EEL-4736/EEL-5737 Principles of Computer System Design

Homework #3

Assigned: 9/21/2017; Due 5 pm on 10/06/2017 – To be done individually

Part A

a) Problem set 1 (Bigger files)
b) Problem set 4 (EZPark)
c) Problem set 9 (Ben's Kernel)

Part B

In HW#2, you split the contents of file in the blocks of fixed size. However, your design was still limited to a flat in-memory file system that supported only a single context (i.e., single directory '/'). Though the design of the original flat file system is pretty straightforward, it is very restrictive since you cannot create files with same name as it will cause naming conflicts. As discussed in class, practical UNIX-like file systems support hierarchical namespace to organize the files into directories.

In this assignment, you will extend the previously implemented flat file system with blocks from HW#2 (memoryBlockFS.py) to support the hierarchical namespace.

To proceed with this task, first you must familiarize yourself with the working and data structures used in the flat filesystem, and built-in data types in Python – specially, the Python *dictionary (dict).* A dictionary in Python provides a flexible primitive to create a context where we can store and look-up name-value bindings. It is extremely important for you to understand how two dictionaries are used for storage of metadata and data of files/directories. Also, observe how the methods implementing the file system functionality interact with these dictionaries to store, retrieve and modify its content. Your goal for this assignment is not only to store file contents, but also to support a hierarchical namespace.

The solution for this problem is not complex, but requires that you consider carefully how to store and lookup objects that includes contexts. Think about the answers to the following questions before you design your solution:

1. What is a file system, and how is it structured in terms of files and directories? This assignment does not require you to worry about low level details (e.g. inodes) but you need to consider the logical organization of files and directories.
2. How is a file structurally different from a directory?
3. How does a File system weaves together a skeleton holding all the files and directories, how does it connects them?
4. Which data structure I can use to build this skeleton?

5. How can I represent a File or Directory as a data structure?
6. How can I organize File/Directory data structures using a FS data structure.?
7. How will I travel the FS data structure to get to a desired File/Directory data structure?
8. How can I do the above (7) while parsing a hierarchical namespace like /dir1/dir2/hello.txt ?

If you have convincing answers to all of the above questions you can start working on the solution. Once you implement the hierarchical file system, make sure that you carry out all regular file system operations. It is also the part of assignments goal that you should come up with your own approach to test the functionality of your implementation. Therefore, no testcases will be provided. Your design will be graded on the ability to support complex file system operations, so it is to your benefit to test thoroughly.

Submission guidelines:

Turn in through Canvas following three files (attach individually):

1. homework3partA.pdf – Solution to the questions in Part A
2. multilevelBlockFS.py – Your Python code that implements multilevel file system.
3. homework3design.pdf – PDF describing the design of your implementation and tests you have conducted to check the functionality of your code.

Make sure your Python code is well commented and tested before submission. Assignment will be graded on the basis of design and functionality. There might be several ways to implement this assignment, choose one which is most feasible, concise and modular allowing you to have minimum changes in the existing code when you extend its functionality. Copy and the paste the python code (multilevelBlockFS.py) at the end of your homework3design.pdf file.