BLAKE:

Test Number	Name	Description	Result
1	TestBlakeLarge	Tests BLAKE on a large file (244.3 MB disk image file). Expected result has been calculated using the b2sum utility.	Pass
2	TestBlakeSmall	Tests BLAKE on a very small file (txt file containing the string "hello abcdefghijklmnop" = 23 bytes).	Pass
3	TestBlakeMix	Tests BLAKE's mixing function (used in compression). Uses values obtained from a working version of the program (checked against official test vectors).	Pass
4	BenchmarkBLAKELarge	Runs test number 1 multiple times (without checking checksum), and calculates the time taken on average for each run.	Time per operation: 1676409286 ns Speed: 145.74 MB/s
5	BenchmarkBLAKESmall	The same as test number 4, however it runs test number 2.	Time per operation: 5282 ns Speed: 4.35 MB/s

AES:

Test Number	Name	Description	Result
1	TestSubBytes	Tests the SubBytes step of AES. Input is some numbers I randomly generated in Python, and the expected result I had to do by hand.	Pass
2	TestInvSubBytes	Tests Inverse SubBytes, using the values in test 1, but inverted (expected values = test values, test values = expected values).	Pass

3	TestShiftRows	Tests the ShiftRows step of AES. Uses the same input values as test 1, and I did the expected result by hand.	Pass
4	TestInvShiftRows	Tests the Inverse ShiftRows step of AES. Uses the same input values as test 3, but inverted.	Pass
5	TestMixColumns	Tests the MixColumns step of AES. Uses test vectors defined here: https://en.wikipedia.org/wiki/Rijndael_MixColumns	Pass
6	TestInvMixColumns	Tests the MixColumns step of AES. Uses the same test vectors in test 5.	Pass
7	TestKeyExpansionCore	Tests the core part of the Key Expansion step at the start of AES. Test vectors calculated by hand again.	Pass
8	TestKeyExpansion	Tests the core Key Expansion step at the start of AES. Test vectors calculated using working version (result of working version was tested against official test vectors).	Pass
9	TestEncDecMediumFile	Encrypts a medium sized file (48.5 MB), then decrypts it again. A checksum of the original file is calculated using the “b2sum” command, and is compared with the checksum of the decrypted file. Both should be the same otherwise there is something wrong with enc/decryption.	Pass
10	TestEncDecSmallFile	Same as test 9 but with a smaller file (A 23 byte txt file containing “hello abcdefghijklmnop”).	Pass
11	BenchmarkEncryptFileLarge	Encrypts a large file (244.3 MB disk image file) multiple times, and gets the average time for	Time taken: 1871293107 ns Speed:

		each run.	130.56 MB/s
12	BenchmarkDecryptFileLarge	Same as test 11, but decrypts instead.	Time taken: 2042360853 ns Speed: 119.63 MB/s
13	BenchmarkEncryptFileMedium	Encrypts a medium file (48.5 MB image) multiple times, and gets the average time for each run.	Time taken: 378539062 ns Speed: 128.09 MB/s
14	BenchmarkDecryptFileMedium	Same as test 13, but decrypts.	Time taken: 417413577 ns Speed: 116.16 MB/s