1. (30 points) **(Finishing the Arborescence proofs)** You will complete the formal arguments for the proofs we only sketched in class.

   (a) (5 points) Recap the primal-dual algorithm (with reverse delete) for min-cost arborescence.

   **Solution:**

   *Proof.* $1 + 1 = 2$. □

   (b) (10 points) In the iterative step, recall that we find a minimal strongly connected component $S$ which has incoming arcs in the current solution. If such a component exists and does not contain the root $r$, then we raise its dual variable $y_S$ and proceed. Show that if we cannot find such a component, then the current solution (before reverse delete) is feasible.

   **Solution:**

   *Proof.* $1 + 1 = 2$. □

   (c) (15 points) Let $F^*$ be the final solution after reverse delete. Then, show that for any variable $y_S > 0$ (i.e., it has strictly positive contribution) to the dual, then $|F^* \cap \delta^-(S)| = 1$, i.e., we satisfy the relaxed complementary slackness condition with $\lambda = 1$. This should use the property of reverse delete, and also how we choose the minimal strongly connected components at any time to raise the dual.

   **Solution:**

   *Proof.* $1 + 1 = 2$. □

2. (15 points) **(Gap Example for Local Search)** In class, we saw that local search yields a 1/2-approximation for Max-$k$-Coverage. Now you will construct an example where it can be stuck at such a solution which is factor 1/2-off from the optimal.

   (a) (15 points) Indeed, we said that if we start with any arbitrary collection of $k$ sets, and keep making swaps as long as we improve the total coverage, we repeat until we stop. Construct an instance of max-$k$-coverage where, if we started off with a bad solution (you can choose this solution), the local search algorithm would not even make one improvement. That is, it stops there. Moreover, if this starting solution only covers 1/2 the number of elements of an optimal solution, then we would have shown a tight bad example for our local search analysis. (Hint: try to construct an instance where all the inequalities we used in our swap-based proof are almost tight. Indeed, if they were sloppy, then we could have done a better analysis).

**Solution:**

*Proof.* $1 + 1 = 2$. □

3. (10 points) **(Connectivity Problem)** Consider the following problem: we have a graph $G = (V, E)$, and edges have cost $c_e \geq 0$. Now, we have a set $S$ of senders, and a set $R$ of receivers such that $S \cap R = \emptyset$. The goal is to find a set of edges $F$ with minimum total cost $\sum_{e \in F} c_e$ such that each receiver $r \in R$ is connected to at least one sender $s \in S$ (it can be any sender, doesn't matter which).

   (a) (10 points) Design a 2-approximation algorithm for this problem. You may reduce it to some problem we've already studied in class.

   **Solution:**

   *Proof.* $1 + 1 = 2$. □

4. (20 points) **(Some Non-Approximability Problems)** We saw in class that the Steiner Tree and Steiner Forest had 2-approximation algorithms. Now we show that a slight change to the problem makes them quite different. Suppose we have a vertex-cost version of the problem. That is, we have a graph $G = (V, E)$ and each vertex has a cost $c_v \geq 0$ (and edges have no cost). We are given a root $r \in V$, and a set of terminals $T \subseteq V$. The goal is to find a set of vertices $V' \subseteq V$ such that in the sub-graph induced by $V'$ (i.e. take vertices in $V'$ and all edges between any pair of vertices in $V'$), the root is connected to every terminal.

   (a) (5 points) Show that if we have an $\alpha$-approximation for this problem, then we can use this to design an $\alpha$-approximation for the Steiner Tree problem also.

   **Solution:**

   *Proof.* $1 + 1 = 2$. □

   (b) (15 points) More interestingly, show that if we have an $\alpha$-approximation for this problem, then we can use this to design an $\alpha$-approximation for the Set Cover problem also. Using this and results mentioned in class, what is the factor of non-approximability you can prove for this problem?

   **Solution:**

   *Proof.* $1 + 1 = 2$. □