1. Let $P$ be a problem. The worst-case time complexity of $P$ is $O(n^2)$. The worst case time complexity of $P$ is also $\Omega(n \lg n)$. Let $A$ be an algorithm that solves $P$. Which subset of the following statements are consistent with this information about the complexity of $P$? Briefly explain your answer.

(a) A has worst-case time complexity $O(n^{\frac{3}{2}})$.
(b) A has worst-case time complexity $O(n)$.
(c) A has worst-case time complexity $\Theta(n^2)$.

2. For each of these questions, briefly explain your answer.

(a) If I prove that an algorithm takes $\Theta(n^2)$ worst-case time, is it possible that it takes $O(n)$ on some inputs?
(b) If I prove that an algorithm takes $\Theta(n^2)$ worst-case time, is it possible that it takes $O(n)$ on all inputs?

3. Exercises 3.1-4; 3.2-4

4. Problems: 3-1; 3-2

5. Challenge Problem [Not to be turned in]: Recall: $\Theta(g(n)) = \{f(n) : \exists$ constants $c_1, c_2,$ and $n_0$ such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \forall n \geq n_0\}$. Show that $\{\lim_{n \to \infty}[f(n)] / [g(n)] = c; 0 < c < \infty\} \implies f(n) = \Theta(g(n))$. Show that the converse may not be true by providing a counter-example.