Figuring out what the mystery program does:
First off, a big give-away was that a function was called compute_fib for obvious reasons. I went line by line through the assembly code and made sense of it by turning what i understood into equivalent C code. Once i got to the first call of num, i had no idea what this was doing. After some research i came to know that is how arrays were implemented as global variables. Upon completing my C code, i still had no idea how this code worked algorithmically even though it was pumping out the correct numbers. When i thought about what the global variable was doing in a recursive program, i realized it could be a history to store calculated values.

Optimized vs. Unoptimized:
The optimized code doesn’t create as much space for local variables as the unoptimized. The unoptimized creates enough room for 28 bits, but only uses about 8 bits. The optimized code performs all the operations which are to be in nested outside the if and else statements such that the code does not contain multiple blocks of similar code. The optimized code uses testl which is the logical operator AND. I assume it does this to knock out an if-else statement and sending the data to its appropriate destinations faster ie. fewer condition checks.