Part 1 of Assignment #2 (1 problem, 50 points)
(Course: CS 401)

This is part 1 of assignment 2. The remaining problems of assignment 2 will be given later.

For regular students, the deadline is **April 1, Monday, in class**.

For special needs students, the deadline is **April 8, Monday, in class**.

No late assignments will be accepted.

Special note: Any answer that is not sufficiently clear even after a reasonably careful reading will not be considered a correct answer, and only what is written in the answer will be used to verify accuracy. No hand waiving, vague descriptions or sufficiently ambiguous statements that can be interpreted in multiple ways will be considered as a correct answer, nor will the student be allowed to add any explanations to his/her answer after it has been submitted.

Problem 1 (50 points): A *string* $\mathcal{S}$ over an alphabet Σ is a *concatenation* of some symbols from Σ. For example, if $\Sigma = \{a, b, c\}$ then both $abacaabca$ and $cbaaab$ are strings over Σ.

For two strings $\mathcal{S}$ and $\mathcal{T}$, we say that $\mathcal{T}$ is a *substring* of $\mathcal{S}$ if $\mathcal{T}$ can be obtained from $\mathcal{S}$ by *deleting* one or more symbols from $\mathcal{S}$. For example, if $\mathcal{T} = cac$ and $\mathcal{S} = babcbaabbccca$ then $\mathcal{T}$ is a *substring* of $\mathcal{S}$ since

$$\overbrace{\not b \not a \not b\ \boldsymbol{c}\ \not b \not a\ \boldsymbol{a}\ \not b \not b \not c\ \boldsymbol{c}\ \not c \not a}^{\mathcal{S}} \text{ is same as } \overbrace{cac}^{\mathcal{T}}$$

Given two strings $\mathcal{S} = s_1 s_2 \ldots s_n$ and $\mathcal{T} = t_1 t_2 \ldots t_m$ over some alphabet Σ, the goal of this problem is to design a greedy algorithm that decides in $O(m + n)$ time if $\mathcal{T}$ is a substring of $\mathcal{S}$. For this purpose, answer the following questions.

***(a)* [15 points]** Describe a greedy algorithm that, given two strings $\mathcal{S} = s_1 s_2 \ldots s_n$ and $\mathcal{T} = t_1 t_2 \ldots t_m$ over some alphabet Σ, does the following:

- decides if $\mathcal{T}$ is a substring of $\mathcal{S}$ and outputs a "yes" or "no" response accordingly, and
- if $\mathcal{T}$ is a substring of $\mathcal{S}$ then shows which symbols of $\mathcal{S}$ are deleted to make it same as $\mathcal{T}$.

It suffices to describe the algorithm in pseudo-codes as long as sufficient details are provided. Justify why your algorithm runs in $O(m + n)$ time.

***(b)* [35 points]** Prove that your greedy algorithm works correctly. For this, you must show both of the following:

(b-1) [10 points] if your algorithm outputs "yes" then $\mathcal{T}$ is a substring of $\mathcal{S}$, and

(b-2) [25 points] if $\mathcal{T}$ is a substring of $\mathcal{S}$ then your algorithm outputs "yes".