**Print all sub sequence of a string**

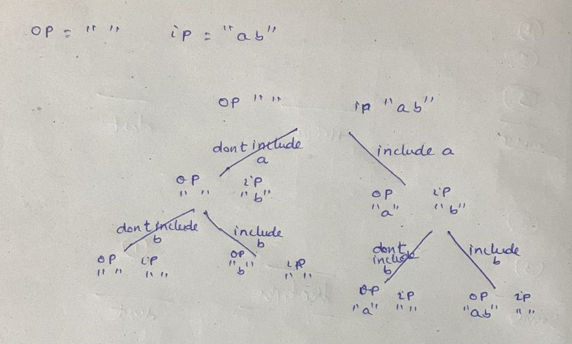
Input: “ab”

Op: “”, “b”, “a”, “ab”

**Sol:**

Take op = “” and ip=”ab” and draw the recursive tree diagram

In the diagram we can see that at leaf node when ip length == 0 we are getting result.



Take String op1 = op

String op2 = op

Op1 we are taking when we are making decision to not include the element into output and op2 we are taking for making decision to include the element.

When we decide to no include the element then op1 will be remain op1 only. but once we take the decision to include ‘a’ then op2 = op2 + ip.charAt(0); and ip = ip.subString(1)

Once we make both decisions then do the recursive call for each decision.

<https://github.com/hareramcse/Datastructure/blob/master/String/src/com/hs/leetcode/PrintAllSubSeq.java>

**Permutation with spaces**

**Given ip = “abc”**

Op: print all combination of abc with spaces in between letters.

**Sol:**

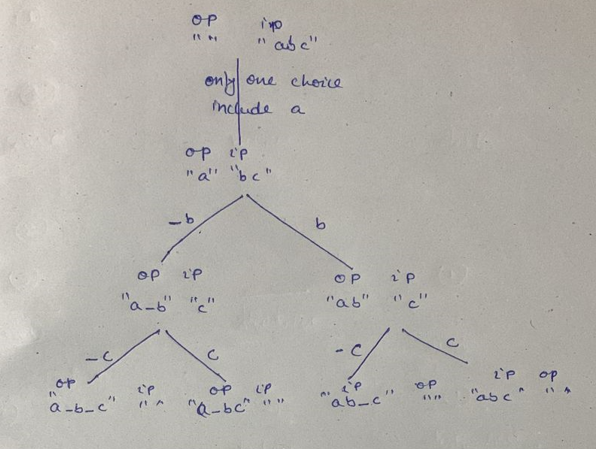
Here we are given that we cant add spaced before first char and in the last of the string. We can add the spaces in between letters.

So in the recursive choice diagram at first we have only one choice to include a in op string

Once we include ‘a’ in op string ip will be ‘bc’

Now we have 2 choices, whether we want to include the spaces or not.

From here it is like previous question.s

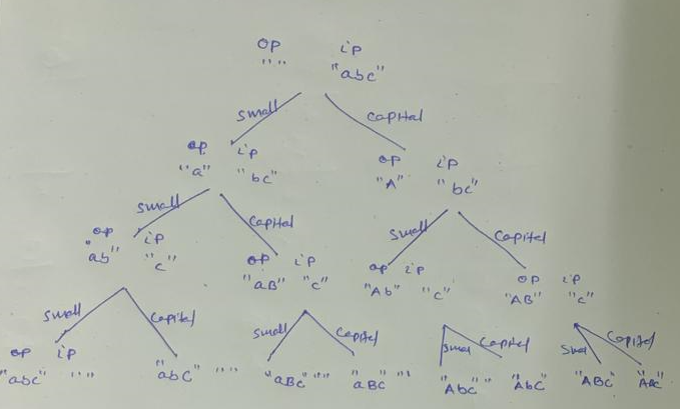


<https://github.com/hareramcse/Datastructure/blob/master/String/src/com/hs/leetcode/PermutationWithSpaces.java>

**Permutation with case change**

Given ip = “abc” we need to find out all the combination if we change the case of the letter

**Sol:**



Similiarly previous problem, here also we have 2 choices.

1st include with small case and another include with upper case

<https://github.com/hareramcse/Datastructure/blob/master/String/src/com/hs/leetcode/PermutationWithCaseChange.java>