CMSC 423 Homework #3:

Due: Nov. 17 at the start of class

You may discuss these problems with other students, but you **must write up your solutions independently**, without using common notes or worksheets. You must indicate at the top of your homework who you worked with. Your write up should be clear, concise, and neat. You are trying to convince a skeptical reader that your algorithms or answers are correct. Messy or hard-to-read homeworks will not be graded.

1. You are given K strings $S_1, \ldots, S_K$ of total length n. Define $\ell(k)$ for $k = 2, \ldots, K$ to be the length of the longest substring that is common to at least k of the strings $S_1, \ldots, S_K$. For example, if the strings are `ration`, `natural`, `nation`, then $\ell(3) = 2$ (`at`), $\ell(2) = 5$ (`ation`).

   Give an $O(Kn)$-time algorithm to compute $\ell(k)$ for all $k = 2, \ldots, K$.

2. **DNA contamination.** When sequencing DNA, often contaminant DNA (e.g., bacteria in the lab equipment, the operator's DNA, etc.) is accidentally sequenced in addition to the target DNA. Suppose you are given a set of strings $C_1, \ldots, C_K$ representing the DNA of known contaminants, where $\sum_i |C_i| = n$.

   A string S of length m is sequenced. Give an algorithm with runtime $O(n+m)$ that finds the locations of **all** the substrings of S that occur in some C_i and that are longer than a given parameter t. (We don't care *which* C_i any contaminant comes from, but we do want to know where every contaminant instance occurs in S.)

3. Let T be a suffix tree for string S. Let $\mathbf{str}(u)$ be the string that is spelled out when walking from the root of T to a node u. A node in T is called *left diverse* if the occurrences of $\mathbf{str}(u)$ in S are not always preceded by the same character. For example, if $S = ababacb$ then the node representing ba is not left diverse since ba is always preceded by a. But the node with $\mathbf{str}(u) = b$ *is* left diverse because sometimes b is preceded by a and sometimes by c.

   Give an $O(|S|)$ algorithm to mark the left-diverse nodes in T.

4. Compute the Burrows-Wheeler transform on string `defenselessness`.

5. Compute the **inverse** Burrows-Wheeler transform of the string `nnooi$`.