Challenge:

Write a method in Java that prints all permutations of the characters in a string.

Input: String object

Output: Display of all possible orderings of the characters in the string

Example:

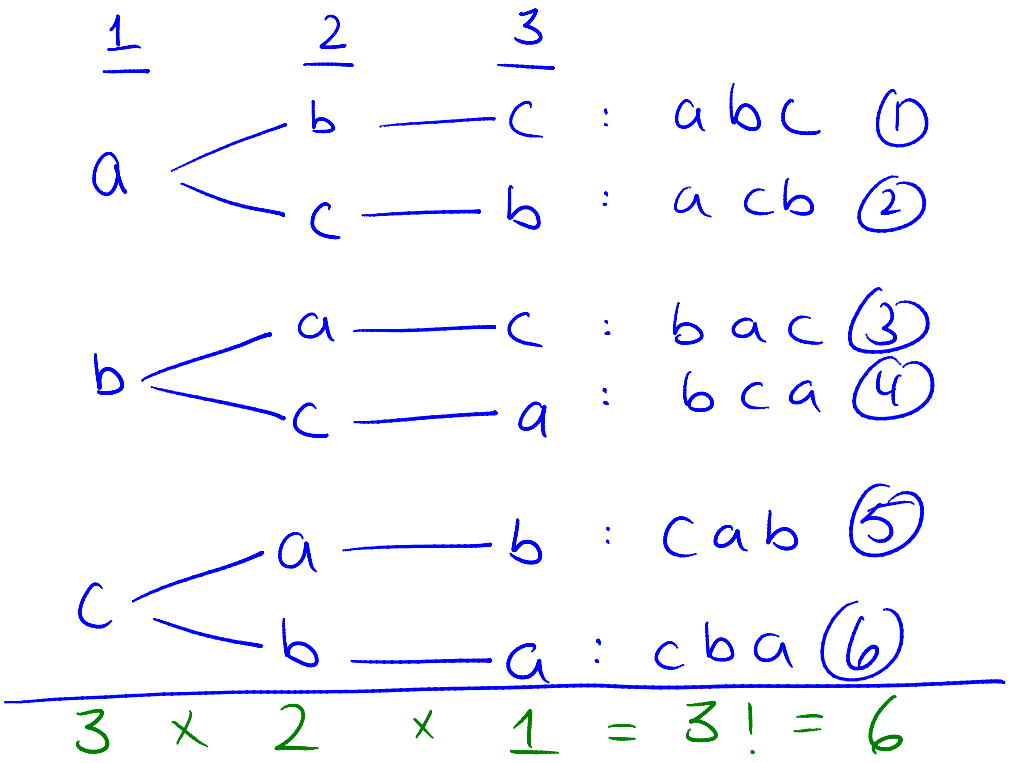
String: "bat"
Output: "bat","bta","abt","atb","tab","tba" (order/formatting does not matter)

String: "aa"
Output: "aa","aa" (each character is distinct)

Method Signature:

```
void permute(String str)
```

- Reverse Engineer Solution from Methodic Process "a b c"



- Number of possibilities = n!

- Formalize process:
  - Choose first character
  - Obtain all permutations with first character fixed
    - Choose second character
    - Obtain all permutation with first+second characters fixed

    ...

- Choosing characters:

- Character picked in previous recursion cannot be chosen again
- Scan previously chosen characters
  OR
- Array of booleans corresponding to positions of characters in string

▶ Algorithm

```
if we picked all characters in string:
    print permutation
else:
    for each character in string:
        if character already used
            skip character
        else
            place character in current position
            mark character as used
            permute remaning characters starting from position+1
            unmark character
```

▶ Code

```
public static void permute(String str){
    int length=str.length()

    boolean[] used=new boolean[length];

    StringBuffer output=new StringBuffer(length);

    permutation(str,length,output,used,0);
}


static void permutation(String str, int length, StringBuffer output, boolean[] used, int position){

        if (position==length){
            System.out.println(output.toString());
            return;
        }
        else{
            for(int i=0;i<length;i++){
                /* skip already used characters */
                if (used[i]) continue;

                /* add fixed character to output, and mark it as used */
                output.append(str.charAt(i));
                used[i]=true;

                /* permute over remaining characters starting at position+1 */
                permutation(str,length,output,used,position+1);

                /* remove fixed character from output, and unmark it */
                output.deleteCharAt(output.length()-1);
```

```
                used[i]=false;
            }
        }
    }
```