Introduction to Data Structure in python

Organising Data in a certain format is known as Data structure

1.List 2.Touple 3.Dict 4.set

1.List

```
In [6]:  li=[]
         type(li)

Out[6]:  list

In [7]:  li#to print entire list

Out[7]:  [1, 2]

In [15]: li=[1,2,34,5,6]
         li[0]# index of list starts from 0

Out[15]: 1

In [16]: li[0:]# accessing all elements

Out[16]: [1, 2, 34, 5, 6]

In [20]: li[:-1]# Reverse of a list

Out[20]: [1, 2, 34, 5]

In [21]: li[:-1]

Out[21]: [1, 2, 34, 5]

In [23]: li[1:3]

Out[23]: [2, 34]

In [24]: li[::-2]

Out[24]: [6, 34, 1]

In [25]: li[::2]

Out[25]: [1, 34, 6]

In [26]: li[::1]

Out[26]: [1, 2, 34, 5, 6]
```

```
In [27]: li[::-1]

Out[27]: [6, 5, 34, 2, 1]

In [28]: li[1::1]

Out[28]: [2, 34, 5, 6]

In [30]: li[1::2]# step=2-1=1 elements will print from index 1 to end alternatively

Out[30]: [2, 5]

In [35]: li[-1::-2]

Out[35]: [6, 34, 1]

In [36]: li[-3]

Out[36]: 34

In [ ]: li[-1::1]

In [3]: name="keerthi"
        type(name)

Out[3]: str

In [5]: name=list()
        type(name)

Out[5]: list

In [6]: name=["katakam","keerthi"]

In [7]: name[0]

Out[7]: 'katakam'
```

append

```
In [8]: name.append("venkata")

In [10]: name

Out[10]: ['katakam', 'keerthi', 'venkata']

In [11]: name.append(["hi","gi","priya"])

In [12]: name

Out[12]: ['katakam', 'keerthi', 'venkata', ['hi', 'gi', 'priya']]

In [13]: len(name)

Out[13]: 4
```

extend

```
In [17]: name1=["manne","priya","priya"]
```

```
In [18]: name.extend(name1)
```

```
In [20]: name
         len(name)
```

```
Out[20]: 9
```

copy

To take new backup of available list

```
In [21]: f=[1,2,3,4]
         d=f.copy()
```

```
In [22]: d
```

```
Out[22]: [1, 2, 3, 4]
```

```
In [23]: girls=["vijaya","rekha"]
```

```
In [24]: girls[0]="jaya"
```

```
In [25]: girls
```

```
Out[25]: ['jaya', 'rekha']
```

```
In [26]: dir(list)
```

```
Out[26]: ['__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']
```

clear

To remove all items of list

```
In [28]: g=[1,2,3]
         g.clear()
```

```
In [29]: g
```

```
Out[29]: []
```

Reverse

```
In [30]: s=[1,2,3,4,5]
         f.reverse()
```

```
In [31]: f
```

```
Out[31]: [4, 3, 2, 1]
```

count(item)

It can search for given item and return the number of occurances in a list

```
In [32]: girls=["bhanu","bhanu","subha","yamini"]
```

```
In [33]: girls.count("bhanu")
```

```
Out[33]: 2
```

```
In [38]: x=[1,2,[1,2,8]]
```

```
In [42]: x.count(2)
```

```
Out[42]: 1
```

index(item,start_index for searching)

```
In [69]: w=[1,2,3,56,2,4,67,2,3,4]
         w.index(2,2)
```

```
Out[69]: 4
```

insert(indexvalue,item)

```
In [70]: w.insert(0,24)
```

```
In [76]: w
```

```
Out[76]: [24, 1, 2, 3, 56, 2, 4, 67, 2, 3, 4]
```

```
In [77]: w.insert(100,30)
```

```
In [78]: w
```

```
Out[78]: [24, 1, 2, 3, 56, 2, 4, 67, 2, 3, 4, 30]
```

```
In [ ]: # pop() Default it removes last element
        # pop(indexvalue) Remove index value element in the list
```

```
In [79]: w.pop()

Out[79]: 30
```

```
In [80]: w

Out[80]: [24, 1, 2, 3, 56, 2, 4, 67, 2, 3, 4]
```

```
In [81]: w.pop(3)

Out[81]: 3
```

```
In [83]: w

Out[83]: [24, 1, 2, 56, 2, 4, 67, 2, 3, 4]
```

```
In [82]: # remove
```

```
In [86]: w.remove(2)
```

```
In [88]: w

Out[88]: [1, 56, 2, 4, 67, 2, 3, 4]
```

```
In [89]: # sort
```

```
In [94]: q=["Bharath","keerthi","Suseela",1,2]# can't sort the string with number
         q.sort()

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-94-235a710f7791> in <module>
      1 q=["Bharath","keerthi","Suseela",1,2]
----> 2 q.sort()
      3

TypeError: '<' not supported between instances of 'int' and 'str'
```

```
In [101]: a=["hi","HI"]
          a.sort()
          a

Out[101]: ['HI', 'hi']
```

```
In [102]: r=[13,2,52,6,24,89]
          r.sort(reverse=True)
          r

Out[102]: [89, 52, 24, 13, 6, 2]
```

```
In [104]: v=[[2,3],[6,7],[4,5,6]]
          v.sort()
          v

Out[104]: [[2, 3], [4, 5, 6], [6, 7]]
```

```
In [105]: # function to identify the second largest element in a unique list
                      # sort the datat and select second last element
          def secondlargest(li):
              li.sort()
              return li[-2]
          secondlargest(r)

Out[105]: 52
```

```
In [108]: # sort the data in reverse order and select second elements
          def secondlarge(li):
              li.sort(reverse=True)
              return li[1]
          secondlarge(r)

Out[108]: 52
```

```
In [128]: p=[1,2,3,7,4]
```

```
In [130]: # remove the max element and find the max element that gives the second max
          def seclarge(li):
              li.sort()
              li.pop()
              return li[-1]
          seclarge(p)

Out[130]: 3
```

```
In [ ]:
```