Functions

Regular Expressions

In [1]:

```
## Funtions
## Function is group of statements to perform specific and rquired task
## Built in functions
## Userdefined Funtions
```

In [6]:

```
## Builtin Functions
# is already developed and available to use
## Range(starting,ending,Step)
## Range(10)
##Range(1,10,1)
## Range(1,10,3)
for i in range(1,10,3):
    print(i,end=":")
```

```
1:4:7:
```

In [7]:

```
# abs()
abs(-10)
```

Out[7]:

```
10
```

In [9]:

```
#len(String)
len("Ravi sastry")
```

Out[9]:

```
11
```

In [ ]:

```
#print()
# help()
help()
```

```
Welcome to Python 3.7's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.7/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help>
```

In [3]:

```
# bin()
bin(3)
```

Out[3]:

```
'0b11'
```

In [4]:

```
# max()
max("RaviZ")
```

Out[4]:

```
'v'
```

In [5]:

```
max("Ravi")
```

Out[5]:

```
'v'
```

In [6]:

```
bin(4)
```

Out[6]:

```
'0b100'
```

In [1]:

```
min("Ravi")
```

Out[1]:

```
'R'
```

In [2]:

```
ord('A')
```

Out[2]:

```
65
```

In [3]:

```
ord('a')
```

Out[3]:

```
97
```

In [4]:

```
ord('Z')
```

Out[4]:

```
90
```

In [5]:

```
ord('z')
```

Out[5]:

```
122
```

In [6]:

```
chr(67)
```

Out[6]:

```
'C'
```

In [7]:

```
chr(97)
```

Out[7]:

```
'a'
```

In [10]:

```
for i in range(ord('A'),ord('Z')+1):
    print(i,end=" ")
```

```
65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89
90
```

In [11]:

```
for i in range(ord('A'),ord('Z')+1):
    if i%2==0:
        print(i,"-->",chr(i))
```

```
66 --> B
68 --> D
70 --> F
72 --> H
74 --> J
76 --> L
78 --> N
80 --> P
82 --> R
84 --> T
86 --> V
88 --> X
90 --> Z
```

In [12]:

```
#dir()
dir(str)
```

Out[12]:

```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rmod__',
 '__rmul__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'capitalize',
 'casefold',
 'center',
 'count',
 'encode',
 'endswith',
 'expandtabs',
 'find',
 'format',
 'format_map',
 'index',
 'isalnum',
 'isalpha',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
 'isupper',
 'join',
 'ljust',
 'lower',
```

```
'lstrip',
'maketrans',
'partition',
'replace',
'rfind',
'rindex',
'rjust',
'rpartition',
'rsplit',
'rstrip',
'split',
'splitlines',
'startswith',
'strip',
'swapcase',
'title',
'translate',
'upper',
'zfill']
```

In [13]:

```
dir(list)
```

Out[13]:

```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__gt__',
 '__hash__',
 '__iadd__',
 '__imul__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__reversed__',
 '__rmul__',
 '__setattr__',
 '__setitem__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'append',
 'clear',
 'copy',
 'count',
 'extend',
 'index',
 'insert',
 'pop',
 'remove',
 'reverse',
 'sort']
```

In [14]:

```
#Sorted()
sorted("ravi")
```

Out[14]:

```
['a', 'i', 'r', 'v']
```

In [15]:

```
sorted("ravi",reverse=True)
```

Out[15]:

```
['v', 'r', 'i', 'a']
```

In [16]:

```
sorted("ravi",reverse=False)
```

Out[16]:

```
['a', 'i', 'r', 'v']
```

In [17]:

```
l=['ravi','ayyappa','ranga']
sorted(l)
```

Out[17]:

```
['ayyappa', 'ranga', 'ravi']
```

In [24]:

```
sum([1,2,3,4])
```

Out[24]:

```
10
```

In [ ]:

```
## User Defined Functions
## it uses the creates Own Functions
##syntax:
#def function_name(Arg1,arg2...):
#### Statements
#### Statements
#function_name(arg1,...)
```

In [26]:

```
def add(a,b):
    print(a+b)
a=10
b=20
add(a,b)
```

```
30
```

In [29]:

```
def add(a,b):
    a=5
    b=10
    print("local variable",a+b)
a=10
b=20
add(a,b)
print("global variable",a+b)
```

```
local variable 15
global variable 30
```

In [30]:

```
def add(a,b):
    return a+b
```

In [31]:

```
add(20,30)
```

Out[31]:

```
50
```

In [32]:

```
def iseven(number):
    if number%2==0:
        return True
    else:
        return False
```

In [33]:

```
iseven(13)
```

Out[33]:

```
False
```

In [41]:

```
def isevennumber(number):
    for i in range(1,number+1):
        if i%2==0:
            print(i)
```

In [39]:

```
isevennumber(20)
```

```
2
4
6
8
10
12
14
16
18
20
```

In [42]:

```
# Foyr types of function Arguments
## Required Arguments
## keyword Arguments
## Default Arguments
### Variable Length Arguments
```

In [46]:

```
# Required Arguments
def add(a,b):
    print(a+b)
add(10,20)
```

```
30
```

In [47]:

```
# Keyword Arguments
def add(a,b):
    print(a+b)
add(a=10,b=20)
```

```
30
```

In [48]:

```
def add(a,b):
    c=20
    print(a+c)
add(a=10,b=30)
```

```
30
```

In [49]:

```
# Default Arguments
def add(c,d,e):
    print(c+d+e)
a=10
b=20
c=30
add(a,b,c)
```

```
60
```

In [51]:

```
# Variable Length Arguments
def add(a,b,*var):
    print(a)
    print(b)
    print(var,type(var))
add(10,20,30,40,50,60)
```

```
10
20
(30, 40, 50, 60) <class 'tuple'>
```

In [52]:

```
# implement the function to check given number prime or not
```

In [53]:

```
# Regular Expressions
## Validate the data
# before using regular expression in your program to must import the regularexpression
## import re
import re
## match(pattern,String)
## search(Pattern,String)
## findall(Pattern,String)

re.match('a',"ravi")
```

In [54]:

```
re.search('a',"ravi")
```

Out[54]:

```
<re.Match object; span=(1, 2), match='a'>
```

In [56]:

```
re.match('avi','avi')
```

Out[56]:

```
<re.Match object; span=(0, 3), match='avi'>
```

In [57]:

```
re.search("a","ravia")
```

Out[57]:

```
<re.Match object; span=(1, 2), match='a'>
```

In [58]:

```
re.findall("a","ravia")
```

Out[58]:

```
['a', 'a']
```

In [60]:

```
re.search('^a','aravi')# ^ matches Begining of a character
```

Out[60]:

```
<re.Match object; span=(0, 1), match='a'>
```

In [61]:

```
re.search('i$','ravi')# $ matches the ending of the line
```

Out[61]:

```
<re.Match object; span=(3, 4), match='i'>
```

In [62]:

```
# . matches the any character
re.search('..','ravi')
```

Out[62]:

```
<re.Match object; span=(0, 2), match='ra'>
```

In [64]:

```
# \s mathes the whitespaces
re.search('\s',"ra vi")
```

Out[64]:

```
<re.Match object; span=(2, 3), match=' '>
```

In [66]:

```
re.match('\s',' ravi')
```

Out[66]:

```
<re.Match object; span=(0, 1), match=' '>
```

In [67]:

```
re.findall('\s','r a v i')
```

Out[67]:

```
[' ', ' ', ' ']
```

In [68]:

```
x=re.findall('\s','apssdc')
x
```

Out[68]:

```
[]
```

In [70]:

```
# \S mathes the non-whitespaces
re.search("\S+"," ravi")
```

Out[70]:

```
<re.Match object; span=(1, 5), match='ravi'>
```

In [72]:

```
re.search("\S+","ra  vi")
```

Out[72]:

```
<re.Match object; span=(0, 2), match='ra'>
```

In [75]:

```
# \d mathes the digits
re.search('\d+','ravi12 3')
```

Out[75]:

```
<re.Match object; span=(4, 6), match='12'>
```

In [76]:

```
# \D mathes the non digits
re.search('\D','123ravi ')
```

Out[76]:

```
<re.Match object; span=(3, 4), match='r'>
```

In [77]:

```
# \w mathes the non Special characters
re.search('\w','@#123')
```

Out[77]:

```
<re.Match object; span=(2, 3), match='1'>
```

In [78]:

```
# \W mathes the special characters
re.search('\W','123@#')
```

Out[78]:

```
<re.Match object; span=(3, 4), match='@'>
```

In [81]:

```
# * mathes the charcters zero or more times
re.search('a*','aaaavaai')
```

Out[81]:

```
<re.Match object; span=(0, 4), match='aaaa'>
```

In [82]:

```
re.findall('a*','aaaavaai')
```

Out[82]:

```
['aaaa', '', 'aa', '', '']
```

In [88]:

```
l=['ravi','ayyappa','anii','srinivas','ranga']
for i in l:
    if re.search('^...i$',i):
        print(i)
```

```
ravi
anii
```

In [90]:

```
# [aeiou] mathes a single character in the listed set
re.findall('[aeiou]','ravi')
```

Out[90]:

```
['a', 'i']
```

In [91]:

```
re.findall('[aeiou]','aouie')
```

Out[91]:

```
['a', 'o', 'u', 'i', 'e']
```

In [92]:

```
re.findall('[aeiou]','aaioue')
```

Out[92]:

```
['a', 'a', 'i', 'o', 'u', 'e']
```

In [94]:

```
re.search('[aeiou]+','aaioue')
```

Out[94]:

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

In [96]:

```
re.search('[a-z]+','Abcde')
```

Out[96]:

```
<re.Match object; span=(1, 5), match='bcde'>
```

In [100]:

```
re.search('[A-Za-z0-9]+','ABCabc123')
```

Out[100]:

```
<re.Match object; span=(0, 9), match='ABCabc123'>
```

In [101]:

```
re.search('a+','raavii')
```

Out[101]:

```
<re.Match object; span=(1, 3), match='aa'>
```

In [102]:

```
re.findall('a','raavii')
```

Out[102]:

```
['a', 'a']
```

In [112]:

```
l=['4567893453','678945322r','9866644432','09505060477']
for i in l:
    if re.findall('^[6-9][0-9]{9}$|^[0][6-9][0-9]{9}$',i):
        print(i)
```

```
9866644432
09505060477
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```

```

In [ ]:

```

```