Tuple is ordered and un-mutable, can have duplicate values

written within ()

In [1]:
```
tuple_1= ("a","b","c","b")
tuple_1
```
Out[1]:
```
('a', 'b', 'c', 'b')
```

In [2]:
```
tuple_1[1]
```
Out[2]:
```
'b'
```

In [3]:
```
len(tuple_1)
```
Out[3]:
```
4
```

In [4]:
```
tuple_2= ("a","b","c","b",11,2323,[[12,32],12,54], (12,43,45))
tuple_2
```
Out[4]:
```
('a', 'b', 'c', 'b', 11, 2323, [[12, 32], 12, 54], (12, 43, 45))
```

In [5]:
```
type(tuple_1)
```
Out[5]:
```
tuple
```

In [6]:
```
tuple_1[-4:-1]
```
Out[6]:
```
('a', 'b', 'c')
```

In [7]:
```
tuple_1= ("a","b","c","b")

#add "Delhi"
x=list(tuple_1)
x.append("Delhi")
tuple_1=tuple(x)
tuple_1
```
Out[7]:
```
('a', 'b', 'c', 'b', 'Delhi')
```

In [13]:
```
(x,y,z,*k)=("as","was","he","r","o","l")
print(x)
print(y)
print(z)
```
```
as
was
he
```

In [8]:
```
#unpacking
(k,h,m,n,*r)=("a","b","c","b",4,5,6,7,8,8,98)
print(n)
print(r)
```
```
b
[4, 5, 6, 7, 8, 8, 98]
```

In [14]:
```
t3=tuple_1+tuple_2
t3
```
Out[14]:
```
('a',
 'b',
 'c',
 'b',
 'Delhi',
 'a',
 'b',
 'c',
 'b',
 11,
 2323,
 [[12, 32], 12, 54],
 (12, 43, 45))
```

```
In [16]: t3.count("a") #it tells the counting of a particular item

Out[16]: 2

In [17]: t3.index("a")

Out[17]: 0

In [19]: T_a= ('Delhi', 'String', 'Mumbai', 'Manali', 'python', 'java')
         len(T_a)

Out[19]: 6

In [20]: range(6)

Out[20]: range(0, 6)

In [21]: range(len(T_a))

Out[21]: range(0, 6)

In [22]: for i in range(len(T_a)):
             print(T_a[i])

         Delhi
         String
         Mumbai
         Manali
         python
         java

In [24]: [T_a[x] for x in range(len(T_a))]

Out[24]: ['Delhi', 'String', 'Mumbai', 'Manali', 'python', 'java']

In [25]: [x for x in T_a]

Out[25]: ['Delhi', 'String', 'Mumbai', 'Manali', 'python', 'java']
```

Sets

{} duplicate values nhi hoti un-indexed unordered

unchangeable * set items are unchangeable but you can remove items and add new items

```
In [30]: set_1={"a","b","c","b","b"}
         set_1

Out[30]: {'a', 'b', 'c'}

In [29]: len(set_1)

Out[29]: 3

In [31]: type(set_1)

Out[31]: set

In [44]: set_2=set((1,2,3,4,"abc", True,False))

In [34]: set_3={True,True,False,False}
         set_3

Out[34]: {False, True}

In [47]: l4={x for x in range(6)}
         l4

Out[47]: {0, 1, 2, 3, 4, 5}
```

```
In [48]: set_1={"a","b","c","b","b"}
         "e" in set_1

Out[48]: False

In [49]: print("b" in set_1)

         True

In [50]: set_4={1,0,True,False}  #True = 1, False= 0
         set_4

Out[50]: {0, 1}

In [51]: set_4={True,False}
         set_4

Out[51]: {False, True}

In [52]: set_4={1,0,True,False}
         set_4.add("Delhi")
         set_4

Out[52]: {0, 1, 'Delhi'}

In [53]: set_4.append("kol")
         set_4

         ---------------------------------------------------------------------------
         AttributeError                            Traceback (most recent call last)
         Input In [53], in <cell line: 1>()
         ----> 1 set_4.append("kol")
               2 set_4

         AttributeError: 'set' object has no attribute 'append'

In [54]: set_4.update(set_1)  #to ass item from another set into the current set we use update()
         set_4

Out[54]: {0, 1, 'Delhi', 'a', 'b', 'c'}

In [55]: set_4 + set_2
         set_4

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         Input In [55], in <cell line: 1>()
         ----> 1 set_4 + set_2
               2 set_4

         TypeError: unsupported operand type(s) for +: 'set' and 'set'

In [56]: 

         ---------------------------------------------------------------------------
         AttributeError                            Traceback (most recent call last)
         Input In [56], in <cell line: 1>()
         ----> 1 set_4.extend(set_1)
               2 set_4

         AttributeError: 'set' object has no attribute 'extend'

In [57]: list_1=[45,45,56,67,78,78]
         set_4.update(list_1)
         set_4

Out[57]: {0, 1, 45, 56, 67, 78, 'Delhi', 'a', 'b', 'c'}
```

In [58]:
```
#to remove from set we usse the remove() or the discard() method
set_4.remove(67)
set_4
```
Out[58]:
```
{0, 1, 45, 56, 78, 'Delhi', 'a', 'b', 'c'}
```

In [59]:
```
set_4.remove(67)
set_4
```
```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Input In [59], in <cell line: 1>()
----> 1 set_4.remove(67)
      2 set_4

KeyError: 67
```

In [60]:
```
set_4.remove(0,1)
set_4
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [60], in <cell line: 1>()
----> 1 set_4.remove(0,1)
      2 set_4

TypeError: set.remove() takes exactly one argument (2 given)
```

In [61]:
```
set_4.discard(0,1)
set_4
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [61], in <cell line: 1>()
----> 1 set_4.discard(0,1)
      2 set_4

TypeError: set.discard() takes exactly one argument (2 given)
```

In [62]:
```
set_4.remove("Delhi")
set_4
```
Out[62]:
```
{0, 1, 45, 56, 78, 'a', 'b', 'c'}
```

In [66]:
```
set_2.clear()
set_2
```
Out[66]:
```
set()
```

In [67]:
```
set1={"a","b","c","b","b"}
set1.pop()
set1
```
Out[67]:
```
{'a', 'c'}
```

In [68]:
```
del set_4

set_4
```
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [68], in <cell line: 2>()
      1 del set_4
----> 2 set_4

NameError: name 'set_4' is not defined
```

```
In [71]: set_a={1,2,3,4,5}
         set_b={6,7,8,9,10}

         set_c=set_a.union(set_b)  # c=a+b
         set_c

Out[71]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

```
In [74]: set_a={1,2,3,4,5}
         set_b={6,7,8,9,10}

         set_a.update(set_b)  #update simply updates the set on which we are applying the update method not add
         set_a

Out[74]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

```
In [76]: set_a={1,2,3,4,5}
         set_b={6,7,8,9,10}

         set_c=set_a.update(set_b)
         print(set_c)

         None
```

```
In [78]: #intersection_update()
         set_a={1,2,3,4,5,6,7}
         set_b={6,7,8,9,10}

         set_a.intersection_update(set_b)
         set_a

Out[78]: {6, 7}
```

```
In [79]: #intersection is what is common among both the items
         set_a={1,2,3,4,5,6,7}
         set_b={6,7,8,9,10}

         set_c=set_a.intersection(set_b)
         set_c

Out[79]: {6, 7}
```

```
In [80]: #symmetric_difference_update THROWS the common Away.
         #jo common nhi hai dono mei use faik dega, baki return kr dega
         set_a={1,2,3,4,5,6,7}
         set_b={6,7,8,9,10}

         set_a.symmetric_difference_update(set_b)
         set_a

Out[80]: {1, 2, 3, 4, 5, 8, 9, 10}
```

```
In [81]: set_a={1,2,3,4,5,6,7}
         set_b={6,7,8,9,10}

         set_c=set_a.symmetric_difference(set_b)
         set_c

Out[81]: {1, 2, 3, 4, 5, 8, 9, 10}
```

```
In [84]: set_a={1,2,3,4,5,6,7}
         set_b={6,7,8,9,10}
         set_d={34,45}
         set_c
         set_a.isdisjoint(set_c)  # it throws true if the values of set d are not there in set c

Out[84]: False
```

```
In [85]: set_d.isdisjoint(set_c) # it returns true if no items in set d is present in set y

Out[85]: True
```

```
In [91]: {1}.isdisjoint({1,2})

Out[91]: False
```

```
In [90]: {1}.isdisjoint({2,2})
```

```
Out[90]: True
```

```
In [92]: {1,2,3,4,5}.isdisjoint({1,8,9,7,9,0})
```

```
Out[92]: False
```

```
In [86]: set_a.issubset(set_c)  # it returns true all values of set a lies in set c
```

```
Out[86]: False
```

A little about Regular Expression

```
In [108]: import re
          a="ELECTRICAL WORLD BHUPENDRA was SINGH"

          #kuch bhi search krne ke liye use SEARCH and (kya chaheye aapko , kis mei se search krna chahate ho)

          b=re.search( "^ELECTRICAL.*SINGH$",a )

          if b != None:
              print(b)
          else:
              print("No match")
```

```
<re.Match object; span=(0, 36), match='ELECTRICAL WORLD BHUPENDRA was SINGH'>
```

Dictionaries

stores data in key value pair form just like a dictionary

```
In [120]: Dictionary_1 = { "Fav food" : ["banana","mango","Pinaple"] , "weight" : "99kg" , "Year of birth" : "1999"}
          Dictionary_1
```

```
Out[120]: {'Fav food': ['banana', 'mango', 'Pinaple'],
           'weight': '99kg',
           'Year of birth': '1999'}
```

```
In [121]: print(Dictionary_1["weight"])
```

```
99kg
```

```
In [122]: Dictionary_1["Year of birth"]
```

```
Out[122]: '1999'
```

```
In [123]: Dictionary_1["1999"]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Input In [123], in <cell line: 1>()
----> 1 Dictionary_1["1999"]

KeyError: '1999'
```

```
In [124]: type(Dictionary_1)
```

```
Out[124]: dict
```

```
In [125]: len(Dictionary_1)
```

```
Out[125]: 3
```

```
In [115]: Dictionary_1.keys()
```

```
Out[115]: dict_keys(['Fav food', 'weight', 'Year of birth'])
```

```
In [126]: Dictionary_1.values()

Out[126]: dict_values([['banana', 'mango', 'Pinaple'], '99kg', '1999'])

In [128]: Dictionary_1.items()

Out[128]: dict_items([('Fav food', ['banana', 'mango', 'Pinaple']), ('weight', '99kg'), ('Year of birth', '1999')])

In [129]: 'Fav food' in Dictionary_1

Out[129]: True

In [130]: '99kg' in Dictionary_1

Out[130]: False

In [131]: Dictionary_1.update({"key":"value"})
          Dictionary_1

Out[131]: {'Fav food': ['banana', 'mango', 'Pinaple'],
           'weight': '99kg',
           'Year of birth': '1999',
           'key': 'value'}

In [132]: dict_1=Dictionary_1
          dict_1

Out[132]: {'Fav food': ['banana', 'mango', 'Pinaple'],
           'weight': '99kg',
           'Year of birth': '1999',
           'key': 'value'}

In [133]: dict_1["key_1"]="Value_1"
          dict_1

Out[133]: {'Fav food': ['banana', 'mango', 'Pinaple'],
           'weight': '99kg',
           'Year of birth': '1999',
           'key': 'value',
           'key_1': 'Value_1'}

In [136]: dict_1.pop('key')

          dict_1

Out[136]: {'Fav food': ['banana', 'mango', 'Pinaple'],
           'weight': '99kg',
           'Year of birth': '1999',
           'key_1': 'Value_1'}

In [137]: dict_1.pop(2)
          dict_1

          ---------------------------------------------------------------------------
          KeyError                                  Traceback (most recent call last)
          Input In [137], in <cell line: 1>()
          ----> 1 dict_1.pop(2)
                3 dict_1

          KeyError: 2

In [138]: dict_1.popitem()
          dict_1

Out[138]: {'Fav food': ['banana', 'mango', 'Pinaple'],
           'weight': '99kg',
           'Year of birth': '1999'}

In [141]: del dict_1['weight']
          dict_1

Out[141]: {'Fav food': ['banana', 'mango', 'Pinaple'], 'Year of birth': '1999'}
```

In [142]:
```
del dict_1
dict_1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [142], in <cell line: 2>()
      1 del dict_1
----> 2 dict_1

NameError: name 'dict_1' is not defined
```

In [143]:
```
dict_2={'Fav food': ['banana', 'mango', 'Pinaple'],
 'weight': '99kg',
 'Year of birth': '1999'}
dict_2
```

Out[143]:
```
{'Fav food': ['banana', 'mango', 'Pinaple'],
 'weight': '99kg',
 'Year of birth': '1999'}
```

In [144]:
```
dict_2.clear()
dict_2
```

Out[144]:
```
{}
```

In [149]:
```
dect_3={'Fav food': ['banana', 'mango', 'Pinaple'],
 'weight': '99kg',
 'Year of birth': '1999'}
```

In [150]:
```
dict_4  =  dect_3
dict_4
```

Out[150]:
```
{'Fav food': ['banana', 'mango', 'Pinaple'],
 'weight': '99kg',
 'Year of birth': '1999'}
```

In [151]:
```
dict_5 =  dict_4.copy()
dict_5
```

Out[151]:
```
{'Fav food': ['banana', 'mango', 'Pinaple'],
 'weight': '99kg',
 'Year of birth': '1999'}
```

In [152]:
```
dict_6  =  dict(dict_5)  # dict ()
dict_6
```

Out[152]:
```
{'Fav food': ['banana', 'mango', 'Pinaple'],
 'weight': '99kg',
 'Year of birth': '1999'}
```

In [156]:
```
ab=dict({"a":"b"})
ab
```

Out[156]:
```
{'a': 'b'}
```

In [ ]: