**Experiment Number – 17**

**Title – Create a list of 10 elements. Write a program to write this list in binary file and then read it back to find out the smallest and largest value.**

**Theory-**

Python has tools for working with binary files. Binary files use strings of type bytes. This means when reading binary data from a file, an object of type bytes is returned. Binary File Handling is a process in which we create a file and store data in its original format. It means that if we store an integer value in a binary file, the value will be treated as an integer rather than text. Binary files are mainly used for storing records, just as we store records in a database. It makes it easier to access or modify the records easily.

The binary file is opened using the open() function, whose mode parameter contains the character **‘b’**.

**Binary File Opening Modes**

Below we have discussed the different types of binary file opening modes used to open a binary file in python.

|  |  |
| --- | --- |
| **Mode** | **Description** |
| rb | Open file in binary mode for reading only. The file pointer stands at the beginning of the file. |
| rb+ | Open file in binary mode for both reading and writing. The file pointer stands at the beginning of the file. |
| wb | Open file in binary mode for writing only. It creates the file if it does not exist. If the file exists, then it erases all the contents of the file. The file pointer stands at the beginning of the file. |
| wb+ | Open file in binary mode for both reading and writing. It creates the file if it does not exist. If the file exists, then it erases all the contents of the file. The file pointer stands at the beginning of the file. |
| ab | Open file in binary mode for appending data. Data is added to the end of the file. It creates the file if it does not exist. The file pointer stands at the end of the file. |
| ab+ | Open a file in binary mode for reading and appending data. Data is added to the end of the file. It creates the file if it does not exist. The file pointer stands at the end of the file. |

## Serialization or Pickling

Serialization or Pickling is the process in which python objects are converted into byte streams (sequence of bytes) so that they can be stored in a binary file or transferred over a network.

## Deserialization or Unpickling

Deserialization or Unpickling is the process in which byte streams are converted back into python objects. Deserialization produces a replica of the original object.

## The pickle Module

The **pickle** module provides methods for serialization and deserialization of python objects. There are four methods available in the pickle module, and they are:

* **dump()**
* **load()**
* **dumps()**
* **loads()**

### dump() Method

The **dump()** method converts the python objects into byte streams and writes them immediately into a binary file.

#### Syntax of dump() method

pickle.dump(obj,file)

### load() Method

The **load()** method reads the byte stream of one or more python objects from a binary file and reconstructs the replica of the original python object.

#### Syntax of load() method

obj=pickle.load(file)

### dumps() Method

The **dumps()** method only converts the python objects into a string of bytes and return them as byte object. This method itself does not write the byte object into a binary file. We must use the **write()** method of the file object to write it into a binary file.

This method is useful when we only want the serialization result of a python object so that we can transfer it over a network or other process.

#### Syntax of dumps() method

byteobj=pickle.dumps(obj)

### loads() Method

The **loads()** method reads the string of bytes and reconstructs the replica of the original python object.

#### Syntax of loads() method

obj=pickle.loads(byteobj)

Exercise –

1. Write a Python program to reverse the content of a file and store it in another file
2. Write a python Copy all the content of one file to another file in uppercase
3. Write a python program to Append content of one text file to another