Problem 1:

Idea:
1). Construct a graph from m judgements, mark each edge with ‘different’ or ‘same’ label.
2). Check vertice with a pair of edges labeled ‘same’. Add and remove edge following the rule shown in Figure.1. If newly added edge’s label is contradicted with existed edge’s label as Figure.2, return FALSE(inconsistent). Delete isolated vertice( with no edge connected).

3). Now, this new graph G’ will only contain vertice and edges with ‘different’ label.
4). Run BFS to traverse G’ starting from root as the solution for Problem 22.2.7 in HW4. All vertices with odd path lengths from the root are labeled as A and all the vertices with even path lengths from the root are labeled B.
5). Next we check every edge in G’. If any edge is between two vertices whose path lengths from root are both even or both odd, we return FALSE(inconsistent). Else return TRUE(consistent).

Time complexity is O(V+E).
**Problem 2:**

**Main Idea of Proof:**
1. If there are only 2 vertices and 1 edge, the best achievable bottleneck rate is the bandwidth of this edge.
2. If there are 3 vertices and n edges, construct a spinning tree following the rule that choosing the edges with bandwidth as large as possible.
   2.1). If the edge i with least bandwidth is not in the spinning tree, then it has no effect to the best achievable bottleneck rate of the graph (You can use contradiction to prove it for this case). Best achievable bottleneck rate of the graph will be the same as best achievable bottleneck rate of the spinning tree.
   2.2). If the edge i with least bandwidth is in the spinning tree, then best achievable bottleneck rate of the graph will be the same as best achievable bottleneck rate of the spinning tree, which is \( b(i) \).
3. If there are m vertices and n edges, still construct a spinning tree following the rule that choosing the edges with bandwidth as large as possible.
   3.1). Sort edges by their bandwidth from largest to least.
   3.2). Case A: the edge i with least bandwidth is the only edge connecting two sub-graph, then it has to be an edge in spanning tree. According to the definition of best achievable bottleneck rate, \( b(i) \) must be the best achievable bottleneck rate of the spanning tree. For the graph \( G \), the best achievable bottleneck rate should be \( \text{Min}\{b(\text{sub-G}1), b(i), b(\text{sub-G}2)\} \). Since \( b(i) \) is the least bandwidth, the best achievable bottleneck rate of \( G \) is also \( b(i) \).
   3.3). Case B: If the edge i with least bandwidth is in one sub-graph. As proved above, if i is in the spinning tree, best achievable bottleneck rate of the graph will be the same as \( b(i) \). If i is not in the spinning tree, it has no effect to the bottleneck rate of the graph.
   3.4). In case B, if i is not in the spinning tree, remove i to construct new graph \( G_1 \). From 3.3 we can get \( b(G) = b(G_1) \), continue to do 3.2-3.3 for the edge with the second least, third least bandwidth, until the original graph will be tightened in to the spinning tree (T). We get \( b(G) = b(G_1) = b(G_2) = \ldots = b(T) \), which means best achievable bottleneck rate of the graph will be equal to best achievable bottleneck rate of the spinning tree.

**Problem 3:**

1) From \( 2x_1 - x_2 + x_3 = 8 \), we can get \( x_3 = 8 - 2x_1 + x_2 \)

2) The objective function can be \( x_1 - 3x_2 + (8 - 2x_1 + x_2) \rightarrow -x_1 - 2x_2 + 8 \)

3) Convert to slack form:
   \[
   \begin{align*}
   \text{maximize} & \quad z = x_1 + 2x_2 - 8 \\
   \text{Subject to} & \quad x_3 = -2 + 2x_1 + x_2 \\
   & \quad x_4 = 10 - x_1 - 2x_2 \\
   \end{align*}
   \]

4) Formulate auxiliary LP
   \[
   \begin{align*}
   \text{maximize} & \quad z = -x_0 \\
   \end{align*}
   \]
subject to
\[ x_3 = -2 + x_0 + 2x_1 + x_2 \]
\[ x_4 = 10 + x_0 - x_1 - 2x_2 \]

5) Pivot \( x_0 \) and \( x_3 \)

maximize \( z = -2 + 2x_1 + x_2 - x_3 \)

subject to
\[ x_0 = 2 - 2x_1 - x_2 + x_3 \]
\[ x_4 = 12 - 3x_1 - 3x_2 + x_3 \]

6) Pivot \( x_0 \) and \( x_1 \)

maximize \( z = -x_0 \)

subject to
\[ x_1 = 1 - \frac{1}{2}x_0 - \frac{1}{2}x_2 + \frac{1}{2}x_3 \]
\[ x_4 = 9 + \frac{3}{2}x_0 - \frac{3}{2}x_2 - \frac{1}{2}x_3 \]

7) Remove \( x_0 \), use original objective function.

maximize \( z = \frac{3}{2}x_2 + \frac{1}{2}x_3 - 7 \)

subject to
\[ x_1 = 1 - \frac{1}{2}x_2 + \frac{1}{2}x_3 \]
\[ x_4 = 9 - \frac{3}{2}x_2 - \frac{1}{2}x_3 \]

8) Pivot \( x_2 \) and \( x_1 \)

maximize \( z = -3x_1 + 2x_3 - 4 \)

subject to
\[ x_2 = 2 - 2x_1 + x_3 \]
\[ x_4 = 6 + 3x_1 - 2x_3 \]

9) Pivot \( x_3 \) and \( x_4 \)

maximize \( z = 2 - x_4 \)
subject to

\begin{align*}
  x_2 &= 5 - \frac{1}{2} x_1 - \frac{1}{2} x_4 \\
  x_3 &= 3 + \frac{3}{2} x_1 - \frac{1}{2} x_4 
\end{align*}

10) Final maximum \( z = 2 \). The minimum objective value is \( -z = -2 \);
    Solution is \( [x_1, x_2, x_3] = [0, 5, 13] \).