1. **Bar Weighing Problem:** Assume we are given 12 gold bars numbered 1 to 12 where 11 bars are pure gold and one is counterfeit: either gold-plated lead (which is heavier than gold), or gold-plated tin (lighter than gold). The problem is to find the counterfeit bar and what metal it is made of using only a balance scale. Any number of bars can be placed on each side of the scale, and each use of the scale produces one of three outcomes, indicating either that both sides are the same weight, or which of the two sides is heavier.

   a. Give a decision tree argument to establish a lower bound on the (worst case) number of weighings which must be performed by any algorithm which solves this problem.

   b. Design an algorithm which solves this problem with the fewest possible (worst case) number of weighings. Represent your algorithm as a decision tree. (You may also write it in pseudo-code, though this is not required.)

2. **Bit String Problem:** Let $b$ be a bit string of length 5. The problem is to determine whether or not $b$ contains 3 consecutive 1’s, i.e. whether $b$ contains the substring 111. We consider algorithms for solving this problem whose only allowable operation is to peek at a bit. Obviously 5 peeks are sufficient. A decision tree argument proves the (useless) fact that at least 1 peek is necessary.

   a. Use an adversary argument to show that any algorithm which solves this problem must perform at least 4 peeks in worst case.

   b. Design an algorithm which solves this problem using 4 peeks in worst case. Represent your algorithm as a decision tree. (You may also write it in pseudo-code, though this is not required.)