**Introduction:**

First of all, we created JSON file structure to implement this file management system. We will do this using python as a programming language. We have attached two files i.e first is file.py and second is main.py

**Implementation in files.py:**

In this file we have created the main skeleton of JSON file that how data is represented in JSON file. The elements in files.py are id, names, size and chunks. The function **create\_f()** create the new file in JSON when this function calls from main.py.

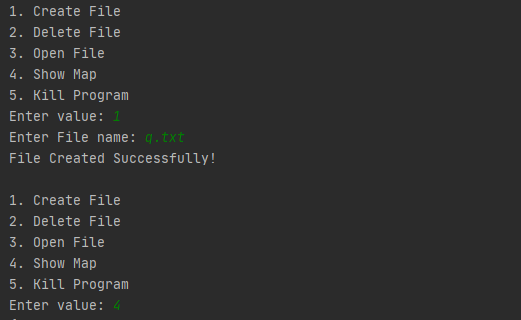
**Implementation in main.py:**

We import the JSON file structure. Then the implementation of the following function are as follows:

**file\_id\_assigner:** It gives the id to each file when it is created and update in file\_structure.json

**chunk\_id\_assigner:** It gives the chunk to each file when it is created and update in file\_structure.json

**create\_file:** This will create file in JSON structure, we use the flat (not hieratical) system for our file system. After this we have an option of write open that or create new file.

****

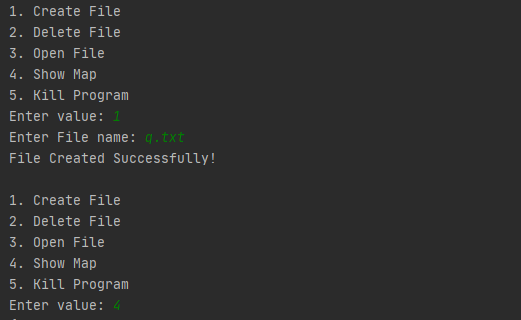
**delete\_file:** This will delete the already existing file

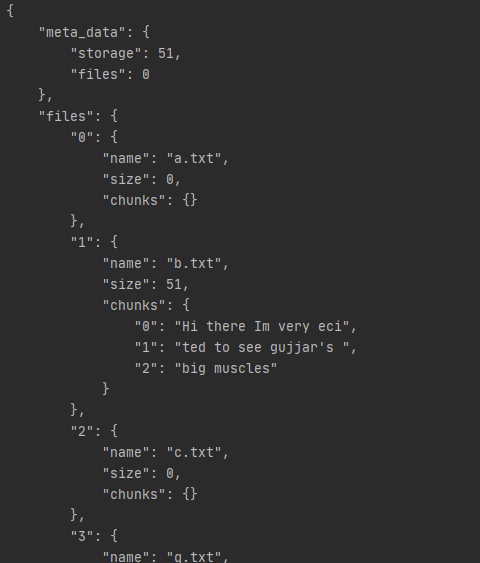
**open\_for\_write:** When we create a file, we have an option to open it and then write data onto it. If this file has already written data onto it, this function will append data at the last of the already existing file.

**open\_for\_read:** This function makes us to look the contents of the file.

**open\_file:** To open a file we use this function. After opening it, we have an option to read, write or close the file.

**show\_map:** This function will give all the contents of JSON file. It displays all the files, their storage, chunks and what’s the data written in that file.

****

****

**dump\_JSON:** This will update the data in JSON file as open\_for\_write function is implemented.

**close\_file:** This will close the already open file

|  |
| --- |
| **Implementation in file\_structure.json:**  In this file, it displays all of the contents that are created with their size, chunks, file id, it’s name, extension of the file name, data written onto the file. We use the chunks of 20 bytes to display data. If data exceeds up 20 bytes, it will create new chunk to store data. It also shows the collective sizes of data in meta\_data. In this example, we have created 4 files namely b.txt, c.txt, q.txt, w.txt respectively. We store some of the data in b.txt. . c.txt,q.txt,w.txt are just created but no data are written onto it, so, their size is 0.  **File\_structure.json** |