**Example of converting json to python and python to json**

```
import json
def main():
    x=65  # int
    y=1.5 # float
    z=[10,20,30,40,50] # list
    d={1:10,2:20,3:30,4:40,5:50} #dictionary
    j1=json.dumps(x) # json string
    j2=json.dumps(y) # json string
    j3=json.dumps(z) # json string
    j4=json.dumps(d) # json string
    print(j1,j2,j3,j4)
    print(type(j1),type(j2),type(j3),type(j4))
    a=json.loads(j1) # json string to python object
    b=json.loads(j2) # json string to python object
    c=json.loads(j3)
    d=json.loads(j4)
    print(a,b,c,d)
    print(type(a),type(b),type(c),type(d))

main()
```

json.dump(*obj*, *fp) : convert python object into json string and write inside file*
json.load(fp): load json string from file and return python object

**# writing python objects into json file**

```
import json
def main():
    student_dict={1:['naresh','python'],
                  2:['suresh','java'],
                  3:['kishore','c++']}
    with open("student_data.json","w") as f:
        json.dump(student_dict,f)

main()
```

**Reading data from json file**

```
import json
def main():
    with open("student_data.json","r") as f:
        student_dict=json.load(f)
        print(student_dict)

main()
```

**Output:**

```
{'1': ['naresh', 'python'], '2': ['suresh', 'java'], '3': ['kishore', 'c++']}
>>>
```

**Pickle module**

The pickle module implements binary protocols for serializing and de-serializing a Python object structure. *“Pickling”* is the process whereby a Python object hierarchy is converted into a byte stream, and *“unpickling”* is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy. Pickling (and unpickling) is alternatively known as “serialization”, “marshalling,” or “flattening”; however, to avoid confusion, the terms used here are “pickling” and “unpickling”.

**pickle.dump(*obj, file)***
Write the pickled representation of the object *obj* to the open file object *file*