Greedy - Advanced: Written Problems

1. You are given a set of points $P = \{p_1, p_2, ..., p_n\}$ on the real line (you may assume these are given to you in sorted order). Describe an algorithm that determines the smallest set of unit-length closed intervals that contains all of the given points. For example, the points $\{0.9, 1.2, 1.3, 2.1, 3.0\}$ can be covered by $[0.7, 1.7]$ and $[2.0, 3.0]$. State the runtime of your algorithm.

2. Prove that your algorithm always returns the optimal result.

3. Consider a situation in which you have n skiers with heights $p_1, p_2, ..., p_n$, and n pairs of skis with length $s_1, s_2, ..., s_n$. The problem is to assign each person a pair of skis so that the overall difference between the height of a skier and the length of their skis is minimized. More precisely, if $\alpha(i)$ returns the index of the skis assigned to person p_i, we want to minimize:

$$\frac{1}{n}\sum_{i=1}^{n}(|p_i - s_{\alpha(i)}|)$$

   Consider two greedy algorithms that may solve this problem:

   1. Find the skier and skis whose height difference is minimized. Assign those skis to that skier and repeat.
   2. Pair the smallest skier with the smallest skis, second smallest skier to second shortest skis, etc. until all pairings are complete.

   One of these greedy algorithms is correct. You must do three things: Prove this problem has optimal substructure. Claim which of the two algorithms is correct. Then show the greedy choice property of your chosen algorithm by using an exchange argument.

4. Let's suppose we want to replicate a single file f over a collection of n servers $S_1, S_2, ..., S_n$. Each server has a cost of storing the file. Let c_i be the cost of storing f on server i. Let's also stipulate that $c_n = 0$ (to guarantee that the file can be stored on at least one server for free).

   Your problem is to figure what subset of the n servers you should store the file on in order to minimize cost (which we define in more detail now). Suppose the file is stored on a subset of server indices $L \subseteq [1, n]$ and that $n \in L$ (L cannot be empty). More formally, file f is on server i iff $i \in L$. When a user tries to access the file they start from some server S_j, and they attempt to access S_j, then S_{j+1} etc until they find the file (they will always find f on S_n in the worst case). We say this costs the user:

$$u_j = \left(\min_{k \geq j : k \in L} k \right) - j$$

For any particular set $L \subseteq [1, n]$, the total cost is thus:

$$cost(L) = \sum_{i \in L} c_i + \sum_{j \in [1,n]} u_j$$

First, provide a brute-force algorithm that solves this problem. What is the run-time and why? Then, provide a better algorithm that solves this problem.