Note: You are supposed to give proofs to the time and processor bounds of your algorithms. Read the questions carefully before attempting to solve them.

1. (16 points) $T(V, E)$ is a given binary tree with $n$ nodes. For every node $v$ in the tree we are supposed to compute the number of nodes in the subtree rooted at $v$. Present an $O(n)$ time algorithm for this problem.
2. (17 points) Input are a sorted sequence \( X = k_1, k_2, \ldots, k_n \) of distinct elements and another element \( x \). The problem is to check if \( x \in X \). Present an algorithm for this problem that uses \( \sqrt{n} \) CREW PRAM processors and runs in time \( O(1) \).
3. (17 points) Input is an undirected graph $G(V, E)$ (with $|V| = n$) in the form of an adjacency matrix. The problem is to check if $G$ has a triangle as a subgraph. Present an $O(1)$ time algorithm for this problem that uses up to $n^3$ common CRCW PRAM processors.
4. (17 points) Input is a (not necessarily sorted) sequence $X$ of $n$ arbitrary real numbers. Present a Monte Carlo algorithm to output an element of $X$ whose rank in $X$ is $\geq \frac{1}{2}n$. Your algorithm should run in $O(\log \log n)$ time using up to $\log \frac{n}{\log \log n}$ CREW PRAM processors. Show that the output of your algorithm will be correct with high probability. (Recall that the rank of an element $x$ in a sequence $X$ is defined to be $|\{q \in X : q < x\}| + 1$.)
5. (17 points) $\pi_1$ and $\pi_2$ are two decision problems. It is known that $\pi_1 \preceq \pi_2$. Any instance of $\pi_1$ can be translated into an instance of $\pi_2$ in $n^3$ time such that the instance of $\pi_1$ has the answer yes iff the corresponding instance of $\pi_2$ has the answer yes. Here $n$ is the size of the instance of $\pi_1$. If $\pi_2$ can be solved in time $m^{\log m}$ on any instance of size $m$, can $\pi_1$ be solved in $O(n^{\log n})$ time using the above reduction on any instance of size $n$?
6. (16 points) The subset sum decision problem takes as input a set $S$ of $n$ elements and an element $M$. The problem is to check if there exists a subset $S'$ of $S$ whose elements sum to $M$. For example, if $S = \{5, 11, 7, 13, 4, 8, 14\}$ and if $M = 28$, the answer is yes since the subset $\{5, 11, 4\}$ sums to 28. In the optimization version of the subset sum problem we are supposed to identify a subset $S'$ that sums to $M$. Assume that SUBSUM is a polynomial time algorithm for solving the subset sum decision problem. Show how you'll use SUBSUM to design a polynomial time algorithm to find a subset whose sum is $M$. 