Permutations of String

Finding a permutation of a string in Java means computing all possible new positions in a string by swapping characters. Also, in Java, the total number of permutations of a string is equal to the factorial of the length of a given string.

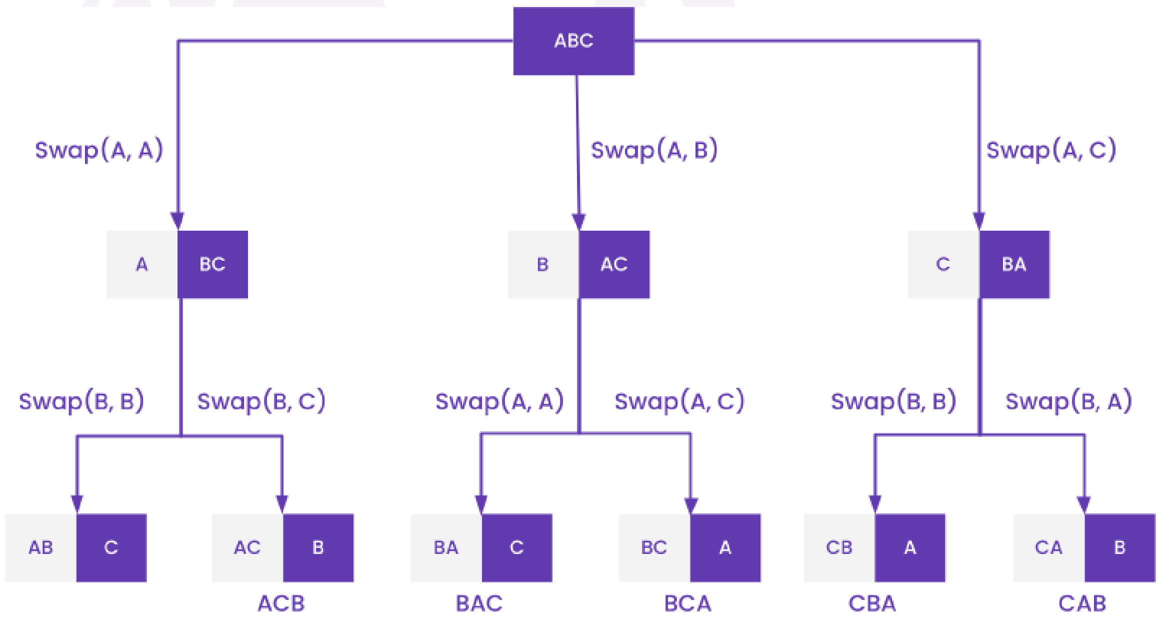
Example:
row XYZ has 3! that
6 permutation lines - [XYZ, XZY, YXZ, YZX, ZXY, ZYX]

Strings in Java are immutable and cannot be changed or modified, so a simple idea is to convert a string to an array of characters to generate permutations. You can use the backtracking concept by replacing each remaining character in the string with the first character and generating all permutations of the remaining characters with a recursive call.

This Java program uses a recursive or backtracking approach to print all possible string permutations.
The input string is converted to a character array using the toCharArray() function and passed to the resolver function. The simple idea is to replace the value in the character array with the index character element passed to keep track of the position of the string, then recursively call the solve function again to find the value (index+1).

The default condition fires when the currently passed index value equals (arr.length - 1), so prints the resulting array as one of the possible permutations of the string.

Replace back to return original value in for loop after recursive call to print all possible permutations. Since this is a return, the current character element can be used in any other possible string scheme.



Recursion Tree of String 'ABC'

Code: LP_CODE1.java