SERIALIZATION

PICKLE AND MARSHAL

Data Persistence

- The term data persistence means it continues to exist even after the application has ended.

- Data stored in a non-volatile storage medium such as, a disk file is a persistent data storage example.

- Python also provides a standard way of interacting with relational databases.

Binary Mode

- By default, read/write operation on a file object are performed on text string data.

- Non-string files, such as media (mp3), executables (exe), pictures (jpg) etc., a ‘b’ prefix is added to read/write operations.

```
f=open('test.bin', 'wb’)
data=b"Hello World"
f.write(data)
f.close()
```

Pickle

- Pickling is the process whereby a Python object hierarchy is converted into a byte stream to be written to a file, also known as serialization.

- Unpickling is the reverse operation, whereby a byte stream is converted back into a working Python object hierarchy.

Serialization

- Pickle is used for serializing and de-serializing Python object structures.

- Serialization refers to the process of converting an object in memory to a byte stream.

- The byte stream can be written to disk or broadcast to a network.

- The byte stream can de-serialized back to a Python object. this character stream can then be retrieved and de-serialized back to a Python object.

Usages

- Pickle is useful for applications where you need persistency in your data.
- A program's state data can be saved to disk, so it can continue to be worked on.
- It can also be used to send data over a Transmission Control Protocol (TCP) or socket connection, or to store python objects in a database.
- Pickle is very useful for when working with machine learning algorithms, where the state is saved to make new predictions, without having to rewrite or train the model all over again.

Serialize - Dump

```
import pickle

exampleObj = {'Python':3,’Java’:8,'Windows':10}

fileObj = open('data.obj', 'wb')
pickle.dump(exampleObj,fileObj)
fileObj.close()
```

De-Serialize Load

```
import pickle

fileObj = open('data.obj', 'rb')
exampleObj = pickle.load(fileObj)
fileObj.close()
print(exampleObj)
```

Marshal

- Object serialization features of marshal module in Python’s standard library are similar to pickle module.

- It is used by Python itself for Python’s internal object serialization to support read/write operations.

- It is also used on compiled versions of Python modules (.pyc files).

Pickle vs Marshal

- The Pickle and Marshal and serialization are similar in the context but semantically different as a matter of intent.

- Marshalling is used to pass an Object to remote objects (RMI). In Marshalling an Object is serialized (member data is generally serialized) **+** Codebase.

- When an object is Pickled, only the member data within that object is written to the byte stream; not the code that actually implements the object.