1. Using asymptotic notation, derive upper and lower bounds to the number of different
   (a) prefixes and suffixes
   (b) substrings, and
   (c) subsequences
   of a string $S = s_1 \ldots s_n$.

2. (Gusfield, Ex. 1.2) A circular string of length $n$ is a string in which character $n$ is considered to precede character 1. (Bacterial and mitochondrial DNA is typically circular.) Design a linear-time algorithm to determine whether a linear string $\alpha$ is a substring of a circular string $\beta$. (Use the existence of a linear-time exact matching algorithm to solve this problem.)

3. (Gusfield, Ex. 1.3) Suffix-prefix matching. Give an algorithm that takes in two strings $\alpha$ and $\beta$, of lengths $n$ and $m$, and finds the longest suffix of $\alpha$ that exactly matches a prefix of $\beta$. The algorithm should run in $O(n + m)$ time.

4. (Gusfield, Ex. 1.11) Let $T$ be a text string of length $m$ and let $S$ be a multiset of $n$ characters. The problem is to find all substrings in $T$ of length $n$ that are formed by the characters of $S$. For example, let $S = \{a, a, b, c\}$ and $T = abahgcabah$. Then $caba$ is of substring of $T$ formed from the characters of $S$.

   Give a solution to this problem that runs in $O(m)$ time. The method should also be able to state, for each position $i$, the length of the longest substring in $T$ starting at $i$ that can be formed from $S$.

5. Simulate Boyer-Moore matching to locate occurrences of pattern “maamamma” in text “jo hommaamme maamamman”
   (a) applying the bad character shift rule
   (b) applying the good suffix shift rule
   (c) selecting the maximum shift given by the bad character rule and the good suffix rule.

   In each of the above cases, indicate the shifts and the character comparisons performed.

6. (Gusfield, Ex. 2.9) Explain why Theorem 2.2.4 holds, and apply it to design a linear-time algorithm to accumulate the $l(i)$ values (or $l'(i)$ using the notation of the textbook) in linear time. (Assume that the $N_j(P)$ values have been computed.)