Question 1 (Huffman Code, 30 points). Consider a text with the following letter frequencies:

- 'A' appears 5 times
- 'B' appears 6 times
- 'C' appears 20 times
- 'D' appears 8 times
- 'E' appears 15 times
- 'F' appears 2 times

Find the Huffman tree for an optimal Huffman encoding of this text.

We employ the standard greedy algorithm. A and F are the least common letters, so we combine them into a new node [A OR F] that appears 7 times. The nodes [A OR F] and B are now the least common, so we combine them into a node [A OR B OR F] that appears 13 times. The lightest pair is now [A OR B OR F] and D, which we combine into [A OR B OR D OR F] which appears 21 times. The lightest pair is now C and E which we combine into [C OR E] which appears 35 times. Finally, we combine [C OR E] with [A OR B OR D OR F] to get the root node.

The tree produced is as given below:

```
  F    A
   B
 /   \
D   C   E
```
Question 2 (Game Analysis, 35 points). Frank and Grace are playing a game. In this game, there is a pile of \( n \) stones. The players take turns and on a turn they remove any square number of stones up to the total number of stones in the pile. The loser is the player who is unable to make a move (in other words the person left with a pile of 0 stones). Devise an algorithm to determine who has a winning strategy for this game (assume that Frank goes first). Your algorithm should have runtime that is polynomial in \( n \) (although you do not have to prove this).

Hint: The game with \( n \) stones is winning for the first player if and only if there is some \( m \leq \sqrt{n} \) so that the game with \( n - m^2 \) stones is losing for the first player.

We note that the game with \( n \) stones is winning if and only if the game with \( n - m^2 \) stones is losing for some \( m \). We use a dynamic program where the subproblems are whether or not the first player can win the game with \( n \) stones for various values of \( n \). We have the following algorithm:

\[\text{IsWinning}(n) \ \text{// determines if Frank can win the game with n stones}\]
\[
\text{Initialize array A[0...n] setting all entries to False}\]
\[
\text{// A[i] is whether or not game with i stones is winning}\]
\[
\text{For i = 1 to n}\]
\[
\text{IsWinning = False} \ \text{// Whether or not we have found a way to win yet}\]
\[
\text{For m = 1 to floor(sqrt(n))}\]
\[
\text{If A[n-m^2] = False}\]
\[
\text{IsWinning = True}\]
\[
A[i] = \text{IsWinning}\]
\[
\text{Return A[n]}\]
Question 3 (Interval Covering Problem, 35 points). Consider the following computational problem. Given \( n \) points on the real line \( x_1 < x_2 < \ldots < x_n \), find a minimum number of unit length intervals \( I_1, I_2, \ldots, I_m \) that cover these points. In particular, each \( I_j \) should be an interval of the form \( [y_j, y_j + 1] \) (unit length), and for every \( 1 \leq i \leq n \) the number \( x_i \) should be an element of \( I_j \) for some \( j \) (the intervals cover the points). Among such coverings, we are looking at one with \( m \), the number of intervals, as small as possible.

It turns out that there is a greedy algorithm for this problem. In particular, after determining the first \( k \) intervals, we let \( I_{k+1} \) be \( [x_j, x_j + 1] \) where \( x_j \) is the smallest point not yet covered. So for example if the points were \( 0, 0.3, 1.5, 2.1, 3 \), we would have intervals \([0, 1], [1.5, 2.5], [3, 4]\). Prove that this algorithm produces an optimal solution. Hint: Show that the smallest interval in an arbitrary solution covers no more points than the smallest interval in the greedy solution.

Let \( I_1, \ldots, I_k \) be the greedy solution and \( I'_1, \ldots, I'_m \) by an arbitrary solution. Suppose that \( I'_1 \) is an interval in the second cover containing \( x_1 \). Note that the right endpoint of \( I'_1 \) is at most \( x_1 + 1 \). Therefore, \( I'_1 \) only covers \( x_i \)'s that are at most \( x_1 + 1 \). However \( I_1 \) covers all the \( x_i \)'s with this property (since none are less than \( x_1 \)). From here we note that \( I'_2, \ldots, I'_m \) must cover all of the \( x \)'s not covered by \( I_1 \). On the other hand, by induction on the number of points, \( I_2, \ldots, I_k \) is an optimal cover of these points. Thus, we must have \( m \geq k \).

As an alternative proof, one should note that if \( I_j \) has left endpoint \( x_{a_j} \) that it must be the case that \( x_{a_j + 1} > x_{a_j} + 1 \) (since otherwise it would be covered by \( I_j \)). This means that no two of the \( x_{a_j} \) can be covered by the same unit interval, and thus proves that any cover must have at least \( m \) intervals. Thus the greedy solution is optimal.