1. **Another MST Algorithm**

   In this question you will design a new, greedy, algorithm for computing the minimum spanning tree (MST) T of a connected, weighted, undirected graph $G = (V, E)$, with edge weights $w_{(u,v)}$ for edges $(u, v) \in E$.

   (a) Consider the following Lemma:

   > **Lemma.** Consider any cycle in G, and let (u, v) be the edge in that cycle with maximum weight. There exists an MST of G that does *not* include edge (u, v).

   Assuming that this lemma is true, design an algorithm that computes an MST of G by greedily/iteratively deleting edges.

   (b) Prove the lemma from part (a).

2. **Points in Space**

   Imagine a set of n points $X = \{x_1, \ldots, x_n\}$ in some unknown space. The only thing we know about them is the distance $d(x, y)$ between each pair of points x and y.

   We would like to compute a partition of X into disjoint sets of points $C_1, \ldots, C_k$ so as to maximize the minimum distance between any two points in different clusters. Each cluster must contain at least one point, and their union is the set X. More formally, we want a partition $\{C_1, \ldots, C_k\}$ of X that achieves

$$\max_{C_1,\ldots,C_k} \min_{\substack{x \in C_i \\ y \in C_j \\ i \neq j}} d(x, y)$$

   Give an efficient algorithm for this task.

   Hint: Try using an MST algorithm as a sub-routine!