- String methods
- Data Structures
    - Lists
    - Tuple
    - Dictionary
    - Set

In [6]:
```
print(dir(str),end=' ')
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewa
rgs__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__l
e__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce
__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__siz
eof__', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'cou
nt', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'inde
x', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'i
slower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join',
'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rind
ex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startsw
ith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
```

In [11]:
```
s = "python workshop"
print(s.capitalize()) ## It converts the first character of the string into upper
```

```
Python workshop
```

In [12]:
```
s1 = "good afternoon all"
print(s1.title()) ## converts the first character of every word in the given stri
```

```
Good Afternoon All
```

In [13]:
```
s = 'Hello Hii'
s.casefold() ## it converts lowercase
```

Out[13]:
```
'hello hii'
```

In [14]:
```
print(s.lower())
print(s.upper())
```

```
hello hii
HELLO HII
```

In [15]:
```
s = 'python'
print(s.startswith('h'))
print(s.endswith('n'))
```
```
False
True
```

In [16]:
```
s2 = 'python programming'
print(s2.count('m'))
```
```
2
```

In [19]:
```
print(s2.index('o'))
print(s2.index('z'))
```
```
4

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-19-43bc76e7e75f> in <module>
      1 print(s2.index('o'))
----> 2 print(s2.index('z'))

ValueError: substring not found
```

In [20]:
```
print(s2.find('o'))
print(s2.find('z'))
```
```
4
-1
```

In [21]:
```
s3 = '34python'
print(s3.isidentifier()) ## it returns true when all the character in the given s
```
```
False
```

In [22]:
```
s = '_python'
s.isidentifier()
```
Out[22]:
```
True
```

In [23]:
```
## 0 1 2 3 4 -- 9

print(s3.isdigit())
```
```
False
```

In [27]:
```
t = '123'
print(t.isdigit())
```
```
True
```

In [28]:
```
print(s3.isalnum())
```
```
True
```

In [29]:
```
y = '34@house'
print(y.isalnum())
```
```
False
```

In [33]:
```
l = '1234'
print(l.isdecimal())
```
```
True
```

In [34]:
```
b = 'asdfertioy'
print(b.isprintable())
```
```
True
```

In [37]:
```
r = "   "
print(r.isspace())
```
```
True
```

In [39]:
```
a = 'python programming'
print(a.find('o'))
print(a.rfind('o'))
```
```
4
9
```

In [43]:
```
s = '   python  '
print(s.strip())
print(s.lstrip())
print(s.rstrip())
```
```
python
python
   python
```

In [44]:
```
s = 'PyThOn'
s.swapcase()
```
Out[44]:
```
'pYtHoN'
```

In [46]:
```
a = 'python workshop'
print(a.split())
print(a.split('o'))
```

```
['python', 'workshop']
['pyth', 'n w', 'rksh', 'p']
```

In [50]:
```
b = 'N', 'a', 'n',  'd', 'i',  'n',  'i'
print(' '.join(b))
print('@'.join(b))
```

```
N a n d i n i
N@a@n@d@i@n@i
```

In [51]:
```
d = 'zpython'
print(d.replace('z','a'))
```

```
apython
```

In [52]:
```
t = 'java python'
print(t.replace('java','c++'))
```

```
c++ python
```

In [58]:
```
u = 'apssdc'
print(u.center(20))
print(u.center(20,'*'))
print(u)
```

```
       apssdc
*******apssdc*******
apssdc
```

In [79]:
```
a = 'hello\thi\t2345\tabc'
b = 'hellohi2345'
print(a.expandtabs())
print(b.expandtabs())
print(a)
```

```
hello   hi      2345    abc
hellohi2345
hello   hi      2345    abc
```

In [80]:
```
s = 'Nandini\tSurya\t12'
print(s)
print('Nandini\tSurya')
print('nandini_surya')
print('nandini\nsurya')
```

```
Nandini Surya   12
Nandini Surya
nandini_surya
nandini
surya
```

In [71]:
```
g = 'stay home stay safe'
print(g.partition('t'))
print(g.partition('y'))
```

```
('s', 't', 'ay home stay safe')
('sta', 'y', ' home stay safe')
```

In [66]:
```
g = 'stay home stay safe'
print(g.split())
```

```
['stay', 'home', 'stay', 'safe']
```

In [61]:
```
ty = 'workshop'
print(ty.zfill(15))
```

```
0000000workshop
```

- format
- formatmap
- translate

In [64]:
```
h = 'CAT'
y = 'rat'
print(h.islower())
print(h.isupper())
print(y.islower())
```

```
False
True
True
```

In [65]:
```
help(str)
```

```
Help on class str in module builtins:

class str(object)
 |  str(object='') -> str
 |  str(bytes_or_buffer[, encoding[, errors]]) -> str
 |  
 |  Create a new string object from the given object. If encoding or
 |  errors is specified, then the object must expose a data buffer
 |  that will be decoded using the given encoding and error handler.
 |  Otherwise, returns the result of object.__str__() (if defined)
 |  or repr(object).
 |  encoding defaults to sys.getdefaultencoding().
 |  errors defaults to 'strict'.
 |  
 |  Methods defined here:
 |  
 |  __add__(self, value, /)
 |      Return self+value.
 |  
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Data Structures

- Data structures are a way of organizing data and stroring data
- We have types of data structures
  - 1.List
  - 2.Tuple
  - 3.Dictionary
  - 4.Set

List

- One of the data structures
- Storing data in a order
- List is mutable
- List follows indexing and slicing

- It allows heterogenous data
- We are representing list with square braces '[]'

In [81]:
```
## Empty list

l = []
type(l)
```
Out[81]:
```
list
```

In [82]:
```
li = [1,2,3,'name',6.78]
li
```
Out[82]:
```
[1, 2, 3, 'name', 6.78]
```

In [83]:
```
li[0]
```
Out[83]:
```
1
```

In [84]:
```
li[4]
```
Out[84]:
```
6.78
```

In [85]:
```
li
```
Out[85]:
```
[1, 2, 3, 'name', 6.78]
```

In [86]:
```
li[2] = 'ap'
```

In [87]:
```
li
```
Out[87]:
```
[1, 2, 'ap', 'name', 6.78]
```

In [96]:
```
## built-in functions for list

l = [60,2,8,0,1,6,3]
print(type(l))
print(len(l))
print(max(l))
print(min(l))
print(sum(l))
print(sorted(l))
print(sorted(l,reverse=True))
```

```
<class 'list'>
7
60
0
80
[0, 1, 2, 3, 6, 8, 60]
[60, 8, 6, 3, 2, 1, 0]
```

In [97]:
```
l
```

Out[97]:
```
[60, 2, 8, 0, 1, 6, 3]
```

In [103]:
```
## Accessing list elements
# Indexing

print(l)
print(l[2]) ## forward index
print(l[-3]) ## backward index
print(l[1::2])
print(l[::-1])
```

```
[60, 2, 8, 0, 1, 6, 3]
8
1
[2, 0, 6]
[3, 6, 1, 0, 8, 2, 60]
```

In [105]:
```
l = [1,2,3,[5,6,7],'python']
print(l[3])
print(l[3][1])
```

```
[5, 6, 7]
6
```

In [106]:
```
## Concatination

li1 = [7,4]
li2 = ['god','dog']
li1 + li2
```

Out[106]:
```
[7, 4, 'god', 'dog']
```

In [107]:
```
## repetition

li1*4
```

Out[107]:
```
[7, 4, 7, 4, 7, 4, 7, 4]
```

In [108]:
```
l
```

Out[108]:
```
[1, 2, 3, [5, 6, 7], 'python']
```

In [111]:
```
# print(l[0])

for i in l:
    print(i)
```

```
1
2
3
[5, 6, 7]
python
```

In [114]:
```
## addition of even numbers in the list
## input: li = [1,2,3,4]
## output: 6

li = [1,2,3,4,7,3,9,2]
# print(li[1]+li[3])
s = 0
for i in li:
    if(i%2 == 0):
        s += i
s
```

Out[114]:
```
8
```

In [115]:
```
print(dir(list),end=' ')
```

```
['__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__', '__setat
tr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'append', 'c
lear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'revers
e', 'sort']
```

In [116]:
```
l1 = [12,5,7,9,3]
l1.append('hello')
l1
```

Out[116]:
```
[12, 5, 7, 9, 3, 'hello']
```

In [117]:
```
l1.clear()
```

In [118]:
```
l1
```

Out[118]:
```
[]
```

In [119]:
```
del l1
```

In [120]:
```
l1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-120-6cf485bc2797> in <module>
----> 1 l1

NameError: name 'l1' is not defined
```

In [126]:
```
l = [1,2,3,4]
h = [5,6,7,8]
l.extend(h)
h.extend(l)
print(l)
print(h)
l.extend(h)
print(l)
```

```
[1, 2, 3, 4, 5, 6, 7, 8]
[5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8]
[1, 2, 3, 4, 5, 6, 7, 8, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8]
```

In [127]:
```
l
```

Out[127]:
```
[1, 2, 3, 4, 5, 6, 7, 8, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8]
```

In [128]:
```
l.count(1)
```

Out[128]:
```
2
```

In [129]:
```
l.count(5)
```

Out[129]:
```
3
```

```
In [130]: l.index(3)

Out[130]: 2

In [131]: l.pop()

Out[131]: 8

In [132]: l.pop()

Out[132]: 7

In [133]: l

Out[133]: [1, 2, 3, 4, 5, 6, 7, 8, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6]

In [134]: l.pop()

Out[134]: 6

In [135]: l.remove(5)  ## syntax: remove(item/element)

In [136]: l

Out[136]: [1, 2, 3, 4, 6, 7, 8, 5, 6, 7, 8, 1, 2, 3, 4, 5]

In [138]: l.append('abc')
          l

Out[138]: [1, 2, 3, 4, 6, 7, 8, 5, 6, 7, 8, 1, 2, 3, 4, 5, 'abc']

In [139]: l.remove('abc')

In [140]: l

Out[140]: [1, 2, 3, 4, 6, 7, 8, 5, 6, 7, 8, 1, 2, 3, 4, 5]

In [141]: ## insert : Syntax: insert(index position,value)
          l.insert(3,'z')

In [142]: l

Out[142]: [1, 2, 3, 'z', 4, 6, 7, 8, 5, 6, 7, 8, 1, 2, 3, 4, 5]
```

```
In [144]: k = [0,7,5,1,2]
          k.sort()
          k

Out[144]: [0, 1, 2, 5, 7]

In [145]: k.reverse()

In [146]: k

Out[146]: [7, 5, 2, 1, 0]

In [147]: l = [123,'str',9.54]
          l.reverse()
          l

Out[147]: [9.54, 'str', 123]

In [151]: r1 = ['q','w','r']
          r2 = r1.copy()
          r2

Out[151]: ['q', 'w', 'r']

In [ ]:

In [ ]:
```

Tuple

- We are representing tuple with open braces '()'
- tuple is immutable

```
In [152]: print(dir(tuple),end=' ')

          ['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__',
          '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewa
          rgs__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__l
          e__', '__len__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__red
          uce_ex__', '__repr__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__s
          ubclasshook__', 'count', 'index']

In [153]: t = (123,'abc',5.67)
          t

Out[153]: (123, 'abc', 5.67)
```

```
In [154]: type(t)

Out[154]: tuple

In [155]: len(t)

Out[155]: 3

In [156]: t[0]

Out[156]: 123

In [157]: t[-1]

Out[157]: 5.67

In [158]: t[0] = 'name'
          t

          ---------------------------------------------------------------------------
          TypeError                                 Traceback (most recent call last)
          <ipython-input-158-8d2a9bb258df> in <module>
          ----> 1 t[0] = 'name'
                2 t

          TypeError: 'tuple' object does not support item assignment

In [159]: t

Out[159]: (123, 'abc', 5.67)

In [160]: t.count('abc')

Out[160]: 1

In [164]: t.index(5.67)

Out[164]: 2

In [165]: t.remove(5.67)

          ---------------------------------------------------------------------------
          AttributeError                            Traceback (most recent call last)
          <ipython-input-165-626f2b359c74> in <module>
          ----> 1 t.remove(5.67)

          AttributeError: 'tuple' object has no attribute 'remove'
```

```
In [166]: t

Out[166]: (123, 'abc', 5.67)

In [167]: type(t)

Out[167]: tuple

In [168]: new = list(t)
          new

Out[168]: [123, 'abc', 5.67]

In [169]: new.remove(5.67)

In [170]: new

Out[170]: [123, 'abc']

In [171]: t = tuple(new)

In [172]: t

Out[172]: (123, 'abc')

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```

Dicionary

- A dictionary is a collection of unordered data
- Which is mutable
- Represent in curly brackets --> {}
- They have keys and values
- Combination of keys and values we are calling a item

In [173]:
```
## creating a dictionary

dic = {'name':'Nandini','clg':'RGUKT','Id_no':130663}
dic
```

Out[173]:
```
{'name': 'Nandini', 'clg': 'RGUKT', 'Id_no': 130663}
```

In [176]:
```
## Accessing

dic['name']
# dic[0]
```

Out[176]:
```
'Nandini'
```

In [177]:
```
## changing

dic['name'] = 'Vanitha'
```

In [178]:
```
dic
```

Out[178]:
```
{'name': 'Vanitha', 'clg': 'RGUKT', 'Id_no': 130663}
```

In [179]:
```
## access the keys by looping

for i in dic:
    print(i)
```

```
name
clg
Id_no
```

In [181]:
```
for i in dic:
    print(dic[i])
```

```
Vanitha
RGUKT
130663
```

In [182]:
```
for i,j in dic.items():
    print(i,j)
```

```
name Vanitha
clg RGUKT
Id_no 130663
```

In [183]:
```
print(dir(dict),end=' ')
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc_
_', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt_
_', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len_
_', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__',
'__reversed__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subcl
asshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys', 'pop', 'popite
m', 'setdefault', 'update', 'values']
```

In [184]:
```
dic
```

Out[184]:
```
{'name': 'Vanitha', 'clg': 'RGUKT', 'Id_no': 130663}
```

In [185]:
```
dic.values()
```

Out[185]:
```
dict_values(['Vanitha', 'RGUKT', 130663])
```

In [186]:
```
dic.keys()
```

Out[186]:
```
dict_keys(['name', 'clg', 'Id_no'])
```

In [187]:
```
dic.items()
```

Out[187]:
```
dict_items([('name', 'Vanitha'), ('clg', 'RGUKT'), ('Id_no', 130663)])
```

In [188]:
```
len(dic)
```

Out[188]:
```
3
```

In [189]:
```
dic.clear()
```

In [190]:
```
dic
```

Out[190]:
```
{}
```

In [191]:
```
del dic
```

In [192]:
```
## fromkeys
## returns a dictionary with the specified key and value

## Syntax: fromkeys(seq,value)

a = ['a','e','i','o','u']
b = dict.fromkeys(a)
b
```

Out[192]:
```
{'a': None, 'e': None, 'i': None, 'o': None, 'u': None}
```

In [205]:
```
## get
## Returns the value of the specified key

## Syntax: get(key,default=None)

r = {'a':1,'b':2,'c':3}
r.get('f','none')
r.get('b')
print(r.get('t'))
```

```
None
```

In [196]:
```
r
```

Out[196]:
```
{'a': 1, 'b': 2, 'c': 3}
```

In [207]:
```
## syntax: pop(key_elemnt)
r.pop('a')
```

Out[207]:
```
1
```

In [208]:
```
r
```

Out[208]:
```
{'b': 2, 'c': 3}
```

In [209]:
```
## popitem()

r.popitem()
```

Out[209]:
```
('c', 3)
```

In [210]:
```
r
```

Out[210]:
```
{'b': 2}
```

In [211]:
```
help(dict)
```

```
Help on class dict in module builtins:

class dict(object)
 |  dict() -> new empty dictionary
 |  dict(mapping) -> new dictionary initialized from a mapping object's
 |      (key, value) pairs
 |  dict(iterable) -> new dictionary initialized as if via:
 |      d = {}
 |      for k, v in iterable:
 |          d[k] = v
 |  dict(**kwargs) -> new dictionary initialized with the name=value pairs
 |      in the keyword argument list.  For example:  dict(one=1, two=2)
 |
 |  Built-in subclasses:
 |      StgDict
 |
 |  Methods defined here:
 |
 |  __contains__(self, key, /)
```

In [ ]: