I tried two different approaches, which I called (1) DumbAdd and (2) SmartAdd. Although both work for the prompt, only DumbAdd works with generality.

(1) DumbAdd just adds the numbers in the “normal” way, but with a different data-structure, treating each number as an array of digits. Let M be the number of numbers added and L be the number of digits. Then this algorithm runs in O(ML) time.

Better would be only paying attention to a small number of leading digits rather than all L digits. (2) SmartAdd assumes that you only need to take account of the D+2 leading digits of each number where D is the number of digits desired. Thus, it runs in O(MD) time, which is great.

Unfortunately, the assumption is false. Without knowing the numbers in question, there is no cutoff of digits to be scrutinized that can be calculated in advance (unless, \*possibly\*, D is less than log-10(m), which seems like it can be ignored, so I didn’t totally investigate this).

To see this, imagine we want the leading D=10 digits, with M=100, and L=50. Let the first 98 numbers be 1000…(49 zeros), the 99th be 19999 (49 nines), and the 100th be 100000….0001 (50 digits). If we scrutinize anything short of all 50 digits, then we’ll get the wrong answer.

Thus, SmartAdd happens to work with the given data, but isn’t guaranteed to work in general (except, possibly, under the rare circumstance that log(m) >> d). DumbAdd is ultimately smarter than SmartAdd.

(3) Finally, here is a brief proposal for a third algorithm, LessDumbAdd. The basic idea is that, starting at the front, keep a running sum, adding each column of digits one at a time and add it to the running sum. When a certain condition is satisfied, we stop looking at additional columns. Let Q be the number of digits we’ve looked at, L the total number of digits per number, D the desired number of digits, and R be the number of digits in the running sum. If we assume everything we haven’t looked at to be a 9, to be safe, then the unscrutinized sum could be M\*(10^(L-Q)-1). The digit-length of this sum is ceiling(log(preceding expression)). Now we go to that digit-length in the running sum, and we keep going forward in the running sum so long as the next digit is a nine. When we’ve stopped, we check to see if we’re within D or R or not. If so, we need to keep checking digit-columns. If not, then we’re good and we can stop the whole thing.

LessDumbAdd would run in O(ML) time on average, like DumbAdd, but since on average (I think?), the terminal Q would be (D+L)/2, it should run in about half the time. It might be possible to construct a monster case that would take O(ML^2) time, but I’m not totally sure about that.

I apologize for any mistakes I’ve made in the (3)-speculations that end up misleading anyone!

Code for DumbAdd

Code for SmartAdd