**Experiment No. 5**

**Title: Persistent Object Storage using Shelves**

**Batch: B1 Roll No: 1914078 Experiment No.:5**

### Aim: Persistent Storage of arbitrary python objects using shelve

**Resources needed**: Python IDE

### Theory:

### A “shelf” is a persistent, dictionary-like object. The difference with “dbm” databases is that the values (not the keys!) in a shelf can be essentially arbitrary Python objects — anything that the [pickle](https://docs.python.org/3/library/pickle.html#module-pickle) module can handle. This includes most class instances, recursive data types, and objects containing lots of shared sub-objects. The keys are ordinary strings.

The shelve module defines three classes as follows −

|  |  |
| --- | --- |
| Sr.No | Shelve Module & Description |
| 1 | Shelf  This is the base class for shelf implementations. It is initialized with dict-like object. |
| 2 | BsdDbShelf  This is a subclass of Shelf class. The dict object passed to its constructor must support first(), next(), previous(), last() and set\_location() methods. |
| 3 | DbfilenameShelf  This is also a subclass of Shelf but accepts a filename as parameter to its constructor rather than dict object. |

The open() function defined in shelve module which return a DbfilenameShelf object.

**open(**filename, flag='c', protocol=None, writeback=False)

Open a persistent dictionary. The filename specified is the base filename for the underlying database. As a side-effect, an extension may be added to the filename and more than one file may be created. By default, the underlying database file is opened for reading and writing. The optional flag parameter has the same interpretation as the flag parameter of [dbm.open()](https://docs.python.org/3/library/dbm.html#dbm.open).By default, version 3 pickles are used to serialize values. The version of the pickle protocol can be specified with the protocol parameter. writeback parameter by default is false. If set to true, the accessed entries are cached. Every access calls sync() and close() operations hence process may be slow.

**Example:**

import shelve

s = shelve.open('test\_shelf')

try:

s['key1'] = { 'int': 10, 'float':9.5, 'string':'Sample data' }

finally:

s.close()

The Shelf object has following methods available −

|  |  |
| --- | --- |
| **Sr.No.** | **Methods & Description** |
| 1 | **close()**  synchronise and close persistent dict object. |
| 2 | **sync()**  Write back all entries in the cache if shelf was opened with writeback set to True. |
| 3 | **get()**  returns value associated with key |
| 4 | **items()**  list of tuples – each tuple is key value pair |
| 5 | **keys()**  list of shelf keys |
| 6 | **pop()**  remove specified key and return the corresponding value. |
| 7 | **update()**  Update shelf from another dict/iterable |
| 8 | **values()**  list of shelf values |

### Activities:

1. Use shelve module to create Student database to store details of the students(at least 10 student details)
2. Perform read, update and delete operations on the Student Database.

### Result: (script and output)

### 

### Create

### 

### 

### Read

### 

### Update

### 

### Delete

# Delete

rollno = input("Enter roll no. of student to delete: ")

students = shelve.open("Devansh", writeback=True)

print(students.pop(rollno,None))

### 

### Outcomes: CO3:Demonstrate handling database with python

**Conclusion:** We learnt about persistent Storage of arbitrary python objects using shelve

**References:**

1. Yuxi Liu ; Python Machine Learning By Example: The easiest way to get into machine learning 1st Edition, Kindle Edition , Packt publishing Ltd , 1st edition 2017.
2. Martin C. Brown, The Complete Reference Paperback, Osborne, 2nd edition 2001
3. Frank Millstein, Data Analytics with Python: Data Analytics In Python Using Pandas, Copyright at Frank Millstein, 1st edition 2018
4. <https://docs.python.org/3/library/shelve.html>