1. Let \( X \) be a set of \( n \) intervals on the real line. A subset \( Y \) of these intervals is called a *tiling path* if the intervals in \( Y \) cover the intervals in \( X \), that is, any real value that is contained in some interval in \( X \) is also contained in some interval in \( Y \). The *size* of a tiling cover is just the number of intervals.

Describe and analyze an algorithm to compute the smallest tiling path of \( X \) as quickly as possible. Assume that your input consists of two arrays \( L[1..n] \) and \( R[1..n] \), representing the left and right endpoints of the intervals in \( X \). *If you use a greedy algorithm, you must prove that it is correct.*

2. Consider a weighted version of the class scheduling problem, where different classes offer different number of credit hours, which are of course totally unrelated to the duration of the class lectures. Given arrays \( S[1..n] \) of start times, an array \( F[1..n] \) of finishing times, and an array \( H[1..n] \) of credit hours as input, your goal is to choose a set of non-overlapping classes with the largest possible number of credit hours.

(a) Prove that the greedy algorithm described in class — Choose the class that ends first and recurse — does not always return the best schedule.

(b) Describe an efficient algorithm to compute the best schedule.
3. After a grueling algorithms midterm, you decide to take the bus home. Since you planned ahead, you have a schedule that lists the times and locations of every stop of every bus in Champaign-Urbana. Unfortunately, there isn't a single bus that visits both your exam building and your home; you must transfer between bus lines at least once.

Describe and analyze an algorithm to determine the sequence of bus rides that will get you home as early as possible, assuming there are $b$ different bus lines, and each bus stops $n$ times per day. Your goal is to minimize your arrival time, not the time you actually spend traveling. Assume that the buses run exactly on schedule, that you have an accurate watch, and that you are too tired to walk between bus stops.

4. Describe and analyze an algorithm to find the second smallest spanning tree of a given graph $G$, that is, the spanning tree of $G$ with smallest total weight except for the minimum spanning tree.