Project 3 Write-Up

**Table of Compressed Sizes and Compression Ratios for Test Files**



1. A table displaying the original file size for all of the test files and the compressed size and compression ratios for all of the compression algorithms is shown above. LZWmod.java with and without the reset and compress.exe seems to have better compression ratios than LZW.java for all of the test files. This is possibly due to the nature of LZWmod.java and compress.exe, where (unlike LZW.java) the programs increase the bit-width as codewords increase during the compression and decompression processes. compress.exe also has a slightly better compression ratio than LZWmod.java and LZW.java for files that would expand during the compression process, such as Lego-big.gif and frosty.jpg. Unix compress will leave the original file as is and avoid compression if the compression ratio is less than 1.
2. wacky.bpm gives the best compression ratio for all algorithms (all above 200), most likely because the file has very little color value variation, and therefore less information entropy. As we discussed in lecture, the less entropy within a file, the better the compression. Lego-big.gif and frosty.jpg gave the worst compression ratios for all of the files. Again, this is likely due to the amount of entropy within both files. Both have relatively small color schemes, but each file lacks a distinct pattern in how the colors are laid out within the files (Lego-big.gif having a “random” layout of different colored Lego blocks and frosty.jpg having the same face presented in different angles).