1. (5 points) Consider the following divide and conquer algorithm:

**Input:** Undirected graph $G = \langle N, A \rangle$, $|N| = |\{v_1, \ldots, v_n\}| > 1$

**Output:** $\langle N, T \rangle$, $T \subseteq A$, a minimum spanning tree of $G$

- $N_1 \leftarrow \{v_1, \ldots, v_{n/2}\}$
- $N_2 \leftarrow \{v_{(n/2)+1}, \ldots, v_n\}$
- $A_1 \leftarrow A \cap (N_1 \times N_1)$
- $A_2 \leftarrow A \cap (N_2 \times N_2)$
- $A_3 \leftarrow A \setminus (A_1 \cup A_2)$ (the arcs of $A$ neither in $A_1$ nor $A_2$)

Use Kruskal’s Algorithm to compute $\langle N_1, T_1 \rangle$, a min spanning tree of $\langle N_1, A_1 \rangle$

Use Kruskal’s Algorithm to compute $\langle N_2, T_2 \rangle$, a min spanning tree of $\langle N_2, A_2 \rangle$

Let $e$ be a minimum cost edge of $A_3$

return $T = T_1 \cup T_2 \cup \{e\}$

Prove or give a counterexample to the

**Conjecture:** If $G = \langle N, A \rangle$ is connected, then $\langle N, T \rangle$ constructed above is a minimum spanning tree of $G$.

2. (6 points) Suppose that you must compute the shortest path between a pair of nodes of a graph many times. After studying the problem, you realize that a spanning tree contains exactly one (simple) path between every pair of nodes. You also realize that finding the shortest path between a pair of nodes in a spanning tree is easier than finding the shortest path between a pair of nodes in a general graph. Consider the following algorithm:

Given undirected graph $G = \langle N, A \rangle$ and function $\text{length} : A \rightarrow R^+$, compute

minimum spanning tree $\langle N, T \rangle$ of $G = \langle N, A \rangle$

For any $v, w \in N$, to find the shortest path from $v$ to $w$ in $G$, simply compute

and return the shortest path from $v$ to $w$ in $\langle N, T \rangle$.

Prove or give a counterexample to the following

**Conjecture 1:** For any $G = \langle N, A \rangle$ and any $v, w \in N$, the above algorithm always returns a shortest path from $v$ to $w$ in $G$.

**Conjecture 2:** For any $G = \langle N, A \rangle$, there exist $v, w \in N$ such that the above algorithm returns a shortest path from $v$ to $w$ in $G$. 
3. (5 points) (From Baase, *Computer Algorithms*) Suppose that array $T$ contains an infinite number of distinct integers in increasing order. Assume the input to your algorithm is an integer $x$ which appears in $T$ (that is, there exists an $n$ such that $T[n]=x$). Your algorithm should return $n$ such that $T[n]=x$. Describe an algorithm to solve this problem in worst-case time in $\Theta(\lg n)$. 