Fundamental Algorithms, Section 003
Homework 12, Fall 22. Due date: Wednesday, December 14, 4:55pm on Gradescope

1. Let $G = (V, E)$ be an undirected, weighted graph. Let U be a subset of the vertices. Give an algorithm to construct a minimum weight spanning tree T in which each vertex $u \in U$ is a leaf. You may assume there is such a spanning tree. Your algorithm should have the same asymptotic running time as one of Prim's or Kruskal's MST algorithms (your choice). Justify your running time, and explain why your algorithm is correct.
Hint. If you remove U from T what do you obtain with respect to the vertex set $V - U$?

2. The following statements about MSTs of weighted undirected graph $G = (V, E)$ are either true or false. For each statement, either prove it is true or give an example to show it is false.
a. Let e be the unique heaviest edge on a cycle C in G (i.e. every other edge on C has smaller weight than e). Then e cannot be part of any MST of G.
b. If e is part of some MST of G, then e must be a lightest edge across some cut of G (there may be other edges of the same weight as e crossing this cut).
c. Prim's algorithm works correctly when there are negative weight edges.
d. Let $U \subset V$ and let $H = (U, E \cap (U \times U)$ be the subgraph of G induced by vertex set U, i.e. including all the edges between vertices in U. Suppose that H is connected. Let $T = (V, D)$ be an MST of G. Then the forest $F = (U, D \cap (U \times U)$, the portion of T joining vertices in U, is contained in some MST of H.

Challenge problem. Do not submit.
In this problem you will show that the cost of the union find operations in Kruskal's algorithm is $O((m + n) \log^* n)$. Actually, an even smaller, but still non-linear bound can be shown.

Define the terminal rank of a node in the union-find forest to be its rank when it ceases to be the root of a tree (as the result of a merge).
a. Show that there are at most $n/2^r$ nodes of terminal rank r.

Let u be a non-root node in the union-find forest, and let v be its parent. Note that $\text{rank}(v) > \text{rank}(u)$. Therefore, whenever a node changes its parent due to path compression, the rank of its parent increases.

Let's group the terminal ranks into the interval $[0, 0]$ and intervals of the form $(k, 2^k]$, where $k = 0, 2^0, 2^{2^0}, 2^{2^{2^0}}, \ldots$, i.e. k consists of a tower of exponents of 2.
b. How often can a terminal rank r node u increase the rank of its parent while having its parent's rank remain in the same interval as its rank (i.e. $\text{rank}(u) = \text{rank}(\text{parent}(u))$)?
c. How many other nodes have the rank of their parent increase, but the parent rank is in a different interval?
d. Note that the cost of a path compression is $O(1 +$ number of nodes whose parent rank is increased). Deduce the desired bound.