1. Prove the following two properties of the Huffman encoding scheme.

   (a) If some character occurs with frequency more than 2/5, then there is guaranteed to be a codeword of length 1. **(8 marks)**

   (b) If all characters occur with frequency less than 1/3, then there is guaranteed to be no codeword of length 1. **(8 marks)**

2. Consider the frequencies $f_1, f_2, \ldots, f_n$ over an alphabet with n distinct letters such that $\sum_{i=1}^{n} f_i = 1$. Further assume that each f_i is a power of 2 (that is, of the form $1/2^k$ for some $k \geq 1$). Show that if Huffman encoding is applied to this set of frequencies then the average number of bits required per letter equals $\sum_{i=1}^{n} f_i \log \frac{1}{f_i}$. **(12 marks)**

3. The supervillain Dr. Evil has hired n drivers to deliver n identical bombs to n different addresses. Each driver has their own well-established delivery route that visits all n addresses, such that no two drivers visit the same addresses at the same time.

   In principle, each of the n drivers can deliver their bombs to any of the n addresses, but there's a complication. Dr. Evil has secretly wired proximity sensors to the bombs. So if two bombs are ever at the same address at the same time, both will explode, destroying both the delivery truck and the building at that address. This can only happen if one driver delivers a bomb to that address, and then later another driver visits that same address *while the bomb is still on their truck.*

   Your job as Dr. Evil's henchman is to assign each driver to a delivery address given the delivery route for each driver as input. That is, describe an algorithm to assign addresses to drivers so that each of the n addresses receives the bomb and there are *no* explosions!

   For example, suppose Driver1's route visits address A1 at 6pm and address A2 at 8pm, and Driver2's route visits A1 at 7pm and A2 at 9pm. Then Driver1 should deliver to A2, and Driver2 should deliver to A1; otherwise, there would be an explosion at A1 at 7pm. **(15 marks)**

4. Let $G = (V, E)$ be a connected undirected graph such that each edge $e \in E$ is colored either Red or Blue. Give a polynomial-time algorithm that takes G, an integer $k \geq 0$ and colors of the edges as input, and either (i) returns a spanning tree with exactly k Red colored edges, or (ii) reports correctly that no such tree exists. **(12 marks)**

5. Let $G = (V, E)$ be a connected undirected graph with edge weights given by $w : E \to \mathbb{R}$. Further let $U \subset V$. Your goal is to find a minimum-weight spanning tree T of G where the nodes in U are leaves in the tree T. There might be other leaves in T as well. Give an algorithm for this problem which runs in $O(|E| \log |V|)$ time. **(14 marks)**

6. Suppose a CS curriculum consists of n courses, all of them mandatory. The prerequisite graph G has a node for each course, and a directed edge from course v to course w if and only if v is a prerequisite for w. Give a linear-time algorithm to compute the minimum number of semesters necessary to complete the curriculum given G as input.

Assume that a student can take any number of courses in one semester, but if v is a prerequisite for w then v must be taken before w. **(8 marks)**

7. Here's a problem that occurs in automatic program analysis. For a set of variables $x_1, \ldots, x_n$, you are given some *equality* constraints, of the form "$x_i = x_j$" and some *inequality* constraints, of the form "$x_i \neq x_j$". Is it possible to sastify all of them?

   For instance, the constraints

   $$x_1 = x_2, x_2 = x_3, x_3 = x_4, x_1 \neq x_4$$

   cannot be satisfied. Give a linear-time algorithm that takes as input m constraints over n variables and decides whether the constraints can be satisfied. **(11 marks)**

8. Design and analyze an algorithm that takes as input an undirected graph $G = (V, E)$ and determines whether G contains a simple cycle of length four. Its running time should be at most $O(|V|^3)$. **(12 marks)**