Assignment #2 (3 problems, 100 points)
(Course: CS 401)

For regular students, the deadline is **November 10, Monday, in class**.

For special needs students, the deadline is **November 17, 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 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

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

Given two strings $\mathcal{S} = s_1 s_2 \dots s_n$ and $\mathcal{T} = t_1 t_2 \dots t_m$ 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*) [20 points] Describe a greedy algorithm that, given two strings $\mathcal{S} = s_1 s_2 \dots s_n$ and $\mathcal{T} = t_1 t_2 \dots t_m$ 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*) [30 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".

Problem 2 (30 points): This question is related to two claims made by Professor *Smart*, who has stated that he is smarter than the instructor and all the students in this class. We will examine his smartness by verifying the claims that he made.

Given an undirected, weighted graph G (with non-negative weights $w(u, v)$ for each edge $\{u, v\}$), and a constant $D > 0$, Professor *Smart* defines the graph G^{+D} as follows:

> *G^{+D} is identical to G except we* **add** *the constant D to each edge weight.*

(a) [15 points] Professor *Smart* claims that:

> *a minimum spanning tree (MST) of G is* **always** *an MST of G^{+D}.*

(b) [15 points] Professor *Smart* claims that:

> *a shortest path between vertices s and t in G is* **always** *a shortest path between s and t in G^{+D}.*

For *each* of the above two claims, your task is the following:

- If the claim is true, then provide a proof of the claim. This will show that Professor *Smart* was *indeed smart*.
- If the claim is false, provide a counter-example and explain why the counter-example shows that the claim is false. This will show that Professor *Smart* was *not so smart* after all.

Hint: For part (a) think about how Kruskal's algorithm works.

Problem 3 **(**20 **points):** Give a counter-example (graph) to show that Dijkstra's algorithm may **not** work correctly for a weighted graph **with negative weights** but **with no cycles of total negative weight**. **Explain clearly why this is a counter-example; full credit will not be given without clear explanation**.

Hint: There is a counter-example graph with only three nodes (but other counter-example graphs are also OK).