Lab 4 **Mouna Kacimi, Werner Nutt,**
Camilo Thorne

4. Recursive Programs to Enumerate Subsets of a Set

This is an exercise to develop recursive programs that enumerate all nonempty subsets of a set. Write each program first in pseudo-code, then code it in Java. The results should be printed to standard output.
The finite set is represented as an array A of n characters.

1. Write a recursive program with the (Java) signature

   ```
   public static void subsets(char[] A)
   ```

   that computes the $2^n - 1$ nonempty subsets of A.
   If A consists of the characters a, b, and c, then `subsets` should return, e.g.,

   $$abc \quad ab \quad ac \quad a \quad bc \quad b \quad c$$

   (the order does not matter).

2. Modify program to a program `subsetsLex` that it prints the subsets in *lexicographic order*. If A consists of the characters a, b, and c, then `subsetsLex` should return, e.g.,

   $$a \quad ab \quad abc \quad ac \quad b \quad bc \quad c$$

   (now, the order matters).

Hint: Suppose your set A consists of $\{a, b, c, d\}$. To print the nonempty subsets of A, print first the subsets that contain a and then those that do not contain a. (Note that this can be achieved by *two* recursive calls.) To print the subsets that contain a, print first those that contain b and then those that do not contain b. And so on.

To print the subsets that do not contain a, print first the ones that contain b and then those that do not contain b. And so on. Finally, to print the subsets containing, say, a, c, d, just print a, c, d.
Generalise a procedure from this idea.
The examples above show that when making a recursive call, you need to remember which are the characters you will print for *all* sets covered by that call. To remember them, combine them into a string and pass the string as an argument.