1. Given a circular string $S$ of $n$ characters, the **circular string linearization problem** is to choose a place to cut $S$ so that the resulting linear string is the lexically smallest of all the possible $n$ linear strings created by cutting $S$. (This problem needs to be solved for storing circular strings in a database in a canonical linear form. Gusfield, Sec. 7.13) Explain how to use suffix trees for solving the problem in linear time.

2. Consider the computation of frequent common substrings discussed at the lecture. Explain how the computation of the $l(i)$ values can be extended to give also pointers to the corresponding substrings.

3. (Gusfield, Ex. 7.19) The Aho-Corasick method solves the exact set matching problem by first building in linear time an automaton of size $O(n)$, where $n$ is the total length of the patterns, and then searching the text (of length $m$) in time $O(m)$. We discussed at the lecture how suffix trees can be used to solve the problem using $O(m)$ time for building a suffix tree, and $O(n)$ time for the search phase. Explain how to solve exact set matching with generalized suffix trees, obtaining exactly the asymptotic time and space bounds of the Aho-Corasick method. (Hint: matching statistics)

4. The substrings of a newly sequenced string $S_1$ that are potential contaminations from some string in a set $S$ can be reported efficiently as an application of matching statistics. Explain the idea.

5. Present the suffix array for the text $baababbac$. Explain how it would be used to locate occurrences of the pattern $ba$ in the text.