Chris Bradshaw
Assignment 3 Commentary

This lab was divided up into two pieces: the API and the test program which uses this API. An input directory is needed to run the program, as well as a specified output folder name and log file name. The log file, summary and HTML file will be output in the current working directory. API code is bundled into a library called “libfstest.a”. This is linked with the test program. All of complication can be achieved with a simple “make” call.

**API Code**
I deviated slightly from the project specifications, although everything behaves as expected. A list of these deviations can be found in the README file. Essentially, I either added/removed parameters from function calls that were necessary/unnecessary. As stated in the project specifications, I have the core API functions, as well as some getters that I thought would simplify things, such as Get_Filesize, Get_UID and Get_GID. These core API calls used additional lower level calls, such as Block_Read, Block_Write, etc. Logging was achieved with a LOG function which had 4 different levels of logging, depending on the scenario. If the LOG function was writing to a log file, a timestamp was automatically added to the message. Additional functions, such as print_directory and print_inodes are also defined.

**Test Code**
The usage for the program is as follows: ./mini_filesystem (input_dir) (output_dir) (log_filename). These arguments are immediately verified at the beginning of the program. Following this, the log file is cleared and created. Then the filesystem is initialized and the input directory is searched and all files (directory structure not maintained) are added to the filesystem. Once all files have been copied, all JPG files are read from the filesystem to the output directory, preserving the UID and GID. Once all files have been copied, thumbnails and an HTML file is generated for the images. This is output in the current working directory.