Final project

Due: May 14 at 11:59 PM

This is a simple project. You are to implement a *string store*. The system should support the following functions:

```
void InitStore();
void ResetStringStore();
void AddString( char *name, char *string);
char *GetString( char *name);
void CommitChanges();
```

`InitStore` is called before call to any other functions. This gives you the opportunity to initialize your data structures. `ResetStringStore` resets the total system so that there are no strings in the system. `AddString` adds a string with the given name to the system. `GetString` returns the last string with that name that was added to the system. If no such string exists, an empty string is returned. Note that there is no function to delete string because you can delete a string by adding an empty string. `CommitChanges` will cause all the changes to be committed to the disk.
Your implementation should use the UNIX file system to implement the store. Create all the files you need in "./StoreFiles/" directory.
For this assignment, performance is important along with correctness. Disk operations are expensive and you should try to minimize the amount of disk access by keeping a cache in memory and and writing to disk only when necessary. The best solutions may use compression on the strings to further increase performance. That is they will keep strings as compressed objects. Also you should be careful in how you organize the writing to the file system. Work out a scheme that does large writes and avoids writing small objects.
You should describe the various things you tried to improve the performance along with explanation as to why it did or didn't help. You should try to optimize both the time and the disk space used. Explain the design tradeoffs made in your implementation.

Testing and submission

Test your functions any way you want. However, your files must compile with the "driver" routine driver.c which will be used to test your functions. The perl script "runTests" runs the driver on several data sets which are in the "/u/cs318/final/Data/" directory. "runTests" generates the transcript of the run in two files "out" and "result" for each of the data sets along with the amount of time and space used. You don't need to know the formats of the data files. You should use the files in "/u/cs318/final/" as a starting point. Submit all the code, makefile and a README file on phoenix. The README file should contain the results of running your code and your conclusions. Also, *coding style counts.* So, please be neat, use mnemonic variable names and comment tricky code.