For `encode-symbol`, the base case executes in constant time, so the order of growth in the number of steps to encode a symbol is dependent on two things: the number of steps to perform `on-branch?` and the number of steps to perform the recursive call to `encode-symbol`.

`on-branch?` makes one call to `is-element-of?`, and the rest of its procedure calls execute in $O(1)$ steps. `is-element-of?` recursively iterates through the elements in a list until it finds the target element or reaches the end of the list. This takes $O(n)$ steps. Therefore `on-branch?` takes $O(n)$ steps.

As for the recursive call, the number of steps required depends on the structure of the tree and the position of the symbol in the tree.

In the case of a balanced tree, the worst-case (encoding the least frequent symbol) and the best-case (encoding the most frequent symbol) are at approximately the same depth. At every node, both branches will have contain approximately half the elements of the sub-tree, so at every recursive call the remaining problem is cut in half. This produces an order of growth in the number of recursive calls required to traverse the tree to any symbol of $O(log\ n)$ (the encoding will always be approximately $log_2\ n$ bits, the same as a fixed-length code).

Since each recursive call involves one call to `on-branch?`, the order of growth in the number of steps to encode a symbol in this type of tree is $O(n\ log\ n)$.

A tree of the type described in problem 2.71, where the weights of the symbols are $1, 2, 4, ..., 2^{n-1}$, is at the opposite end of the spectrum: it is maximally unbalanced; all symbols but the two least frequent occupy different depths in the tree. In this case, the order of growth in the number of recursive calls required to traverse the tree to the most frequent symbol is $O(1)$ (the encoding is always 1 bit), and the order of growth in the number of recursive calls to traverse the tree to the least frequent symbol is $O(n)$ (the encoding is always $n - 1$ bits).

So in this type of tree, the order of growth in the number of steps to encode the most frequent symbol is $O(n)$, and the order of growth in the number of steps to encode the least frequent symbol is $O(n^2)$.