REGEX

REGEX is used tofind a pattern or a string in the text

[abc]->a,b or c

[a-z] -> a to z

[A-Z] -> A to Z

[^abcd] -> all characters except a,b,c,d

[a-z A-Z] -> a toz, A to Z

[0-9] -> 0 to 9

Below are the QUANTIFIERS:

[ ]? -> Occurs 0 or one time

[ ]+ -> Occurs one or more time

[ ]* -> Occurs 0 or more time

[ ] {n} -> Occurs n time

[ ] {n,} -> Occurs n time or more than n

[ ] {x,y} -> Occurs atleast x times, but less than y times

REGEX METACHARACTERS:

\d -> [0-9]

\D -> [^0-9]

\w -> [a-z A-Z 0-9]

\W -> [^\w]

\ is called as escape character

. represents any character other than newline.

r stands for Raw String

In [ ]:

In [8]:
```
import re
txt = "It is better to fail in originality than to succeed in imitation."
x = re.findall("[a-e]", txt)
print(x)
```

```
['b', 'e', 'e', 'a', 'a', 'a', 'c', 'c', 'e', 'e', 'd', 'a']
```

In [12]:
```
import re
txt = "It is better to fail in originality than to succeed in imitation."
x = re.findall("[a-e][k-t]", txt)
print(x)
```

```
['et', 'er', 'al', 'an', 'at']
```

In [13]:
```
import re
txt = "It is better to fail in originality than to succeed in imitation."
x = re.findall("[a-e][k-t]*", txt)
print(x)
```

```
['b', 'ett', 'er', 'a', 'al', 'an', 'c', 'c', 'e', 'e', 'd', 'at']
```

In [29]:
```
# Searching the 10 digit mobile number
import re
txt = "Call me on 9876543211, if I my number: 8977553122 is not reachable"
x = re.findall("[8 9][0-9]{10}", txt)
print(x)
```

```
[' 9876543211', ' 8977553122']
```

In [28]:
```
#Searching a pattern where first letter is Capital with containing one number
import re
txt = "Call me on Ad6asd if I my number: Gdsdas8e is not D4 reachable"
x = re.findall("[A-Z][a-z]*[0-9][a-z]+", txt)
print(x)
```

```
['Ad6asd', 'Gdsdas8e']
```

In [27]:
```
#Searching a pattern where first letter is Capital with containing one number
import re
txt = "Call me on Ad6asd if I my number: Gdsdas8e is not D4 reachable"
x = re.findall("[A-Z][a-z]*[0-9][a-z]*", txt)
print(x)
```

```
['Ad6asd', 'Gdsdas8e', 'D4']
```

In [56]:
```
# Finding email pattern in a string
import re
txt = "mail me at  shahrukh@gmail.com and keep CC to julie@yahoo.initrrr as we
txt1 = "shahrukh@gmail.com and keep CC to julie@yahoo.initrrr as well."
x = re.findall("[a-z,A-Z,0-9,\-,\.,_]*[@][a-z]+[\.][a-z]{2,3}", txt)
c = re.findall("\w+@\w+.\w{2,4}\W+", txt)

y=re.search("\w+@\w+.\w+",txt)
r=re.match(r"\w+@\w+.\w+",txt1)
s=re.match(r"\w+@\w+.\w+",txt)
print(x)
print(c,"\n-----------------------------------------------")
print(y)
print(r)
print(s)
```

```
['shahrukh@gmail.com', 'julie@yahoo.ini']
['shahrukh@gmail.com ', 'julie@yahoo.']
-----------------------------------------------
<re.Match object; span=(12, 30), match='shahrukh@gmail.com'>
<re.Match object; span=(0, 18), match='shahrukh@gmail.com'>
None
```

match() funcction -> This function only checks for a match at the beginning of the string.

search() -> looks for occurrences of the regex pattern inside the entire target string

and returns the corresponding Match Object instance where the first match is found.

SPLIT Function

In [57]:
```
text="Nothing is impossible. The word itself says, 'I'm possible!"

a=re.split("[\s]",text)
print(a)
```

```
['Nothing', 'is', 'impossible.', 'The', 'word', 'itself', 'says,', "'I'm", 'p
ossible!']
```

In [58]:
```
text="Nothing is impossible. The word itself says, 'I'm possible!"

a=re.split("[\.\s]",text)
print(a)
```

```
['Nothing', 'is', 'impossible', '', 'The', 'word', 'itself', 'says,', "'I'm",
'possible!']
```

In [59]:
```
text="Nothing is impossible. The word itself says, 'I'm possible!"

a=re.split("[\.]",text)
print(a)
```

```
['Nothing is impossible', " The word itself says, 'I'm possible!"]
```

In [62]:
```
text="Nothing is impossible. The word itself says, 'I'm possible!"

a=re.split("[\s']",text)
print(a)
```

```
['Nothing', 'is', 'impossible.', 'The', 'word', 'itself', 'says,', '', 'I',
'm', 'possible!']
```

sub function

In [67]:
```
para="""There are a great many million fish in the sea,
but this story is about just one of them and a very small one at that.
Now, this little fish had everything in the sea to make him contended,
but he was not happy. You will laugh when I tell you why he was not.
He was unhappy because he was very small."""

sub_he= re.sub("he","HE", para)
print(sub_he)
```

```
THEre are a great many million fish in tHE sea,
but this story is about just one of tHEm and a very small one at that.
Now, this little fish had everything in tHE sea to make him contended,
but HE was not happy. You will laugh wHEn I tell you why HE was not.
He was unhappy because HE was very small.
```

In [71]:
```
para="""There are a great many million fish in the sea,
but this story is about just one of them and a very small one at that.
Now, this little fish had everything in the sea to make him contended,
but he was not happy. You will laugh when I tell you why he was not.
He was unhappy because he was very small."""

sub_he= re.sub("\she\s"," HE ", para)
print(sub_he)
```

```
There are a great many million fish in the sea,
but this story is about just one of them and a very small one at that.
Now, this little fish had everything in the sea to make him contended,
but HE was not happy. You will laugh when I tell you why HE was not.
He was unhappy because HE was very small.
```

In [96]:
```
para="""There are a great many million fish in the sea, w@y
but this story is about just one of them and a very small one@two at that.
Now, this little fish had everything in the sea to make him contended,
but he was not happy. You will laugh when I tell you why he ha was not.
He was unhappy because he was very small."""

sub_he= re.search("(\s\w+)@(\w+)", para)
print(sub_he)
print(type(sub_he))
sub_he.group(1),sub_he.group(2)
```

```
<re.Match object; span=(47, 51), match=' w@y'>
<class 're.Match'>
```

Out[96]:
```
(' w', 'y')
```

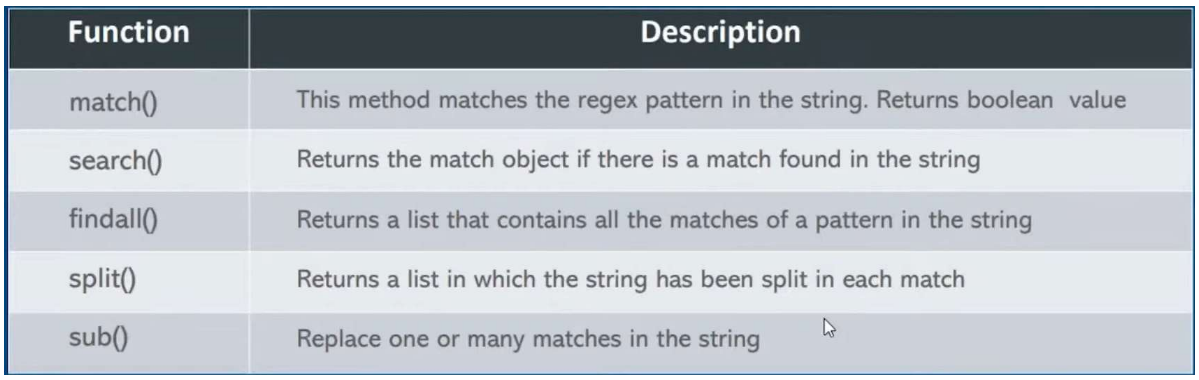
In [94]:
```
para="""There are a great many million fish in the sea, w@y
but this story is about just one of them and a very small one@two at that.
Now, this little fish had everything in the sea to make him contended,
but he was not happy. You will laugh when I tell you why he ha was not.
He was unhappy because he was very small."""

sub_he= re.findall("(\s\w+)@(\w+)", para)
print(sub_he)
print(type(sub_he))
sub_he.group(1),sub_he.grou
```

```
[(' w', 'y'), (' one', 'two')]
<class 'list'>

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[94], line 10
      8 print(sub_he)
      9 print(type(sub_he))
---> 10 sub_he.group(1),sub_he.grou

AttributeError: 'list' object has no attribute 'group'
```



Function	Description
match()	This method matches the regex pattern in the string. Returns boolean value
search()	Returns the match object if there is a match found in the string
findall()	Returns a list that contains all the matches of a pattern in the string
split()	Returns a list in which the string has been split in each match
sub()	Replace one or many matches in the string

```
In [97]: message="Hi Julie, call me on 9876542222 while I am in office, else whatsapp m

         re.search("\d{10}",message)

Out[97]: <re.Match object; span=(21, 31), match='9876542222'>
```

```
In [98]: message="Julie! Why did you called me on 9876-542-222 while I was at Home, I h

         re.search("\d{4}-\d{3}-\d{3}",message)

Out[98]: <re.Match object; span=(35, 47), match='9876-542-222'>
```

```
In [99]: message="Julie! Why did you called me on 9876-542-222 while I was at Home, I h

         re.findall("\d{4}-\d{3}-\d{3}",message)

Out[99]: ['9876-542-222', '9962-222-123']
```