Each problem is worth 10 points.

1. The problem is to take a given set of activities (intervals) and schedule these in the fewest number of rooms so that activities assigned to each room are mutually compatible. More precisely, the input is a set \( A = \{a_1, a_2, \ldots, a_n\} \) of intervals, where, for each \( i \), \( a_i = [\ell_i, r_i) \) such that \( \ell_i < r_i \). The output that is sought is the smallest collection \( \{C_1, C_2, \ldots, C_k\} \) of sets of intervals \( C_i \) such that \( \bigcup_{i=1}^{k} C_i = A \) and for each \( i \), \( C_i \) contains mutually compatible intervals. Consider the following greedy algorithm for this problem:

\[
\text{GreedyActivityScheduling}(A) \{
\text{Sort the activities in } A \text{ by increasing right endpoint and label the} \\
\text{intervals } a_1, a_2, \ldots, a_n \text{ in order.} \\
\text{for } i \leftarrow 1 \text{ to } n \text{ do} \\
\text{Find the smallest } j \text{ such that } a_i \text{ is compatible with every} \\
\text{interval in } C_j \text{ and add } a_i \text{ to } C_j;
\}
\]

Prove the correctness of this algorithm.
\textbf{Hint:} Proceed as follows. Suppose that the answer produced by the algorithm is \( \{C_1, C_2, \ldots, C_k\} \). Show that there is a point \( x \) and \( k \) intervals \( a_{i_1}, a_{i_2}, \ldots, a_{i_k} \) such that \( x \in a_{i_j} \) for each \( j = 1, 2, \ldots, k \). This means that any pair of the intervals in \( \{a_{i_1}, a_{i_2}, \ldots, a_{i_k}\} \) are mutually incompatible. This means that each of these has to be assigned to a distinct set \( C_i \). That in turn means that any solution to the problem contains at least \( k \) sets of intervals. Since we have a solution with \( k \) sets, it is optimal.

2. Consider the problem of finding a maximum size independent set in an arbitrary graph. Prove or disprove the correctness of the following greedy algorithm.

\[
\text{GreedyMaximumIndependentSet}(G) \{
S \leftarrow \emptyset; \\
\text{while } (G \text{ has at least one vertex}) \text{ do } \{ \\
\text{Find a vertex } v \text{ with minimum degree;} \\
S \leftarrow S \cup \{v\}; \\
\text{Delete from } G \text{ the vertex } v \text{ and all neighbors of } v;
\}
\text{return } S;
\}
\]

3. The graph coloring problem is to find a smallest set \( S \) of colors such that when each vertex of the given graph \( G \) is assigned a color from \( S \), no two neighboring vertices are assigned the same color. A given graph can be greedily colored as follows. Suppose that the palette of colors we want to use is \( \{1, 2, 3, \ldots\} \). Process the vertices in any order and to each vertex assign the smallest available color.

(a) Prove that if the given graph \( G \) has maximum vertex degree \( \Delta \), the above algorithm will use at most \((\Delta + 1)\) colors.

(b) Draw a tree that needs 3 or more colors if we color it using the above greedy algorithm. Briefly describe the running of the algorithm, with emphasis on why it needs more than 2 colors.

(c) Show a coloring of the above tree that uses only two colors.
4. Problem 17.3-2 on page 344.

5. Problem 23.2-7 on page 476.
   **Hint:** The diameter of an arbitrary graph can be computed in \( \Theta(|V|(|V|+|E|)) \) time by performing \(|V|\) breadth-first-search operations, one at each vertex. In a tree \(|E| = |V| - 1\) and therefore this simplifies to \( \Theta(|V|^2) \). However, by paying attention to the fact that the given graph is a tree the problem can be solved in \( \Theta(|V|) \) time. In particular, you only need to do 2 breadth-first-search operations.