Problem One: Xzibit Words

One possible implementation is shown here:

```cpp
string mostXzibitWord(Lexicon& words) {
    /* Track the best string we've found so far and how many subwords it has. */
    string result;
    int numSubwords = 0;

    foreach (string word in words) {
        /* Store all the subwords we find. To avoid double-counting words, we'll hold this in a Lexicon. */
        Lexicon ourSubwords;

        /* Consider all possible start positions. */
        for (int start = 0; start < word.length(); start++) {
            /* Consider all possible end positions. Note that we include the string length itself, since that way we can consider substrings that terminate at the end of the string. */
            for (int stop = start; stop <= word.length(); stop++) {
                /* Note the C++ way of getting a substring. */
                string candidate = word.substr(start, stop - start);

                /* As an optimization, if this isn't a prefix of any legal word, then there's no point in continuing to extend this substring. */
                if (!words.containsPrefix(candidate))
                    break;

                /* If this is a word, then record it as a subword. */
                if (words.contains(candidate))
                    ourSubwords.add(candidate);
            }
        }

        /* Having found all subwords, see if this is better than our best guess so far. */
        if (numSubwords < ourSubwords.size()) {
            result = word;
            numSubwords = ourSubwords.size();
        }
    }

    return result;
}
```

In case you're curious, the most Xzibit word is “foreshadowers,” with 34 subwords!
Problem Two: RNA Protein Codes

Here is one possible implementation:

```cpp
Vector<string> findProteins(string rna, Map<string, string>& codons) {
    Vector<string> result;

    /* Track at which index we are in the string. We'll be going one character
     * at a time through the string.
     */
    int index = 0;
    while (true) {
        /* Find the next start codon, stopping if none are left. */
        index = rna.find("AUG", index);
        if (index == string::npos) {
            return result;
        }

        /* Keep decoding codons until we hit a stop codon. */
        string protein;
        while (true) {
            /* Read the codon. */
            string codon = rna.substr(index, 3);
            index += 3;

            /* If it's a stop codon, we're done with this protein. */
            if (codons[codon] == "stop")
                break;

            /* Otherwise, add it to the result. To get the commas right, we'll
             * only add commas if the string isn't empty. */
            if (!protein.empty()) protein += ", ";
            protein += codons[codon];
        }

        /* Add this protein to the result. */
        result += protein;
    }
}
```

A process similar to this one is actually going on right now in every single cell in your body. Isn't that amazing?