In [2]:

```
# coding: utf-8

# In[41]:

######Importing the required Libraries######
try:
    import json
    import os
    import time
except:
    get_ipython().system('pip install json')
    get_ipython().system('pip install os ')
    get_ipython().system('pip install time')
    import json
    import os
    import time



def save(dic,dic2,time_dic,t_allowed_dic):
    """Funtion used to save the dictionary(Key-value Pairs) into a JSON file"""

    dic.update(dic2)
    out_file = open(path +"/pairs.json", "w")
    json.dump( dic,out_file)
    out_file.close()

    out_file = open(path +"/ttlpairs.json", "w")
    json.dump( dic2,out_file)
    out_file.close()

    out_file = open(path +"/start_time.json", "w")
    json.dump( time_dic,out_file)
    out_file.close()

    out_file = open(path +"/time_allowed.json", "w")
    json.dump( t_allowed_dic,out_file)
    out_file.close()

def getPath(c):
    """Funtion used to set directory to the input/ default directory"""
    if (c=="y"):
        path=input("\nEnter destination to store the data \n")
    else:
        path = os.getcwd()
    path.replace('\\', '/')
    return (path)


def getKey():

    """Funtion used to obtain key input from user"""

    key = str(input("\nEnter Key \n"))
    return(key)

def getValue():
```

```
    """Funtion used to obtain value input from user"""

    value=input("\nEnter Value \n")
    return(value)
def createKeyValuePair(dic,dic2,time_dic,t_allowed_dic):

    """Funtion used to create key value pair"""

    key=getKey()
    value=getValue()

    #check if key is present
    if (key in dic):
        #if yes : error already present
        print("\nError:Key already present.Try with a different key \n.")
    #if no :
    else:
        #checkkey
        if(checkKey(key)):
            #checkValue
            if(checkValue(value)):
                #add to dictionary
                #ask for time to live property
                ttlsave = input("\nDo you want to have ttl property? y for yes else any cha
                if (ttlsave=="y"):
                    #t_allowed: Time allowed for the key value pair before it gets deleted
                    t_allowed = int(input("Enter seconds \n"))
                    t_allowed_dic[key]=t_allowed
                    start_time = time.time()
                    #time_dic stores the start time of all ttl key-value pairs
                    time_dic[key]=start_time
                    dic2[key] = value
                #ttl:no
                else:
                    dic[key] = value
                save(dic,dic2,time_dic,t_allowed_dic)
            #value validation error
            else:
                print("\nError: Value is not a JSON Object or size is not less than 16KB \n
        #key validation error
        else:
            print("\nError: The key has more than 32 characters, Please try again with less

def readValue(dic,dic2,time_dic,t_allowed_dic):
    """function to retrieve value from the data store given a key"""

    key=getKey()

    #check if key is present and was of ttl
    if(key in time_dic):
        end = time.time()
        elapsed = int(end-time_dic[key])
        #print(elapsed,t_allowed_dic[key]):used for debugging
        if(elapsed>t_allowed_dic[key]):
            print("Cannot access : key expired")
            del time_dic[key]
            del dic[key]
            del dic2[key]
            del t_allowed_dic[key]
        else:
            print(dic2[key])
```

```
    #if key was not ttl and is present
    elif(key in dic):
        #if yes : #Write code to retrieve and return the value obtained
        print(dic[key]+ '    is the value stored for key  ' + key)

    #key not present
    else:
        print("\n Error: The key is not present in the data store.\n")

def deleteKey(dic,dic2,time_dic,t_allowed_dic):

    """Funtion to delete a key value pair """

    key=getKey()

    #check if key is present
    if (key in dic):
        #if yes : Delete
        del dic[key]
        #if no : Error : Not present

    elif(key in time_dic):
        end = time.time()
        elapsed = int(end-time_dic[key])
        if(elapsed>t_allowed_dic[key]):
            print("\nCannot access : key expired")
            del time_dic[key]

        else:
            print("\nKey Deleted")
            del time_dic[key]

    else:
        print("\nError: The key is not present in the data store. \n")
    save(dic,dic2,time_dic,t_allowed_dic)

def checkKey(key):

    """Funtion to check the requirements for a key"""

    #write code to check the requirements for key
    if(len(key)<=32):
        return True
        #good to go
    else:
        return False

def validateJSON(jsonData):
    """Funtion to check the requirements of json object value"""

    try:
        json.loads(jsonData)
    except ValueError as err:
        return False
    return True

def checkValue(value):

    """Funtion to check the requirement of value"""
```

```
    if(validateJSON(value)):
        #check if size is less than 16KB
        if(len(value)<128000):
            return True
        else:
            return False
    else:
        return False

def main():

    """Main funtion which sets directory and calls other funtions

        dic: Stores all key value pairs
        dic2 : stores all tll based key-value pairs
        time_dic: Stores the start time of all pairs
        t_allowed_dic : Stores the time allowed before deletion for all pairs."""
    #ask for directory
    c = str(input("\nDo you want to set a working directory? if yes :enter y as input \n"))
    global path
    path=getPath(c)

    #retrieve files at direcctory
    with open('pairs.json') as json_file:
        dic = json.load(json_file)
    with open('ttlpairs.json') as json_file:
         dic2= json.load(json_file)
    with open('start_time.json') as json_file:
         time_dic= json.load(json_file)
    with open('time_allowed.json') as json_file:
         t_allowed_dic= json.load(json_file)

    #print(dic) : used for debugging

    while(True):
        #looping until user enters end
        functions = {"create":createKeyValuePair,"read":readValue,"delete":deleteKey}
        choice = input("Enter choice create/read/delete/end \n")
        if(choice=="end"):
            print("The program has ended \n")
            break
        elif(choice=="create" or choice=="read" or choice=="delete"):
            functions[choice](dic,dic2,time_dic,t_allowed_dic)
        else:
            print("Please enter valid choice \n \n")
```

In [5]:

```
main()
#test1 done:
#set directory
# Created a pair (not -ttl)
# entered valid key,value
# Read operation
# delete operation
# reading after delete
# end operation
```

```
Do you want to set a working directory? if yes :enter y as input
y

Enter destination to store the data
C:\Users\Charu\Desktop\keyvalue_datastore
Enter choice create/read/delete/end
create

Enter Key
test1

Enter Value
{ "name":"John", "age":30, "car":null }

Do you want to have ttl property? y for yes else any character
n
Enter choice create/read/delete/end
read

Enter Key
test1
{ "name":"John", "age":30, "car":null }    is the value stored for key  test
1
Enter choice create/read/delete/end
delete

Enter Key
test1
Enter choice create/read/delete/end
read

Enter Key
test1

 Error: The key is not present in the data store.

Enter choice create/read/delete/end
end
The program has ended
```

All of test1 ran successfully.

In [6]:

```
main()
#test 2
# create a ttl pair of 10 sec
# read before 10 sec
# read after 10 sec
# create a pair with valid key but invalid value
# create pair with invalid key but valid value
```

```
Do you want to set a working directory? if yes :enter y as input
y

Enter destination to store the data
C:\Users\Charu\Desktop\keyvalue_datastore
Enter choice create/read/delete/end
create

Enter Key
test2

Enter Value
{ "name":"John", "age":30, "car":null }

Do you want to have ttl property? y for yes else any character
y
Enter seconds
10
Enter choice create/read/delete/end
read

Enter Key
test2
{ "name":"John", "age":30, "car":null }
Enter choice create/read/delete/end
read

Enter Key
test2
Cannot access : key expired
Enter choice create/read/delete/end
create

Enter Key
test2

Enter Value
"json":value

Error: Value is not a JSON Object or size is not less than 16KB

Enter choice create/read/delete/end
create

Enter Key
12345612345612345612345612345612345

Enter Value
{ "name":"John", "age":30, "car":null }
```

```
Error: The key has more than 32 characters, Please try again with lesser c
haracters.

Enter choice create/read/delete/end
end
The program has ended
```

All of test2 ran successfully.