REGULAR EXPRESSIONS:

STEP-1: Create pattern object.

compile() - Convert the pattern into regex object.

I/p - import re
Pattern = re.compile('Python')
Print(type(Pattern))

O/p - <class 're.Pattern'>

STEP-2: Create Matcher Object.

finditer() - we can check how many characters are available.

I/p - matcher = pattern.finditer('I Python am learning python Regex in Python!')
Print(type(matcher))

O/p - <class 'callable_iterator>

STEP-3 : Iterate over the Matcher

start() - starting index of matched string.

end() - end+1 index

group() - returns matched string.

```
i/p- for m in matcher:
        Print(type(m))
        Print('Match is at:{}, End:{},
              Pattern found:{}'.format(m.start(),
              m.end(), m.group()))
  Print('DONE')
```

```
O/P - <class 're.Match'>
      Match is at:2, End: 8, Pattern found:Python
      <class 're.Match'>
      Match is at : 37, End : 43, Pattern found:Python
      DONE
```

USING VARIOUS COMBINATIONS:

```
I/p - matcher = re.finditer('ab','abaababa')
    count = 0
    for m in matcher:
        count += 1
        Print('Match is at: {}, End: {}, Pattern
                found : {}'.format(m.start(),
                m.end(), m.group()))
    Print('Total count : ', count)
```

```
O/p - Match is at: 0, End: 2, Pattern found : ab
      Match is at 3, End : 5, Pattern found : ab
      Match is at 5, End: 7, Pattern found : ab
      Total count : 3
```

```
I/p - matcher = re.finditer('ba','abaababa')
    count = 0
    for m in matcher:
        count += 1
        Print('Match is at : {}, End : {}, Pattern
                found : {}'.format(m.start(), m.end
                (), m.group()))
```

```
Print('Total count: ', count)
```

O/p - Match is at : 1, End: 3, Pattern found : ba

Match is at : 4, End: 6, Pattern found : ba

Match is at : 6, End: 8, Pattern found : ba

Total count : 3

I/p -

```
matcher = re.finditer('[abc]', 'a7b@K9z')
count = 0
for m in matcher:
    count += 1
    Print('Match is at: {}, End: {}, Pattern
        found: {}'.format(m.start(),
        m.end(), m.group()))
Print('Total count: ', count)
```

O/p - Match is at : 0, End: 1, Pattern found : a

Match is at : 2, End: 3, Pattern found : b

Total count : 2

```
I/p - matcher = re.finditer('[a-z]', 'a7b@ K9z')
      count = 0
      for m in matcher:
          count += 1
          Print('Match is at : {}, End : {}, Pattern
          found : {}'.format(m.start(), m.end(),
          m.group()))
      Print('Total count : ', count)
```

```
O/p - Match is at : 0, End : 1, Pattern found : a
      Match is at : 2, End : 3, Pattern found : b
      Match is at : 5, End : 6, Pattern found : K
      Match is at : 7, End : 8, Pattern found : z
      Total count : 4
```

```
I/p - matcher = re.finditer('[^a-z]', 'a7b@ K9z')
      count = 0
      for m in matcher :
          count += 1
          Print('Match is at : {}, End : {}, Pattern
          found : {}'.format(m.start(), m.end(),
          m.group()))
      Print('Total count : ', count)
```

O/p - Match is at : 1, End: 2, Pattern found : 7

Match is at: 3, End: 4, Pattern found : @

Match is at: 4, End: 5, Pattern found :

Match is at: 6, End: 7, Pattern found : 9

Total count : 4

I/p - matcher = re.finditer('\$', 'a7b@ k9z')

```
count = 0
for m in matcher:
    count += 1
    Print('Match is at :{}, End:{}, Pattern
    found : {}', .format(m.start(), m.end(),
    m.group()))
Print('Total count :', count)
```

O/p - Match is ~~for~~ at 4, End : 5, Pattern found :

Total count : 1

```
I/p - matcher = re.finditer('\w', 'a7b@ k9z')
    count = 0
    for m in matcher:
        count += 1
        Print('Match is at : {}, End: {}, Pattern
        found : {}'.format(m.start(), m.end(),
        m.group()))
    Print('Total count : ', count)
```

```
O/p - Match is at : 0, End : 1, Pattern found : a
    Match is at : 1, End : 2, Pattern found : 7
    Match is at : 2, End : 3, Pattern found : b
    Match is at : 5, End : 6, Pattern found : k
    Match is at : 6, End : 7, Pattern found : 9
    Match is at : 7, End : 8, Pattern found : z
    Total count : 6
```

```
I/p - matcher = re.finditer('\W', 'a7b@ k9z')
    count = 0
    for m in matcher :
        count += 1
        Print('Match is at : {}, End : {}, Pattern
        found : {}'.format(m.start(), m.end()
```

```
m.group()))
Print('Total count : ', count)
```

o/p - Match is at : 3, End : 4, Pattern found : @

Match is at : 4, End : 5, Pattern found :

Total count : 2

i/p -

```
matcher = re.finditer('^a', 'a7b@ K9z')
count = 0
for m in matcher:
    count += 1
    Print('Match is at : {}, End : {}, Pattern
    found : {}', format(m.start(), m.end(),
    m.group()))
Print('Total count : ', count)
```

o/p - Match is at : 0, End : 1, Pattern found : a

Total count : 1

i/p -

```
matcher = re.finditer('z$', 'a7b@ K9z')
count = 0
for m in matcher:
    count += 1
    Print('Match is at : {}, End : {}, Pattern
```

```
        found : {}'.format(m.start(), m.end()
        m.group()))
    Print('Total count : ', count)
```

o/p - Match is at : 7, End : 8, Pattern found : z

Total count : 1

```
i/p - matcher = re.finditer('a','abaababa')
    count = 0
    for m in matcher:
        Count += 1
        Print('Match is at : {}, End : {}, Pattern
            found : {}'.format(m.start(), m.end()
            m.group()))
    Print('Total count : ', count)
```

o/p - Match is at : 0, End : 1 , Pattern found : a

Match is at : 2, End : 3 , Pattern found : a

Match is at : 3, End : 4 , Pattern found : a

Match is at : 5, End : 6 , Pattern found : a

Match is at : 7, End : 8 , Pattern found : a

Total count : 5

```
i/p - matcher = re.finditer('a+','abaababa')
      for m in matcher:
          Print('Match is at : {}, End:{}, Pattern
            found : {}'.format(m.start(), m.end(),
            m.group()))
```

o/p - Match is at:0, End: 1, Pattern found: a
Match is at: 2, End: 4, Pattern found: aa
Match is at: 5, End: 6, Pattern found: a
Match is at: 7, End: 8, Pattern found: a

```
i/p - matcher = re.finditer('a*','abaababa')
      for m in matcher:
          Print('Match is at : {}, End:{}, Pattern
            found : {}'.format(m.start(), m.end(),
            m.group()))
```

o/p - Match is at : 0, End : 1, Pattern found : a
Match is at : 1, End : 1, Pattern found :
Match is at : 2, End : 4, Pattern found : aa
Match is at : 4, End : 4, Pattern found :
Match is at : 5, End : 6, Pattern found : a
Match is at : 6, End : 6, Pattern found :
Match is at : 7, End : 8, Pattern found : a

Match is at: 8, End: 8, Pattern found:

```
i/p - matcher = re.finditer('a?', 'abababa')
for m in matcher:
    Print('Match is at: {}, End: {}, Pattern
    found: {}'.format(m.start(), m.end(),
    m.group()))
```

```
o/p - Match is at: 0, End: 1, Pattern found: a
Match is at: 1, End: 1, Pattern found:
Match is at: 2, End: 3, Pattern found: a
Match is at: 3, End: 4, Pattern found: a
Match is at: 4, End: 4, Pattern found:
Match is at: 5, End: 6, Pattern found: a
Match is at: 6, End: 6, Pattern found:
Match is at: 7, End: 8, Pattern found: a
Match is at: 8, End: 8, Pattern found:
```

↳ findall() - This function returns a list containing all matches.

↳ search() - this function searches the string for a match and returns a Match object if there is a match.

NOTE: If no matches are found, the value NONE is ~~RET~~ returned.

↳ split() - This function returns a list where the string has been split at each match.

↳ sub() - This function replaces the matches with the text of our choice.

↳ Match Object: A Match object is an object containing information about the search and the result.

NOTE: If there is no match, the value NONE will be returned, insted of Match Object

The Match Object has properties and method used to retrieve information about the search and the result:

↳ .span() - returns a tuple containing the start-, and end positions of the match.

↳ .string - returns the string passed into the function.

↳ .group() - returns the part of the string where there was a match.

```
I/P - s = input('Enter pattern: ')
      m = re.search(s, 'abaabaaab')
      if m != None:
          Print('Match available')
          Print('First occurance is at: {}, End: {
          Pattern found: {}'.format(m.start(),
          m.end(), m.group()))
      else:
          Print('Match is not available in the whole string
```

O/p - Enter pattern : ba

Match available.

First occurance is at 1, End : 3, Pattern found : ba

I/p - lst = re.findall('[0-9]', '0@KM29-1')

Print(lst)

O/p - ['0', '2', '9', '1']

I/p - line = 'cats are smarter than dogs'

```
m = re.match('dogs', line)

if m:
    Print('Match found at starting')

else :
    Print('No match')

s = re.search('dogs', line)

if s:
    Print('search successful')

else:
    ~~Noth~~
    Print('Nothing found')
```

O/p - No match

Search successful.

re.sub(regex, replacement, targetstring)

I/p - s = re.sub('\d', '#', 'a7b6j9k2h4')

Print (s)

O/p - a#b#j#k#h#

I/p - l = re.split('-', 'I-learn-Python')

Print (l)

O/p - ['I', 'learn', 'Python']

I/p - l = re.split('\W', 'abcdefg@gmail.com')

Print (l)

O/p - ['abcdefg', 'gmail', 'com']

I/p - s = 'I am learning Python'

m = re.~~match~~ search('Python$', s)

if m:

Print('cool!')

else:

Print('Not cool!')

O/p - cool!

I/p - s = input('Enter any identifier to validate : ')

m = re.fullmatch('[a-zA-Z_]+[a-zA-Z0-9]*', s)

if m:
    Print(s, 'is valid')

else:
    Print(s, 'not valid')

O/p - Enter any identifier to validate: 2two

2two not valid.

I/p - s = input('Enter a mobile number to validate : ')

m = re.fullmatch('[6-9]\d{9}', s)

if m:
    Print(s, 'is valid')

else:
    Print(s, 'not valid')

O/p - Enter a mobile number to validate : 123

123 not valid.

```
I/p - line = 'Cats are smarter than dogs'
      m = re.match('(.*) are (.*).*', line)
      if m:
          Print('Match found: ', m.group())
      else:
          Print('NO match')
```

O/p - Match found: Cats are smarter than dogs.

The Ways to write RE:

WAY-1:

```
I/p - pattern = re.compile('ab')
      matcher = pattern.finditer('abaababa')
      for m in matcher:
          Print(m.group(), m.start(), m.end())
```

O/p - ab 0 2
ab 3 5
ab 5 7

WAY-2: From WAY-1- adding step-1 directly to step-2

```
I/P - matcher = re.compile('ab').finditer
                                ('abaababa')

      for m in matcher:
          Print(m.group(), m.start(), m.end())
```

```
O/P -   ab  0  2
        ab  3  5
        ab  5  7
```

WAY-3: re.finditer(pattern, target)

```
I/P - matcher = re.finditer('ab', 'abaababa')

      for m in matcher:
          Print(m.group(), m.start(), m.end())
```

```
O/P -   ab  0  2
        ab  3  5
        ab  5  7
```