A - AK the Problems

B - Bigram Language Model

For each query (s, t), the nominator is the number of co-occurrences of (s, t) in the corpus, and the denominator is the number of occurrences of s as non-terminal word in a sentence. These values can be pre-processed by a single pass over the corpus.

The tricky part is that the number of distinct words appearing in the corpus w may be up to 10^5, so it is impossible to explicitly construct the co-occurrence matrix of size $w \times w$. The observation is that the sum of all elements in this matrix is bounded by the size of corpus, so we can use some sparse-matrix representation techniques (e.g. only recording which entries are non-zero and the values they hold).

G - Greenberg Mass Comparison

The core of this problem is asking you how many ways are there to partition a set of n elements into disjoint, non-empty subsets. If you know Bell number, that's exactly what we are asking for (but you still have to do the programming). If not, not a problem! Let's work out a recurrence formula for it.

Let B_n denote the answer for n. Consider the number of elements **NOT** in the subset containing element 1. If there are k such elements, then there are $\binom{n-1}{k}$ ways to choose these elements, and we can just ignore the subset containing 1 and do the partition on these k elements. Therefore,

$$B_n = \sum_{k=0}^{n-1} \binom{n-1}{k} B_k$$

with base case $B_0 = 1$.

Notice that Bell numbers can get very large quickly, so coding this problem in Python or Java will save you a lot of trouble.