In [ ]:
```
#Agenda of Today:
                            1. Regular Expressions
                            2. Functional Programming
```

In [ ]:
```
#Regular Expressions?
    #Def: Its a sequence of characters that defines a search pattern.
     -Its also called as RegEx or re module (Its a predefined module)
  #what is the use of re?
     - it can be used to search, edit and manipulate the text/string.
#Note:
    Search pattern can be formed along with some rules:
        those rules including
         (i) Meta Characters   -        [],.,^,$,?,{},(),\,|
         (ii) Special Sequences  - \A,\a,\B,\b, \s,\S,\d, \D
        (iii) Sets  - [a-z],   {0-9} etc.
```

In [ ]:
```
#Re module - Functions:
1. match()
2. search()
3. findall()
4. split()
5. sub()
```

In [ ]:
```
#match():
this function searches the pattern only at the begining of the string and returns a first occurence only.
and if match is not found then its returns NONE object.

#syntax:
    match(pattern,string)
```

In [8]:
```
#Example 1:
import re
string = "today is Saturday"
pattern = "t"
print(re.match(pattern,string))
```

```
<re.Match object; span=(0, 1), match='t'>
```

In [16]:
```
#Example 2:
import re
string = "Python if Fun"
pattern = "\APython"       #Returns a match if the specified characters are at the beginning of the string
match = re.match(pattern,string)
print(match)
```

```
<re.Match object; span=(0, 6), match='Python'>
```

In [20]:
```
#Search():
#its searches the entire lines of string and returns its first occurence only
#syntax: re.search(pattern,string)
import re
print(re.search("c","apssdc in vijaywada city"))
```

```
<re.Match object; span=(5, 6), match='c'>
```

In [25]:
```
#example 2:
import re
names = ["surya","sun","srinivas","ramya","karthik","zakir","yashu","vamsi"]
for name in names:
    if re.search("s",name):
        print(name)
```

```
surya
sun
srinivas
yashu
vamsi
```

In [33]:
```
#findall(): re.findall(pattern,string)
#It returns the a list of strings containing all matches in a given string.
#To extract the numbers of from combo string.
import re
string = "hello 3278383 good evening 8593373,.How r u Mr.503"
pattern = "\d+"  #Returns a match where the string contains digits (numbers from 0-9)
output = re.findall(pattern,string)
output
```

Out[33]:
```
['hello ', ' good evening ', ',.How r u Mr.']
```

In [40]:
```
#Split():
 #Syntax:     split(pattern,string,maxlimit)

    #Def: Its Spilts the string where is match found and return a list of strings where the split have occure
d.
import re
string = "Monday Everyone Should attend the meet at exactly at 4.00 PM along with ur lappy"
#import re
pattern = "a"
re.split(pattern,string,3)
```

Out[40]:
```
['Mond',
 'y Everyone Should ',
 'ttend the meet ',
 't exactly at 4.00 PM along with ur lappy']
```

In [48]:
```
#Sub function:
#Syntax:  re.sub(pattern,replace,string)
 # Its retunrs a string where matched occurences are replaced with values of replace variable.
import re
string = "abc 123\ xyz \n 3563 464848 \n python 36353626 \n"
pattern = "\s" #Returns a match where the string contains a white space character
replace = ""
new_string = re.sub(pattern,replace,string)
new_string
```

Out[48]:
```
'abc123\\xyz3563464848python36353626'
```

In [55]:
```
#finditer():
string = "apssdc is always inform to students along with latest information"
for word in re.finditer("inform.",string):   #. means its matches any character (expect newline character)
    res = word.span()
    print(res)
print(len(string))
```

```
(17, 24)
(54, 61)
65
```

In [ ]:
```
#practical cases of regular Expressions:
1. Phone Number Validation
2. E-mail Address Validation
3. Web Scraping
```

In [ ]:
```
#Phone Number Validation:
444-333-12345
356-679-4546
67-7363-3737
536-336-9999
#Rules :
        1. starts with 3 digits and - sign and
        2. middle 3 digits and - sign
        3  ends with 4 digits.
```

In [58]:
```
#example:
import re
phone = "356-679-4546"
pattern= "\w{3}-\w{3}-\w{4}"  #Returns a match where the string contains any word characters
                              #(characters from a to Z, digits from 0-9,
                              #and the underscore _ character)   and {}-Exactly the specified number of occur
rences
if re.search(pattern,phone):
    print(phone,"num is valid")
else:
    print(phone,"num is not valid")
```

```
356-679-4546 num is valid
```

In [65]:
```
#Standard phone number validation:
#number = "+918328401910"                  #1. starting digit start with 6,7,8,9
number = input("enter your number")             #| -Either or   "falls|stays"    ^-starts with and $-ends wit
h
                                                # []                              A set of characters
pattern = "^[6-9][0-9]{9}$|^[+][9][1][6-9][0-9]{9}$"
if re.search(pattern,number):
    print(number,"its valid number")
else:
    print(number,"its invalid number")
```

```
enter your number8373736336373
8373736336373 its invalid number
```

In [ ]:
```
#Email-Id Validation:
#Rules:
              example: iamsurya93@gmail.com
1. before @ its allow 1 to 25 lower or upper case letters(a-zA-Z) and numbers(0-9),.,_
2. after @ characters (a-z) 3-8 followed by .
3. after .com ,.in,co.in,.info..etc [2-8] characters
```

In [69]:
```
import re
#myemail_id= "iamsurya93@gmail.com"
myemail_id  = input("Enter your mail id")
pattern = "^[a-zA-Z0-9._]{4,25}[@][a-z]{3,8}[.][a-z]{2,8}$"
if re.search(pattern,myemail_id):
    print(myemail_id,"is valid")
else:
    print(myemail_id,"is not valid")
```

```
Enter your mail idkaranam.s@apssdc.in
karanam.s@apssdc.in is valid
```

In [ ]:
```
#Functional Programming:
1. map()
2. filter()
3. reduce()
4. lambda
```

In [ ]:
```
#map() function:
#syntax:
    map(functionname,sequence)
```

In [76]:
```
def addsum(x):
    return x*x+5+pow(x,5)
li = [35,364,4784,4994,9202]      #its applies the functionality to each element of given sequence
list(map(addsum,li))
```

Out[76]:
```
[52523105,
 6390089298325,
 2505854525396302085,
 3106294946057332265,
 6597982270376612841]
```

In [78]:
```
#filter():  filter(functionname,sequence)
def numFilter(n):
    if n>=5:
        return n
list(filter(numFilter,[4,3,5,6,8,9,90,0,27383]))
```

Out[78]:
```
[5, 6, 8, 9, 90, 27383]
```

In [80]:
```
#string filter
def stringFilter(s):
    if s == s[::-1]:
        return s
list(filter(stringFilter,["python","LOL","MADAM","ABA","saturday","today"]))
```

Out[80]:
```
['LOL', 'MADAM', 'ABA']
```

In [ ]: