Homework 3

1. [2] Suppose a system uses 1KB blocks and 16-bit addresses. What is the largest possible file size for this file system for the following inode organizations?

   (a) The inode contains 12 pointers directly to data blocks.

   (b) The inode contains 12 pointers directly to data blocks and one indirect block. Assume the indirect data block uses all 1024 bytes to store pointer to data blocks.

   (c) The inode contains 12 direct pointers to data blocks, one indirect data block, and one doubly indirect data block.

2. *Fun with* `dup`: You are to write a program that writes to and reads from a local database. The database program is implemented by a separate program called, say `picodbd` (for pico DB daemon). Once `picodbd` runs, it reads requests from standard-in (`cin` in C++ parlance) and writes results to standard-out (`cout` in C++ parlance). Your program is supposed to run `picodbd` in a separate process (using `fork` and `exec`, similar ot MP2). How will your program communicate with the DB daemon? Use the following skeleton to connect to connect your program to the DB daemon.

   ```
   ...
   <set up the communication channels>
   ...
   if (fork() == 0) { // child
       ...
       <use dup() to link communication channels to daemon>
       ...
       <start daemon with exec()>
   }
   ...
   ```

   Hint: The easiest way is to use two unnamed pipes (one for each direction) and connect them to standard in and standard out of the DB daemon through the judicious use of `dup2()`.

3. (Special Files) Your company has been using a device on a computer to collect logs and store them in a tamper-proof fashion. Programs log to this device by writing to the device `/dev/tplog`. You have been asked to implement a software solution. After having all the functionality implemented, now you wonder how to make your solution work for all the existing legacy programs. Fortunately, the device file name (`/dev/tplog` currently) is defined by some global environment variable, and you can change it. How would you design your program to interact with the environment so that the legacy programs don't have to be modified?

4. (Small Programming Assignment) Write a program to list *and count* the entries in a directory and all its subdirectories. A process can either list entries or count them, but not both. In order to do both you need to create at least two processes: the first process lists the entries of the directory and sends them to the second process, which in turn counts the number of entries received from the first process. The processes communicate using *unnamed pipes*. Your program should have one argument, namely the name of the directory you want to list and count (e.g. `/foo/bar/tree`). Make sure that all processes are terminated once your program terminates.

References

[1] A. Silberschatz, P. Galvin, and G. Gagne, *Applied Operating Systems Concepts*, John Wiley & Sons, Inc., New York, NY, 2000.

[2] Deitel, Deitel, and Choffnes, *Operating Systems*, Pearson / Prentice Hall, 2004

[3] Mark J. Rochkind, *Advanced Unix Programming*, Addison Wesley, 2008.