1. If there are two algorithms \( A \) and \( B \) which require only addition and multiplication, and if algorithm \( A \) requires 25% fewer multiplications than algorithm \( B \), then we can expect algorithm \( A \) to be substantially faster. True or false, and why?

2. You have a long list of records which are kept sorted according to some key. New records are being added all the time. What are both the strong and weak points of each of the following data storage options with regards to: lookup, addition of an element, full sort?

   2a. Keep the records in a sorted linked-list.
   2b. Keep the records in a sorted array.
   2c. Keep the records in a binary search tree (not necessarily balanced) implemented with pointers.
   2d. Keep the records in an AVL tree implemented with pointers.

3. For each of the following functions classify their complexity as one of: constant, linear, quadratic, cubic, \( \Theta(n \log n) \), polynomial of degree greater than three, exponential, worse than exponential, or none of the previous.
   i. \( h(n) = \sum_{k=1}^{n} 3^k \).
   ii. \( j(n) = \sum_{k=1}^{n} 2^k \).
   iii. \( g(n) = (n - 6) \log_2 n^5 \).
   iv. \( r(n) = 2^{\log_2 n} + \log_2 (\log_2 n) \).
   v. \( s(n) = n! \).

4. If we try to find the median of \( n \) elements in a data set by Lomuto partitioning what is \( C_{\text{worst}} \)? What is \( C_{\text{ave}} \)? Why is there a difference and what can be done about it?

5. Assume that \( G \) is a directed acyclic graph (DAG) in which each node represents a task. An arrow from task A to task B indicates that task A must be finished before task B can begin. How can you find an acceptable order in which to do the tasks using some type of traversal of the graph? Why to you need the graph to be acyclic?

6. Suppose we have to examine all subsets containing precisely four elements of an \( n \)-element set? How many subsets need to be looked at?

7. The algorithm \textit{merge-sort} is guaranteed to sort in \( \Theta(n \log n) \). What is its main drawback?

8. Suppose that a function \( M \) has initial value \( M(1) = 1 \) and recursion equation
   \[
   M(n) = 3M(n/4) + 1
   \]
   Find the complexity of \( M(n) \) in terms of \( \Theta \).

9. Suppose that a function \( M \) has initial value \( M(1) = 1 \) and recursion equation
   \[
   M(n) = M(0.9 \cdot n) + n
   \]
   Find the complexity of \( M(n) \) in terms of \( \Theta \).

10. How can be we convert a parse-tree for an algebraic expression such as \( 4 \times 7 + 3 \times 6 \times (-4)/9 \) into the postfix (RPN) form of the expression?

11. Given the matrix equation \( A\vec{x} = \vec{b} \) how do we use elementary row operations to solve the system (we are assuming that \( A^{-1} \) exists)? What is the complexity of the solution? How can we use this
method to also get a decomposition of $A = LU$ where $L$ is a lower-triangular matrix and $U$ is an upper triangular matrix? Why would this help us (in terms of complexity) if we had lots of matrix equations $Ax_i = b_i$ for $i = 1, 2, 3, \ldots, k$ to solve?

12. When we use a computer program to solve the matrix equation $Ax = b$ what are some other important considerations (e.g. condition number, choice of pivot row, etc.) which affect the accuracy and/or the run-time of the solution?