This is a four hour exam. There are six questions, worth 10 points each. Answer all questions and all their subparts.

This is a closed book exam. No books, notes, reference material, either hard-copy, soft-copy or online, is allowed.

In the past years, to pass this exam, one needed to answer 3 or 4 problems well, instead of answering all the problems poorly.

Please print your name and SID on the front of the envelope only (not on the exam booklets). Please answer each question in a separate booklet, and number each booklet according to the question.

Read the questions carefully. Keep your answers legible, and brief but precise. Assume standard results and algorithms (i.e., those taught in class or referred to in the homeworks).

You must prove correctness of your algorithm and prove its time bound unless stated otherwise. The algorithm can be written in plain English (preferred) or as a pseudo-code.

Best of luck!
Problem 1 (Graphs are undirected)

In this problem, we will study an algorithm to find the minimum spanning tree in a graph that is different from the one presented in class. Let \( G(V, E) \) be a connected undirected graph and \( \text{wt}(e) \geq 0 \) denote the weight of an edge \( e \in E \). The algorithm proceeds as below. Here \( S \subseteq V \) is a set that grows larger as the algorithm proceeds and \( T_S \) is a graph on the vertex set \( S \). All the edges of \( T_S \) are from \( E \).

1. Initialization: Let \( S = \{v\} \) where \( v \in V \) is an arbitrary vertex and let \( T_S \) have no edges.

2. Let \( e \) be the edge with minimum weight among the edges in \( E \) whose one endpoint is in \( S \) and the other endpoint is in \( V \setminus S \). Call the endpoints of \( e \) in \( S \) and \( V \setminus S \) as \( u \) and \( w \) respectively.

3. Add \( w \) to \( S \) and add the edge \( e = (u, w) \) to \( T_S \).

4. Stop if \( S = V \). Otherwise repeat Steps (2) and (3).

You need to show the correctness of this algorithm. Do not worry about the running time.

(a) Show that \( T_S \) is a tree on the set of vertices \( S \) throughout the execution of the algorithm.

(b) Show that when the algorithm stops, \( T_S \) is a minimum spanning tree of \( G \).

\textit{Hint: Induction? Replacement argument?}

Problem 2 (Graphs are directed)

Let \( G(V, E) \) be a directed graph whose vertices are colored red or blue. A \( t \)-alternating walk is a walk (i.e., a path where vertices could repeat) \( v_0, v_1, \ldots, v_k \) that has at least \( t \) color transitions, from red to blue or vice versa. Assume \( 1 \leq t \leq |V|^2 \).

(a) If \( G \) is acyclic, design a \( O(|V| + |E|) \) algorithm to decide whether \( G \) has a \( t \)-alternating walk.

(b) Now let \( G \) be a general directed graph (not necessarily acyclic). Design a \( O(|V| + |E|) \) algorithm to decide whether \( G \) has a \( t \)-alternating walk.

Assume adjacency list representation of graphs.
Problem 3
Let $x$ denote a string of length $n$ over the 4-symbol alphabet $\{A, C, G, U\}$. For example,

$$x = AGCUUCGAU, \quad n = 9.$$  

For an index $1 \leq i \leq n$, let $x_i$ denote the $i^{th}$ symbol of $x$. A folding of the string $x$ is a set $E$ of unordered pairs of indices from $\{1, \ldots, n\}$ satisfying the following.

1. $E$ is a matching, i.e., no index $i$ appears in more than one pair in $E$.

2. For all $(i, j) \in E$, $(x_i, x_j)$ is either $(A, U)$, $(U, A)$, $(C, G)$ or $(G, C)$. I.e., pairs can only connect $A$ with $U$ and $C$ with $G$.

3. For all $(i, j) \in E$ with $i < j$, it holds that for all $(k, \ell) \in E$, either both $k, \ell$ are in the interval $\{i + 1, \ldots, j - 1\}$ or both are outside the interval $\{i, \ldots, j\}$. I.e., the matching is non-intersecting.

For example, if $x = AGCUUCGAU$, a possible folding is given by $E = \{(1, 9), (2, 3), (5, 8), (6, 7)\}$.

Design a polynomial time algorithm that given a string $x$ finds a folding of maximum cardinality.

Problem 4 (Graphs are undirected)
Let $G(V, E)$ be an undirected graph. For a subset $S \subseteq V$ of vertices, its neighborhood $\text{Nbd}(S)$ is defined as

$$\text{Nbd}(S) = \{v \mid v \in S \text{ or } (u, v) \in E \text{ for some } u \in S\}.$$  

An $n$-vertex graph is called an expander graph if for every subset of vertices $S \subseteq V$, $1 \leq |S| \leq \frac{n}{2}$, we have $|\text{Nbd}(S)| \geq \frac{11}{10} \cdot |S|$. The goal is to show that the diameter of an expander graph, i.e., the maximum distance between any pair of vertices, is $O(\log n)$. Assume therefore that $G$ is an expander graph. For a vertex $s$ and an integer $j \geq 0$, let $D_j(s)$ denote the set of all vertices whose distance from $s$ is at most $j$.

(a) Show that for every vertex $s$, and any $j \geq 0$, we have $|D_j(s)| \geq \min\left(\frac{11}{10}^j, \frac{n+1}{2}\right)$.

(b) Show that for every two distinct vertices $s$ and $t$, the distance between $s$ and $t$ is at most $C \log_2 n$, where $C$ is a constant that does not depend on $s$ or $t$ or $n$. 

3
Problem 5 (Graphs are undirected)

Construct an undirected graph \( G(V, E) \) at random as follows. Let \( V \) be a set of \( n \) vertices. For each pair of distinct vertices \( u, v \in V \), let \((u, v) \in E\) with probability \( \frac{1}{2} \), independently for all vertex pairs. That is, for each vertex pair \((u, v)\), the pair is included as an edge with probability \( \frac{1}{2} \) and left out with probability \( \frac{1}{2} \), independently for all vertex pairs.

We intend to analyze the size of the largest independent set in this random graph. Recall that an independent set is a subset of vertices with no edge amongst them. For a subset \( S \subseteq V, 1 \leq |S| \leq n \), let \( E_S \) be the event that \( S \) is an independent set.

(a) What is \( \Pr[E_S] \)?

(b) For \( 1 \leq k \leq n \), let \( D_k \) be the event that \( G \) has an independent set of size \( k \). Can you express \( D_k \) in terms of the events \( E_S \)?

(c) Can you provide an upper bound on \( \Pr[D_k] \) as a function \( k \)?

(d) What is the smallest value of \( k \) for which you can show that \( \Pr[D_k] \leq \frac{1}{100} \)?

The desired value of \( k \) should be in terms of number of vertices \( n \). Give the smallest value you can. It is enough to be correct up to a constant factor. \( n \) can be thought of as sufficiently large.

Problem 6 (Graphs are undirected)

Recall that a vertex cover in an undirected graph is a subset of vertices that touches every edge. Consider the following language:

\[ L = \left\{ G \mid G \text{ is an undirected graph with } n \text{ vertices and has a vertex cover of size at most } \frac{n}{4} \right\}. \]

Show that \( L \) is NP-complete. You can assume NP-completeness of any of the problems discussed in class.