DIFFIE-HELMAN

A short Python script[1] I wrote provided me with the following: a = 41, b = 23, k = 52.

I shudder to think what would have happened if the numbers had been larger. My computer would likely have melted trying to brute force a compatible number with the proper remainder, and if I'd done it by hand (which I didn't even have the commitment to do with the two digit variables) I would have collapsed into a pile of dust before I got halfway there.

RSA

In order to avoid having to brute force prime factors by hand, I also wrote a short C script to find the prime factors of a given integer[2] (if they exist in primes less than 1000). Given the number 170171, my program produced p = 379 and q = 449. This means that (p-1)(q-1) = 169344. I detoured again briefly to produce a C script that gave me a suitable d[3], and it produced d = 119537. Finally, in order to obtain the message text from Alice's ciphertext, I created one last Python script[4] to produce message text from ciphertext. I converted the decimal blocks into hexadecimal, then split each into two ASCII characters. The message reads as follows:

[Hi Bob. I'm walking from now on. Your pal, Alice.
https://foundation.mozilla.org/en/privacynotincluded/articles/its-official-cars-are-the-worst-product-category-we-have-ever-reviewed-for-privacy/]

I would have failed entirely trying to find the prime factors had the number been larger. It's the fundamental problem. Alice's encoding itself wasn't all that hard to crack once it was private-key decrypted. Hex and ASCII aren't exactly a secret. The only other thing I can think of is encrypting it with her own private key, to show that it was truly her sending it to Bob.

[1] See diffieHelman.py
[2] See primeFactors.c
[3] See findD.c
[4] See deRSA.py