1. Minimum Vertex Cover in Trees

   Let $\mathcal{G}$ be an unweighted graph. A vertex cover of $\mathcal{G}$ is a set S of vertices in $\mathcal{G}$ such that every edge in $\mathcal{G}$ is incident to at least one vertex in S (i.e., the vertices in S cover the edges in $\mathcal{G}$.)

   Describe a greedy algorithm that computes the vertex cover of $\mathcal{G}$ if $\mathcal{G}$ is a tree, and prove its correctness.

2. Maximum Independent Sets in Trees

   Let $\mathcal{G}$ be an unweighted graph. An *independent set* of $\mathcal{G}$ is a set S of vertices in $\mathcal{G}$ such that no two vertices in S are connected by an edge. Finding the maximum independent set in a general graph is considered very hard.

   Suppose $\mathcal{G}$ was a tree. Describe a greedy algorithm that computes the maximum-size independent set in a tree.

3. Covering points by intervals

   Consider the problem of covering points by intervals. Specifically, assume that you are given a set P of n points on the real line and a set of m intervals F each specified by two points on the real line. Two discussions ago we solved the *weighted* case for the minimum *weight* set of intervals covering of P.

   In the unweighted case, describe a greedy algorithm to find the minimum number of intervals needed to cover all the points of P.

4. Piercing Intervals

   Let X be a set of n closed intervals on the real line. A set P of points *pierces* X if every interval in X contains at least one point in P. Describe and analyze an efficient algorithm to compute the smallest set of points that stabs X.

5. Weighted Scheduling

   We have n jobs $J_1, J_2, \ldots, J_n$ which we need to schedule on a machine. Each job J_i has a processing time t_i and a weight w_i. A schedule for the machine is an ordering of the jobs. Given a schedule, let C_i denote the finishing time of job J_i. For example, if job J_j is the first job in the schedule, its finishing time C_j is equal to t_j ; if job J_j follows job J_i in the schedule, its finishing time C_j is equal to $C_i + t_j$. The weighted completion time of the schedule is $\sum_{i=1}^{n} w_i C_i$.

   (a) For the case when $w_i = 1$ for all i, show that choosing the shortest job first is optimal.

   (b) (Harder) Give an efficient algorithm that finds a schedule with minimum weighted completion time given arbitrary weights.