PROJECT 3 FREQUENTLY ASKED QUESTIONS

1. **Will I need to implement new classes and data structures?**

Yes, you will have to implement the following data structures and classes:

- **A system's open file class**: Needs to contain information such as the filename, an OpenFile object, and the number of processes that have this file open. The OpenFile object is what will be used to actually read and write to this file. You can look at this in filesys/openfile.h directory where the stubs are contained. The number of process with the file open will be helpful to determine if a file really can be closed or not because other processes are still using it. It will also determine if a file needs to be opened or if it is already open.
- **The system's open file structure**: This will contain the array of the system's open files and allow access to those objects. It will decide if a file already exists or if it really needs to be closed or not. You can use the bitmap (that you used in part one of the project) for managing the array of open files.
- **A process' open file class**: Needs to contain information such as the file name, the offset into this file for this particular process, and the index into the system wide file table. The offset is used for when you are reading or writing into the file. This is where you start reading from again if you do another read (or writing if you did a write). The index into the system's file table gives us access to the actual file of the system (that is synchronized from all different processes that access it). This mapping of a process' open file to the system file table allows each process to have its own offset into the file (different from other processes) and makes it appear to the user that he actually has this file open all to himself (when really there is only one open copy of this file in the system's open file table).
- **The process' open files structure**: This will be an attribute of each PCB. It will have information such as an array of this process' open files which include the "always present" stdin and stdout. You can make the maximum number of open files for a process anything greater than 20. Once again, you can use the bitmap for this too.

2. **Should we make any changes to the Project 2 code?**

Yes, you will need to close all your process' file when you exit, and you will need to add the process' open file data structure to your PCB class.

3. **What should I take into consideration for the console implementation?**

To write to the console a user does not need to open or create any new files (this should be done in the constructor of the process' open file list). The user can simply call Read with "stdin" or write with "stdout" as the file name. Writing to stdout should just print to the terminal (use printf for this). Reading should just wait for the user to type something in and then press enter (use scanf to do this). Remember you cannot read from stdout and you cannot write to stdin.