Name:

Student ID #:

- You are expected to work on this assignment on your own
- Use pseudocode, Python-like or English to describe your algorithms. Absolutely no C++/C/Java
- When designing an algorithm, you are allowed to use any algorithm or data structure we explained in class, without giving its details, unless the question specifically requires that you give such details
- Always remember to analyze the time complexity of your algorithms
- Homework has to be submitted electronically on Gradescope by the deadline. No late assignments will be accepted
- Unless otherwise noted, for all questions about graphs, $n = |V|$ is the number of vertices/nodes, and $m = |E|$ is the number of edges/links
Problem 1. (25 points) [Graph Traversals]

Give a $O(n + m)$ time algorithm to determining whether the vertices of a connected undirected graph $G$ can be colored by two different colors (say, red and blue) such that for every edge $(u, v)$, $u$ and $v$ have different colors. When such coloring exists, your algorithm should also compute it.

Answer:
Problem 2. (25 points) [Divide-and-conquer on Graphs]

Let $G = (V, E)$ be an undirected graph. A *triangle* in $G$ is a cycle consisting of exactly three vertices (or, equivalently, three edges). Suppose that $G$ is represented as an adjacency matrix. Give an algorithm to determine whether $G$ contains any triangle in $O(n \log_2 7)$ time.

Answer:
Problem 3. (25 points) [Greedy on Graphs]

Given an undirected graph $G = (V, E)$, an independent set in $G$ is any set $I \subseteq V$ of vertices such that no two vertices in $I$ are connected by an edge. In the maximum independent set problem (MIS), for a given graph $G$, we want to find an independent set of maximum size.

Here is our proposed greedy algorithm: (1) Set $I \leftarrow \emptyset$; (2) Repeat (3-4) until no nodes are left; (3) Choose a vertex $v$ in $G$ of minimum degree (breaking ties arbitrarily). (4) Add $v$ to $I$ and remove from $G$ vertex $v$ and all its neighbors.

Does this greedy algorithm always return the optimal solution? If you think it does, give a proof for the greedy choice property. If you think it does not, give a counterexample.

Answer:
Problem 4. (25 points) [Dynamic Programming on Graphs]

Given a directed graph with non-negative integer edge weights, a pair of vertices s and t, and integers $K$ and $W$, describe a dynamic-programming algorithm for deciding whether there exists a path from s to t that has total weight W and uses exactly $K$ edges. Your algorithm should run in time $O((n + m)WK)$. Analyze the time- and space-complexity of your solution. **Hint:** You will have to define a three-dimensional table for the recurrence relation.

Answer: