Problem 1. [5 points]
Let’s consider a long, quiet country road with house scattered very sparsely along it. (We can picture the road as a long line segment, with an eastern endpoint and a western endpoint.) Further, let’s suppose that despite the bucolic setting, the residents of all these houses are avid cell phone users. You want to place cell phone base stations at certain points along the road, so that every house is within four miles of one of the base stations.

(a) **Give an efficient algorithm** that achieves this goal, using the minimum number of base stations.
(b) **Prove** that your algorithm is correct.

Problem 2. [10 points]
You’re helping a group of ethnographers analyze some oral history data they have collected by interviewing members of a village to learn about the lives of people who have lived there over the past two hundred years.

From these interviews, they have learned about a set of \( n \) people (all of them now deceased), whom we’ll denote as \( P_1, P_2, ..., P_n \). They have also collected facts about when these people lived relative to one another.

Each fact has one of the following two forms:
* For some \( i \) and \( j \), person \( P_i \) died before person \( P_j \) was born; or
* For some \( i \) and \( j \), the life spans of \( P_i \) and \( P_j \) overlapped at least partially.

Naturally, they are not sure that all these facts are correct; memories are not so good, and a lot of this was passed down by word of mouth. So what they’d like you to determine is whether the data they’ve collected is at least internally consistent, in the sense that there could have existed a set of people for which all the facts they’ve learned simultaneously hold.

(a) **Give an efficient algorithm** that can verify the information collected by these ethnographers. In other words, it should report whether the events recorded by them are consistent or not. Your answer should clearly describe the algorithm (you can write it in plain English sentences), the input given to the algorithm, and the output expected.
(b) **What is the worst case running time complexity** of your algorithm?

Problem 3. [10 points]
a) The way that Dijkstra’s algorithm is written in page 138 of your textbook has a lot of details compressed into that first line of the body of the while loop. **Rewrite the algorithm**, making sure to clarify the exact operations performed within the body of the loop. You don’t need to go into the data structure details, or write in a Java-ish way. All you need to do is break down that down into smaller steps, and explain them in plain sentences.

b) The worst-case running time complexity of Dijkstra’s algorithm can be made to be \( \Theta(m \log n) \), if we use the right data structures. **Briefly explain** what data structures would you use to implement the algorithm, and how they affect the running time complexity of the algorithm.
Problem 4. [5 points]
For the following statements below, decide whether they are true or false. If a statement is true, give a short explanation. If it is false, give a counterexample.

a) Suppose we are given an instance of the Shortest s-t Path Problem on a directed graph G. We assume that all edge costs are positive and distinct. Let P be a minimum-cost s-t path for this instance. Now suppose we replace each edge cost $c_e$ by its square, $c_e^2$, thereby creating a new instance of the problem with the same graph but different costs. True or false? P must still be a minimum-cost s-t path for this new instance.

b) True or false? If G is a directed graph that has a node with no incoming edges, then G is a DAG.

Problem 5. [10 points - Proof Problem - Please submit separately]
A small business—say, a photocopying service with a single large machine—faces the following scheduling problem. Each morning they get a set of jobs from customers. They want to do the jobs on their single machine in an order that keeps their customers happiest. Customer $i$’s job will take $t_i$ time to complete. Given a schedule (i.e., an ordering of the jobs), let $C_i$ denote the finishing time of job $i$. For example, if job $j$ is the first to be done, we would have $C_j = t_j$; and if job $j$ is done right after job $i$, we would have $C_j = C_i + t_j$. Each customer $i$ also has a given weight $w_i$ that represents his or her importance to the business. The happiness of customer $i$ is expected to be dependent on the finishing time of $i$’s job. So the company decides that they want to order the jobs to minimize the weighted sum of the completion times, $\sum_{i=1}^{n} w_i C_i$.

(a) Design an efficient algorithm to solve this problem. That is, you are given a set of $n$ jobs with a processing time $t_i$ and a weight $w_i$ for each job. You want to order the jobs so as to minimize the weighted sum of the completion times, $\sum_{i=1}^{n} w_i C_i$.

(b) Prove why it solves the problem efficiently.

Example. Suppose there are two jobs: the first takes time $t_1 = 1$ and has weight $w_1 = 10$, while the second job takes time $t_2 = 3$ and has weight $w_2 = 2$. Then doing job 1 first would yield a weighted completion time of $10 \cdot 1 + 2 \cdot 4 = 18$, while doing the second job first would yield the larger weighted completion time of $10 \cdot 4 + 2 \cdot 3 = 46$.

Problem 6. [10 points]
For the upcoming bake sale, you are planning to bake $n$ different recipes, labeled $r_1, r_2, ..., r_n$. Recipe $r_i$ requires $p_i$ minutes of preparation time and $b_i$ minutes of baking time. Fortunately, you have access to a test kitchen with $n$ ovens, so all of recipes can bake simultaneously once they are prepared. However, there is only one of you, so you need to decide in which order to complete the preparation of each recipe.

For example: as soon as you complete preparing the first recipe, you can put it in the oven to bake and immediately begin preparing the second recipe. When you complete the second recipe, you can put it in the oven to bake whether or not the first recipe is done baking; and so on.

Let’s say that a schedule is an ordering for preparation of the recipes, and the completion time of the schedule is the earliest time at which all recipes are done baking. This is an important quantity to minimize, because the bake sale is coming up soon!

(a) Write the pseudocode of a polynomial-time algorithm that finds a schedule with as small a completion time as possible.

(b) Show that your algorithm is correct. A proof is recommended but not necessary, you can justify the algorithm’s correctness in your own words.