Individual Assessment 2: String Data Structures

This is an individual assessment, and, as the name suggests, must be completed individually. Specifically, you're not allowed to work with a partner, and you should not discuss these problems with other students in CS166. However, the course staff are happy to answer clarifying questions on Piazza (if you do, please post the question privately) or in our office hours.

Due Tuesday, April 28th at 2:30PM Pacific time.
Problem One: The Anatomy of Suffix Trees

Consider the following paragraph, which is taken from an English translation of the excellent short story “Before the Law” by Franz Kafka:

Before the law sits a gatekeeper. To this gatekeeper comes a man from the country who asks to gain entry into the law. But the gatekeeper says that he cannot grant him entry at the moment. The man thinks about it and then asks if he will be allowed to come in later on. "It is possible," says the gatekeeper, "but not now."

Actually building a suffix tree for this text, by hand, would be quite challenging. But even without doing so, you can still infer much about what it would look like.

i. Without building a suffix tree or suffix array for the above text, determine whether the suffix tree for this passage contains a node labeled with the string "moment". (Here, the label on a node is the string read by tracing out a path from the root of the tree all the way down to that node.) Briefly justify your answer.

ii. Repeat the above exercise for the string "man".

iii. Repeat the above exercise for the string "gatekeeper".

Problem Two: Longest k-Repeated Substrings

Design an O(m)-time algorithm that, given a string T of length m and a positive integer k, returns the longest substring that appears in T in at least k different places. The substrings are allowed to overlap with one another; for example, given the string HAHAHAHAHA and k = 3, you'd return HAHAHA. Then, prove that your algorithm is correct and justify the runtime. (There’s no need to code this up – just describe it in plain English.)

As a note, the runtime of O(m) is independent of k. For full credit, your solution use should not use suffix trees, though you're welcome to use any other data structures you'd like.