Kick Start 2018 - Round F

Analysis: Palindromic Sequence

To solve the Small dataset, we will build up the palindrome one letter at a time. Suppose we know the first X letters of the **K**th lexicographically least palindrome. Call this prefix of the answer S. We can guess the next letter and see how many palindromes there are with the new prefix (we will address this computation shortly). If there are more than **K**, then we know we need to guess a lexicographically smaller letter. Let P(S) denote the number of palindromes of Hannah's language that have S as a prefix. Then we are looking for the lexicographically largest character c such that P(S + c) ≤ **K**. Naively, we have to make O(**NL**) guesses.

So how do we actually calculate the number of possible continuations? Fortunately, the bounds are Small enough that we don't need to do anything sophisticated. For every possible length M, check and see if the current prefix S is consistent with a palindrome of length M. If it is, then there are $\mathbf{L}^D$ possible palindromes of length M, where D = max(0, floor((M+1)/2) - X). d represents the number of characters that can be freely chosen (the rest are fixed to keep the string a palindrome). This can be done naively in $O(\mathbf{N}^2)$ time. In total, the Small can be solved in $O(\mathbf{N}^3\mathbf{L})$ time.

To solve the Large dataset, notice that when **K** ≤ **N**, the answer is a string that consists of the letter `a` repeated **K** times. When **K** > **N**, it's important to notice that the answer is very close to **N**.

For example, consider the case when **N** = 10^5 and **L** = 2. How many palindromes begin and end with 49900 letter `a`s? To make a rough count, let's consider only the palindromes with length exactly **N**. Of the 200 letters between the `a`s, the first 100 are unconstrained, while the last 100 have to match the corresponding letter in the first 100. In total this is 2^{100} different palindromes, which far exceeds that maximum bound for **K** (10^{18} is about 2^{60}).

The example above can be generalised to state that whenever **K** > **N**, the **K**th palindrome will begin and end with at least $\mathbf{N}-\log_{\mathbf{L}}(\mathbf{K})$ copies of the letter `a`, leaving at most $2*\log_{\mathbf{L}}(\mathbf{K})$ letters in the middle that might not be the letter `a`. The algorithm in the Small dataset can be adapted to solve the Large dataset after these observations have been made. $2*\log_{\mathbf{L}}(\mathbf{K})$ is no more than 100, so this has a similar running time to the Small dataset.