Dictonaries

It works on the concept of set Unique data

Keys,Values

- **Key** unique Identifier for a value
- **Value** is data that can be acessed with a key

In [16]:

```
d1={'Name':'Anil Peter','Role':'Trainer cum Developer','Organisation':'APSSDC'}
d1 # returns entire Dictonary
d1['Name'] # We can accessing the value only by Keys
d1.keys()  # return list of keys
d1.values() # return list of values
d1.items()  # returns the list of tuples of Keys and Values
d1['Brother']='Raju' # Updationg for a value
d1.pop("Organisation")
d1
```

Out[16]:

```
{'Name': 'Anil Peter', 'Role': 'Trainer cum Developer', 'Brother': 'Raju'}
```

In [14]:

```
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)
 |
 |  Methods defined here:
 |
 |  __contains__(self, key, /)
 |      True if the dictionary has the specified key, else False.
 |
 |  __delitem__(self, key, /)
 |      Delete self[key].
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(...)
 |      x.__getitem__(y) <==> x[y]
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setitem__(self, key, value, /)
 |      Set self[key] to value.
 |
 |  __sizeof__(...)
 |      D.__sizeof__() -> size of D in memory, in bytes
 |  
 |  clear(...)
 |      D.clear() -> None.  Remove all items from D.
 |  
 |  copy(...)
 |      D.copy() -> a shallow copy of D
 |  
 |  get(self, key, default=None, /)
 |      Return the value for key if key is in the dictionary, else defaul
t.
 |  
 |  items(...)
 |      D.items() -> a set-like object providing a view on D's items
 |  
 |  keys(...)
 |      D.keys() -> a set-like object providing a view on D's keys
 |  
 |  pop(...)
 |      D.pop(k[,d]) -> v, remove specified key and return the correspondi
ng value.
 |      If key is not found, d is returned if given, otherwise KeyError is
raised
 |  
 |  popitem(...)
 |      D.popitem() -> (k, v), remove and return some (key, value) pair as
a
 |      2-tuple; but raise KeyError if D is empty.
 |  
 |  setdefault(self, key, default=None, /)
 |      Insert key with a value of default if key is not in the dictionar
y.
 |      
 |      Return the value for key if key is in the dictionary, else defaul
t.
 |  
 |  update(...)
 |      D.update([E, ]**F) -> None.  Update D from dict/iterable E and F.
 |      If E is present and has a .keys() method, then does:  for k in E:
D[k] = E[k]
 |      If E is present and lacks a .keys() method, then does:  for k, v i
n E: D[k] = v
 |      In either case, this is followed by: for k in F:  D[k] = F[k]
 |  
 |  values(...)
 |      D.values() -> an object providing a view on D's values
 |  
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |  
 |  fromkeys(iterable, value=None, /) from builtins.type
 |      Create a new dictionary with keys from iterable and values set to
value.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signa
ture.
 |  
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |  
 |  __hash__ = None
```

Contacts Application

- Add Contact
- Search for contacts
- List all contacts
- Modify contact
- Remove contact

In [5]:

```
Contacts={}
```

In [25]:

```
def AddContact(Name,Contact_Number):
    if Name not in Contacts:
        Contacts[Name]=Contact_Number
        print("Contact has been added")
    else:
        print("Contact with ",Name,"Already Exists")
    return

def SearchContact(Name):
    print(Name,':',Contacts[Name])
    return

def ModifyContact(Name,Contact_Number):
    if Name in Contacts:
        Contacts[Name]=Contact_Number
        print("Contact has been Modified")
    else:
        print("Contact with ",Name,"Doesn't Exists")
    return

def ListAllContacts():
    for keys,values in Contacts.items():
        print(keys,':',values)
    return

def DeleteContacts(Name):
    if Name in Contacts:
        Contacts.pop(Name)
        print("Contact has been sucessfully deleted")
    else:
        print("Contact with ",Name,"Doesn't Exists")
    return
```

In [28]:

```
x=int(input("Enter 1 to Add a contact \nEnter 2 to Delete a Contact \nEnter 3 to Modify
a Contact \n"
            "Enter 4 to Search a contact \nEnter 5 to Show all Contacts \n"))
if x==1:
    Name=input("Enter contact name: ")
    Contact_Number=int(input("Enter Contact number: "))
    AddContact(Name,Contact_Number)
elif x==2:
    Name=input("Enter Contact Name to be Deleted: ")
    DeleteContacts(Name)
elif x==3:
    Name=input("Enter Contact Name to be Modified: ")
    Contact_Number=int(input("Enter Contact Number to be modified: "))
    ModifyContact(Name,Contact_Number)
elif x==4:
    Name=input("Enter Name of the contact do you need: ")
    SearchContact(Name)
elif x==5:
    ListAllContacts()
```

```
Enter 1 to Add a contact
Enter 2 to Delete a Contact
Enter 3 to Modify a Contact
Enter 4 to Search a contact
Enter 5 to Show all Contacts
1
Enter contact name: Anil Peter
Enter Contact number: 8886785229
Contact has been added
```