1. (20%) (Greedy) A telecom company needs to install base stations to cover all houses along a long road. These houses can be sparsely distributed along the road. Suppose the coverage is 5 miles per station. Design an optimal solution that covers all houses using as few base stations as possible. Prove that your algorithm is optimal.

2. (20%) (Linear programming) Solve the following linear program using SIMPLEX and show all the relevant steps:

   maximize \( x_1 + 2x_2 \)
   
   subject to
   
   \( 4x_1 - x_2 \leq 9 \)
   \( x_1 + x_2 \leq 8 \)
   \( 5x_1 - 2x_2 \geq -3 \)
   \( x_1, x_2 \geq 0 \)

   Provide a geometric explanation of the solution by plotting the corresponding feasible region in a 2-D space.

3. (20%) (Divide-and-conquer) Suppose the only way to access a database of student GPA is through a simple query \( k \) and that the system returns the \( k \)th smallest value that it contains. Design an algorithm that finds the median GPA from two separate databases \( A \) (with \( m \) values) and \( B \) (with \( n \) values) using at most \( \Theta (\log (m + n)) \) queries. Show explicitly how your solution meets the requirement. Note that the median GPA is the \( \lceil (m + n)/2 \rceil \)th smallest value in \( A \) and \( B \).

4. (20%) (Brute-force) Let \( G = (V, E) \) be a \( k \)-nary tree with \( n \) nodes. The distance between two nodes in \( G \) is the length of the path connecting these two nodes (neighbors have distance 1). The diameter of \( G \) is the maximal distance over all pairs of nodes. Design a linear-time solution (i.e. \( \Theta(n) \)) to find the diameter of \( G \).

5. (20%) (Dynamic programming) Design an optimal solution using dynamic programming for the general coin changing problem. Let a coin of denomination \( i, 1 \leq i \leq n \), have value \( d_i \). Use the example with three coins with values 1, 4, and 6 units to illustrate the correctness of your solution by showing optimal results for changes from 1 to 10.

6. (Bonus: 20%) Quicksort can be modified to find the \( k \)th smallest element from \( n \) elements so that in most cases it does much less work than is needed to sort the set completely.

   (a) Write a modified quicksort algorithm for this purpose.
   (b) Show that when this algorithm is used to find the median, the worst case is \( \Theta(n^2) \).
   (c) Develop a recurrence equation for the average running time of this algorithm.
   (d) Analyze the average running time of the algorithm. What is the asymptotic order?