Algorithms: Assignment 10

Due date: November 30 (Thursday)

Problem 1 (2 points)
Assume that all characters in a pattern $P[1..m]$ are *distinct,* and you need to find all occurrences of P in a text $T[1..n]$. Write an "accelerated" version of NAIVE-STRING-MATCHER, which solves this problem in $O(n)$ time.

Problem 2 (3 points)
Write an algorithm that looks for a given $m \times m$ pattern in an $n \times n$ array of characters, based on the Rabin-Karp method. The pattern may be shifted vertically or horizontally within the $n \times n$ array, but it cannot be rotated.

Problem 3 (5 points)
Design an efficient algorithm for finding a longest common substring of two strings. Your algorithm should output not only the length of this substring, but also the substring itself.