**Problem summary**

Given an ordered alphabet of AA characters , print all words of length NN sorted according to that alphabet.

**Algorithm idea**

First , let us see how many words the solution will print. The answer is simply : AA raised to the N−thN−th power , or ANAN. That is because , for every position ppin a word ww , there are AA candidate characters.Considering the limits of the problem : N<=10N<=10 and A<=10A<=10 , there can be at most 1010 billionbillion words to print. However , that would require roughly 1 Gigabyte of data to store , so the real testcases were much simpler. This can be achieved on a reasonably-fast processor in the **ROSALIND** time limit of 5 minutes.

Next , we actually have to find the suitable way to find those words. The well-known technique that presents itself here is RecursionRecursion . The idea is to build a function ff that re-calls itself , or *recurses* , providing an *exit condition* : **some condition that tells the function when to stop moving to the next level of Recursion**. Please see the following Pseudo-Code for details.

**Pseudo code**

**function f ( length , word ) .. (1)**

**if length = N then .. (2)**

**print the word and exit**

**else**

**for a = 1 to A**

**f(length + 1 , word + alphabet[a]) ..(3)**

(1) ....(length) is the length of word formed so far , (word) is the word formed so far. (2) .... this is the exit condition , we have built a word of the required length , so print it and stop. (3) .... function f recalls itself because the current word is shorter than what is required , this action will increase the current word by one character , trying all possible characters in alphabetical order , so that the words are actually ordered.

**main block**

**input the A characters of the alphabet and N**

**f ( 0 , empty\_word) ... first call to the recursive function**

**end.**