WHAT ARE COLLECTIONS IN PYTHON

* There are four collection data types in python which are used to store collections of data.
  + LIST
  + TUPLES
  + SETS
  + DICTIONARY

WHAT IS A COLLECTION MODULE IN PYTHON?

* Built in modules has collection specialist for data structures.

SPECIALISED COLLECTIONS DATA TYPES

* namedtuple()
  + returns a tuple with named value for each element in the tuple.
  + Details = (name = ‘osama’, course = ‘python’, course2 = ‘data science’)
  + **from** collections **import** namedtuple
  + a = namedtuple(**'courses'**, **'name, technology'**)  
    s = a(**'data science'**,**'python'**)  
    d = a.\_make([**'artificial intelligence'**,**'python'**])  
    print(s)  
    print(d)
  + Output: courses(name='data science', technology='python')

courses(name='artificial intelligence', technology='python')

* Deque
  + Deque pronounced as ‘deck’ is an optimized list to perform insertion and deletion easily.
  + **from** collections **import** deque  
      
    a = [**'o'**,**'s'**,**'a'**,**'m'**,**'a'**]  
    d = deque(a)  
    print(d)  
    d.append(**'python'**)  
    print(d)  
    d.appendleft(**'python'**)  
    print(d)  
    d.pop()  
    print(d)  
    d.popleft()  
    print(d)
  + output: deque(['o', 's', 'a', 'm', 'a'])

deque(['o', 's', 'a', 'm', 'a', 'python'])

deque(['python', 'o', 's', 'a', 'm', 'a', 'python'])

deque(['python', 'o', 's', 'a', 'm', 'a'])

deque(['o', 's', 'a', 'm', 'a'])

* Chainmap
  + Chainmap is a dictionary like class for creating a single view of multiple mappings.
  + **from** collections **import** ChainMap  
    a = {1: **'osama'** , 2: **'python'**}  
    b = {3: **'ML'** , 4: **'AI'**}  
    a1 = ChainMap(a,b)  
    print(a1)
  + Output: ChainMap({1: 'osama', 2: 'python'}, {3: 'ML', 4: 'AI'})
* Counter
  + Counter is a dictionary subclass for counting hashtable objects
  + **from** collections **import** Counter  
    a = [1,1,2,2,2,3,2,4,5,4,5,2,5,4,6,2,2,]  
    c = Counter(a)  
    print(c)  
    print(list(c.elements()))  
    print(c.most\_common())  
    sub = {2:1, 6:1}  
    c.subtract(sub)  
    print(c.most\_common())
  + Output: Counter({2: 7, 4: 3, 5: 3, 1: 2, 3: 1, 6: 1})

[1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 4, 4, 4, 5, 5, 5, 6]

[(2, 7), (4, 3), (5, 3), (1, 2), (3, 1), (6, 1)]

[(2, 6), (4, 3), (5, 3), (1, 2), (3, 1), (6, 0)]

* OrderedDict
  + OrderedDict is a dictionary subclass which remembers the order in which entries were done.
  + **from** collections **import** OrderedDict  
      
    d = OrderedDict()  
    d[1] = **'o'**d[2] = **'s'**d[3] = **'a'**d[4] = **'m'**d[5] = **'a'**print(d)  
    print(d.keys())  
    print(d.items())  
    d[1] = **'a'**print(d)
  + Output: OrderedDict([(1, 'o'), (2, 's'), (3, 'a'), (4, 'm'), (5, 'a')])

odict\_keys([1, 2, 3, 4, 5])

odict\_items([(1, 'o'), (2, 's'), (3, 'a'), (4, 'm'), (5, 'a')])

OrderedDict([(1, 'a'), (2, 's'), (3, 'a'), (4, 'm'), (5, 'a')])

* Defaultdict
  + Defaultdict is a dictionary subclass which calls a factory function to supply missing values.
  + **from** collections **import** defaultdict  
    d = defaultdict(int)  
    d[1] = **'python'**d[2] = **'osama'**print(d)  
    print(d[3])
  + Output: defaultdict(<class 'int'>, {1: 'python', 2: 'osama'})

0

* UserDict
  + UserDict is a wrapper around dictionary objects for easier dictionary sub-classing.
* Userlist
  + UserList is a wrapper around list of objects for easier List sub-classing.
* UserString
  + UserString is a wrapper around string objects for easier string sub-classing.