Homework 2- Design

1. **Hierarchical FS Implementation (**hierarchicalFS.py)

   The hierarchical file system has been implemented using nested Directories.

   For example, say T is a dictionary.

   T= {}, then the hierarchical file system is represented in the below format:

   T ={'/': {' a'=1 , 'dir1': {'b'=1 , 'dir2':{c=1, 'hello.txt'={}}}}

   Here '/', 'dir1' and 'dir2' are directory names. Here ' dir1' is the child of '/' and 'dir2' is the child of 'dir1'. And 'hello.txt' is a file inside 'dir2'

   This is the implementation that has been followed throughout.

   To access the children, we can traverse the dict tree as follows:

   T['/']['dir1']['dir2']['hello.txt']

2. **Remote File system (Hierarchical)**

   Here also the same nested directory structure has been followed.

   The self.files is initialized with the root directory '/' and it has been uploaded to the server. Now this will act as the parent to all the other directories and files within it.

   The connection to the server is being made via an XMLRPC Call. I have tested the same using localhost.

   Every time I need to modify or retrieve data from the file systems, the tree as a whole (starting from the root directory) is being retrieved, and after making the necessary modifications to it in the client, is then re-uploaded to the server via RPC.

   For data, a flat dictionary is implemented with keys corresponding to the path of the file.

   Here also, a dictionary 'data' is used, which is uploaded to the server. Every time a modification is made to the data dictionary, the modified dictionary is re-uploaded to the server.

   Test cases performed:

   - Creating and removing directories
   - Creating and removing files
   - Appending data into the file.
   - Symbolic link creation, deletion
   - Listing the directories

3. **Test-client.py**

   This file implements the client interface which communicates to the simpleht.py server. It does the same via an XMLRPC call.

   Here we are storing and retrieving integers, strings, lists and dictionaries and reading and writing files.