How to Change/Replace an Element of a list?

In [0]:
```
x=[10,20,30,40,50]
x[1]=200
x
```

Out[0]:
```
[10, 200, 30, 40, 50]
```

In [2]:
```
#Syntax  list_name[index]=new_value
x=[10,20,30,40,50,20,30,30,20]
value=20
Count=0
for i in x:
    if i==value:
        print(i)
        Count=Count+1
        print(Count)
        break

print(x)
x[Count]=200
x
```

Out[2]:
```
20
1
[10, 20, 30, 40, 50, 20, 30, 30, 20]
[10, 200, 30, 40, 50, 20, 30, 30, 20]
```

Introduction to Tuple

In [ ]:
```
--> Tuple is exactly same as list but the only difference  between list and tuple is
    lists are mutable(Changeable) and tuples are immutable(Not changeable).
--> Tuple elements are fixed you can not perform any change in it(Immutability). In other words
    we can also say that tuple is a read only mode of list.
--> Indexing is very important is case of tuple(Ordered collection of element)
--> It stores dissimilar data like list.
--> Duplicates are allowed in case of Tuple
--> Parenthesis are used to represent a Tuple.
```

Creation of Tuple Object

In [20]:
```
#if you know the tuple element
x=(10,20,30,40)
type(x)
```

Out[20]:
```
tuple
```

In [21]:
```
#if you want to implement empty tuple
x=tuple()
type(x)
```

Out[21]:
```
tuple
```

In [22]:
```
x=()
type(x)
```

Out[22]:
```
tuple
```

In [24]:
```
#Creation of tuple with one element
x=(10,)
type(x)
```

Out[24]:
```
tuple
```

In [27]:
```
x=10,20,"str",40  #Tuple Packing
type(x)
print(x)
#Whatever elements are there all elements are packed together to form a tuple.
```

Out[27]:
```
(10, 20, 'str', 40)
```

In [ ]:
```
Note --> In case of tuple we don't have any method for insertion, deletion, updation because
tuple is immutable that means we cannot perform any change in the tuple object. We can only
access the content of the tuple.
```

Slicing and Indexing in Tuple

In [ ]:
```
How we can access the element of a tuple:
    1.Indexing
    2.Slicing
```

In [33]:
```
#Indexing  --> tuple_name[index_value]
#Slicing --> tuple_name[begin_index : end_index :step]
x=(10,20,30,40,50)
print(x[1])   #20
print(x[10])  #error
print(x[-1:-6:-1])#whole tuple
#print(x[20])  #error
print(x[2:8])  #30 40 50
print(x[::-1])  #50 40 30 20 10
print(x[::-4])  #50 10
print(x[10:9])  #empty tuple
print(x[0:0:0])  #error

20
(10, 20, 30, 40, 50)
(50, 40, 30, 20, 10)
(50, 10)
30
()
```

In [38]:
```
x=("Python","Hello",786,90,229)
print(x[1][1])#e
print(x[3][1])  #o
print(x[1][6])  #error
print(x[5][1])  #error

e
o
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [38], in <cell line: 6>()
      3 print(x[3][1])#o
      4 print(x[1][6])  #error
----> 5 print(x[5][1])  #error
      6 print(x[4][1])

TypeError: 'int' object is not subscriptable
```

In [43]:
```
x=(10,20,[20,30,(70,80)],[50,60])
x[2][2][1]
```

Out[43]:
```
80
```

Tuple vs Immutability

In [44]:
```
x=(10,20,30,40,50)
x[1]=200
x

Note --> Tuple are immutable,Immutable means we cannot perform any change in it If we will try to
perform any change then either you will get an error or a new object will be created
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [44], in <cell line: 2>()
      1 x=(10,20,30,40,50)
----> 2 x[1]=200
      3 x

TypeError: 'tuple' object does not support item assignment
```

Concatenation Operator

In [45]:
```
#Join two Tuple
x=(10,20,30)
y=(50,60,70)
z= x+y
z
```

Out[45]:
```
(10, 20, 30, 50, 60, 70)
```

Repitition Operator

In [50]:
```
#Repeat the Tuples
x=(10,20,30)
y=x*3
y
```

Out[50]:
```
(10, 20, 30, 10, 20, 30, 10, 20, 30)
```

Membership Operator

In [47]:
```
#Check weather a given element is member of the tuple or not
x=(10,20,30,40)
40 not in x
```

Out[47]:
```
True
```

Max and Min Function

In [ ]:
```
Max function --> It will give you the maximum element of the sequence.
Min Function --> It will give you the minimum element of the sequence.
Note --> Max and Min function will work with list as well as tuple
```

In [51]:
```
x=(10,20,30,40)
max(x)
```

Out[51]:
```
40
```

In [52]:
```
x=(10,20,30,40)
min(x)
```

Out[52]:
```
10
```

In [53]:
```
x=(10,20,30,40)
min(x)
```

Out[53]:
```
10
```

In [55]:
```
x=(10,20,30,40)
max(x)
```

Out[55]:
```
40
```

In [57]:
```
x=(10,20,30,40)
sorted(x)
x[-1:3:-1]
```

Out[57]:
```
(40, 30)
```

In [58]:
```
x=(10,20,30)
max(x)
```

Out[58]:
```
30
```

Python Problems

In [ ]:
```
Write a Python program which iterates the integers from 1 to 50.
For multiples of three print "Fizz" instead of the number and for the multiples of five
print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".
```

In [59]:
```
for i in range(1,51):
    if i%3==0:
        print("Fizz")
    elif i%5==0:
        print("Buzz")
    elif i%3==0 and i%5==0:
        print("FizzBuzz")
```

```
Fizz
Buzz
Fizz
Fizz
Buzz
Fizz
Buzz
Fizz
Fizz
Buzz
Fizz
Fizz
Buzz
Fizz
Buzz
Fizz
Fizz
Buzz
Fizz
Buzz
Fizz
Fizz
Buzz
```

Lists Vs Tuple

In [ ]:

List	Tuple
1.List is represented with the help of square brackets.	Tuple is represented with parenthesis
2.Lists are mutable.	Tuples are immutable
3.List is slower than Tuple	Tuple is Faster
4.List consumes more memory	Tuple consumes less memory
5.If you want to perform any updation like deletion, insertion etc then you should use list	Tuple is useful to access the element

Introduction to Dictionary

In [ ]:
```
Dictionary --> if you want to represent a group of element in the form of keys and values then
               we should use dictionary.

name:"Pratyush"
batch:DS290922A
college :Amity
```

In [ ]:
```
Properties of Dictionaries:
    1.keys cannot be duplicate if you are using duplicate keys then only last one will be considered
    2.Dissimilar objects are allowed for keys as well as values.
    3.Indexing is not possible(unordered collection of data)
    4.Dictionaries are mutable(changeable)
    5.Indexing and slicing concept is not applicable in case of dict and set.
    6.Hetable objects can be used as a keys
    7.Curly braces are used to represent a dictionary and each and every key value pair is
      seprerated by colon.
```

Creation of dictionary object

In [ ]:
```
Empty dictionary:
    d={}
    d=dict()

If you know the element:
    d={"name":"Pratyush","class":9D}
```

How to add entry in dictionary

In [60]:
```
d={}
d["name"]="Pratyush"
d["batch"]="DS2909022"
d
```

Out[60]:
```
{'name': 'Pratyush', 'batch': 'DS2909022'}
```

How to Access Elements from a Dictionary

In [ ]:
```
We can only access the elements of a dictionary with the help of keys.
Syntax:
    dictname[key]
```

In [3]:
```
d={1:2,3:4,5:6}
d[3]
```

Out[3]:
```
4
```

How to Update the Value of a dictionary

In [ ]:
```
We can update the value in the dictionary with the help of keys.
Syntax:
    dict_name[key]=updated_value
```

In [70]:
```
d={"name":"Pratyush","City":"Delhi","Mobile_no":9721890876,"Education":"M.Tech"}
print(d)
d["Batch"]="DS290922A"
print(d)
d["name"]="Noor"
d["Mobile_no"]=9000000
print(d)
```

```
{'name': 'Pratyush', 'City': 'Delhi', 'Mobile_no': 9721890876, 'Education': 'M.Tech'}
{'name': 'Pratyush', 'City': 'Delhi', 'Mobile_no': 9721890876, 'Education': 'M.Tech', 'Batch': 'DS2909022A'}
{'name': 'Noor', 'City': 'Delhi', 'Mobile_no': 9000000, 'Education': 'M.Tech', 'Batch': 'DS2909224'}
```

In [ ]:
```
Note --> If you want to access any value of the dictionary you can access with the help of key. if the key is
not present you will get the key error
```

How to delete a key from the dictionary

In [ ]:
```
We can delete the key value pair from the dictionary with the help of del keyword.

If we want to delete a key value pair then
Syntax:
    del  dict_name[key]

If we want to delete whole dictionary then
Syntax:
    del dict_name

If we want to delete the all content of a dictionary then
Syntax:
    dict_name.clear()
```

del Keyword

In [ ]:
```
d={"name":"Pratyush","City":"Delhi","Mobile_no":9721890876,"Education":"M.Tech"}
print(d)
del d["City"]
print(d)
d["State"]="Uttar Pradesh"
print(d)
```

```
{'name': 'Pratyush', 'City': 'Delhi', 'Mobile_no': 9721890876, 'Education': 'M.Tech'}
{'name': 'Pratyush', 'Mobile_no': 9721890876, 'Education': 'M.Tech'}
{'name': 'Pratyush', 'Mobile_no': 9721890876, 'Education': 'M.Tech', 'State': 'Uttar Pradesh'}
```

clear function

In [ ]:
```
clear function --> If we want to delete all key value pair from the dictionary then we should use clear function.
```

In [79]:
```
d={"name":"Pratyush","City":"Delhi","Mobile_no":9721890876,"Education":"M.Tech"}
print(d)
d.clear()
print(d)
```

```
{'name': 'Pratyush', 'City': 'Delhi', 'Mobile_no': 9721890876, 'Education': 'M.Tech'}
{}
```

In [ ]: