresh i Technologies Leader in IT')
print (result)
```

Return the domain type of given email-ids
Extract all characters after “@”

Example:
```
import re
result=re.findall(r'@\w+',"""raju.ksr@gmail.com, xyz@test.in,
test.first@nareshit.com, first.test@nit.biz""")
print (result)
```

NOTE: Above, you can see that ".com", ".in" part is not extracted.

Example:
```
import re
result=re.findall(r'@\w+.\w+',"""raju.ksr@gmail.com, xyz@test.in,
test.first@nareshit.com, first.test@nit.biz""")
print (result)
```

Example: Extract only domain name using "( )"
```
import re
result=re.findall(r'@\w+.(\w+)',"""raju.ksr@gmail.com, xyz@test.in,
test.first@nareshit.com, first.test@nit.biz""")
print (result)
```

Return date from given string: Here we will use "\d" to extract digit.
Example:
```
import re
result=re.findall(r'\d{2}-\d{2}-\d{4}',"""Amit 34-3456 12-05-2007,
XYZ 56-4532 11-11-2011, ABC 67-8945 12-01-2009""")
print (result)
```

NOTE: If you want to extract only year again parenthesis "( )" will help you.

Example:
```
import re
result=re.findall(r'\d{2}-\d{2}-(\d{4})',"""Amit 34-3456 12-05-2007,
XYZ 56-4532 11-11-2011, ABC 67-8945 12-01-2009""")
print (result)
```

re.split Method:
"s": This expression is used for creating a space in the string

Example:
```
import re
print(re.split(r'\s','Naresh i Technologies Ameerpet'))
print(re.split(r'\s','Leader in IT Training'))
```

Example: Split a string with multiple delimiters
```
import re
line = 'asdf fjdk;afed,fjek,asdf,foo'
result= re.split(r'[;,\s]', line)
print(result)
```

NOTE: We can also use method re.sub() to replace these multiple delimiters with one as space.

re.sub(pattern, repl, string):

It helps to search a pattern and replace with a new sub string. If the pattern is not found, string is returned unchanged.

Example:

```
import re
result=re.sub(r'India','the World','NiT is largest Training Center of India')
print(result)
```