Experiment No.6
Serialization in python using Pickel
Date of Performance:
Date of Submission:

Experiment No. 6

Aim : Serialization in python using Pickel

Objective: To introduce basic concept of Pickel module

Theory :

- What is Serialization?
- Serialization is the process of converting a Python object into a byte stream that can be stored in a file or transmitted over a network.
- What is Pickle?
- Pickle is a Python module used for serializing and deserializing Python objects.
- Why Pickle?
- Pickle provides a convenient way to save Python objects to disk and load them back into memory later.
- How to use Pickle?
- The pickle module provides two main functions: dump() for serialization and load() for deserialization.

1) **pickle.dump(obj, file)** :
   - The **pickle.dump()** function is used to serialize a Python object **obj** and write it to a file specified by the file object **file** .
   - This function takes two parameters:
   - **obj** : The Python object to be serialized.
   - **file** : A file object opened in binary write mode ('wb') where the serialized data will be written.

2) **pickle.load(file)** :
   - The **pickle.load()** function is used to deserialize data from a file specified by the file object **file** and reconstruct the original Python object.
   - This function takes one parameter:

- **file** : A file object opened in binary read mode ('rb') from which the serialized data will be read and deserialized.

Code:-

```
import pickle


class Employee:     def __init__(self,

name, age, ID, salary):

        self.name = name

self.age = age

self.ID = ID

self.salary = salary


    def __str__(self):

        return f'Employee({self.name}, {self.age}, {self.ID}, {self.salary})'


emp = Employee('Jidnyasa Naik', 19, '0123456', 100000)
```

with open('employee.pickle', 'wb') as file:

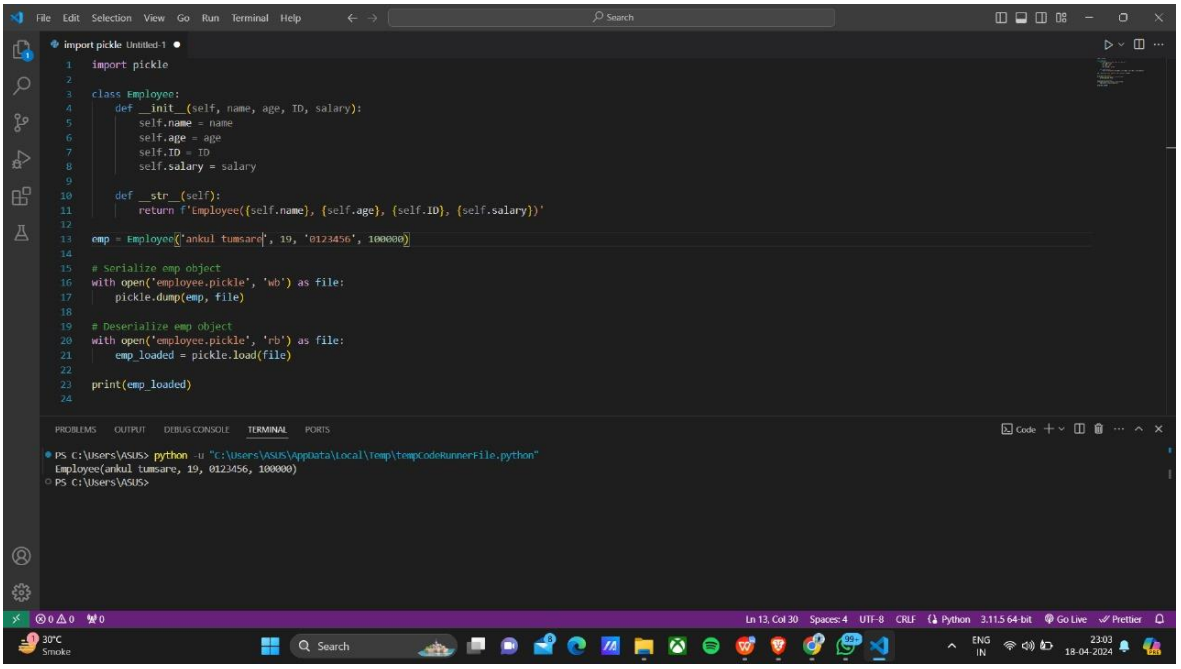
pickle.dump(emp, file)

with open('employee.pickle', 'rb') as file:

emp_loaded = pickle.load(file)

print(emp_loaded)

Output:-



Conclusion:

Pickle is a powerful tool for serializing Python objects, offering simplicity and flexibility. However, it's essential to be aware of its limitations and potential security risks, especially in environments where data integrity and compatibility are critical. Always consider the specific requirements of your application when choosing a serialization method.