Joseph A. Boyle

Identifying Functionality

By performing several calculations of mystery (1, 2, 3, 5, 10, etc), it became evident quickly that the mystery code produced the **n**th value in the Fibonacci sequence. Upon noticing that dothething was recursive (two calls to itself), one reducing n by 1 and the other reducing n by 2, this hypothesis was confirmed.

The usage of **num** was not immediately obvious, but upon observing that it was 800 bytes, the value $199 was skewed throughout the program, and we reference it in various parts of the code, it became obvious that **num** was a static variable which held 800 bytes, divided into chunks of 4 -- likely an array of ints (integers are 4 bytes * 200 == 800 bytes). Looking into dothething, it became obvious that it was checking if the value of num at an index **n** was -1 (as set in main). IF so, it continued the computation. Otherwise, it returned **num**[**n**]. Thus, I concluded that the program uses an int array to store previously calculated fib values.

Optimizations

The optimized code uses unsigned arithmetic (signified by using **ja** instead of **jg**, and other jump conditions) when possible, and heavily reduces the number of machine instructions required. It also removes the call to **add** *function*, instead replacing it with a machine instruction for add, which will reduce overhead for a number of reasons -- less stack frames being created, less jumping, etc. The unoptimized version of the code also uses significantly more jumps, which will require more comparisons to be computed. Additionally, the optimized version utilizes leal much more commonly, which reduces the need for several machine instructions.

Minimizing the number and complexity of instructions ultimately yields to faster code, as the compute doesn't have to wait until the next fetch-decode-execute cycle -- instead, it can handle computations faster by not having to perform as many. In terms of readability, despite being shorter, the optimized version seems much closer to the C code we produced -- it follows the logical progression, without taking convoluted steps around the logic.