Lexicographic order

The computer-science-y way to list all the three-element subsets of $S = \{a, b, c, d\}$ is to use a divide-and-conquer strategy. There are two kinds of three-element subsets of $S$, namely, those that contain $a$ and those that don't. The former consist of $a$ along with two other elements of $S$ while the latter consist of three elements of $\{b, c, d\}$. That is, the problem of listing all the three-element subsets of $\{a, b, c, d\}$ can be reduced to the problem of listing all the two-element subsets of $\{b, c, d\}$ and the problem of listing all three-element subsets of $\{b, c, d\}$. More generally, the problem of listing all the $k$-element subsets of $\{a_1, a_2, \ldots, a_n\}$ can be reduced to the problem of listing all the $(k-1)$-element subsets of $\{a_2, \ldots, a_n\}$ and the problem of listing all $k$-element subsets of $\{a_2, \ldots, a_n\}$. By applying this reduction recursively, we can generate all the $k$-element subsets of an $n$-element set. This is called **lexicographic order**. To see why this name is used, notice that in the case of $\{a, b, c, d\}$, we get the list $\{a, b, c\}$, $\{a, b, d\}$, $\{a, c, d\}$, $\{b, c, d\}$; this corresponds to the dictionary-ordering of the (nonsense) words *abc*, *abd*, *acd*, *bcd*.

Here's pseudocode to generate all the two-element subsets of $\{1, 2, \ldots, 10\}$ in lexicographic order:

```
for i from 1 to 10 do
  for j from i+1 to 10 do
     print i, j
```

We can make it slightly more efficient if in the outer loop we let `i` go from 1 to 9 (do you see why?). If we wanted more general pseudocode to generate all the two-element subsets of $\{1, 2, \ldots, n\}$ in lexicographic order, we'd replace 9 and 10 by $n - 1$ and $n$.

As a check on your understanding, can you write pseudocode to generate all the three-element subsets of $\{1, 2, \ldots, n\}$ in lexicographic order? For fun, implement it in code and run it with $n = 5$, to see if it correctly generates the correct number of subsets (agreeing with what you get for problem E in homework assignment #1).