Recursion on Strings

Wednesday, 14 February 2024 8:17 PM

↳ Integers → Number changed
↳ Arrays → Index changes ✗
↳ Strings → Index changes
OR
substring ✓

1) Print all subsequences of a string

str = "abc"

Subsequence

↳ May or may not contain any character, but the order is maintained.

abc ✓ bc ✓ c ✓ "" ✓
ab ✓ b ✓
ac ✓
a ✓
cb ✗
ba ✗ abc ✓
ca ✗ cab ✗

substring

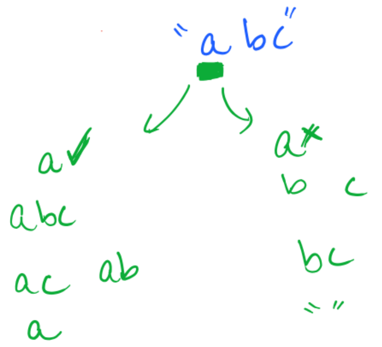
↳ Are contiguous & will not skip any character in between. si, ei can be different. Order needs to be maintained.

abc

a ✓ b ✓ c ✓
ab ✓ bc ✓
cb ✗
abc ✓ acb ✗
~~ac~~ ✗

Q. Print all the subsequences.

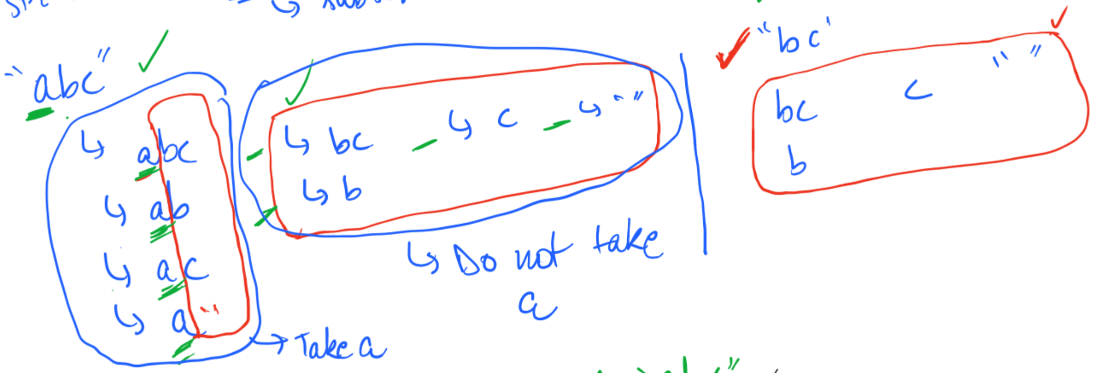
Print all the subsequences



↳ Find all the subsequences that have "a"

↳ Find all the subsequences that donot have 'a'

str (len) → 2^{len} ✓
↳ subsequences



BP: Find the answer of "abc",

SP: Find the answer of "bc" = $\boxed{}$

SW: $(a + \boxed{}) + (\boxed{})$

↓ Take 'a'

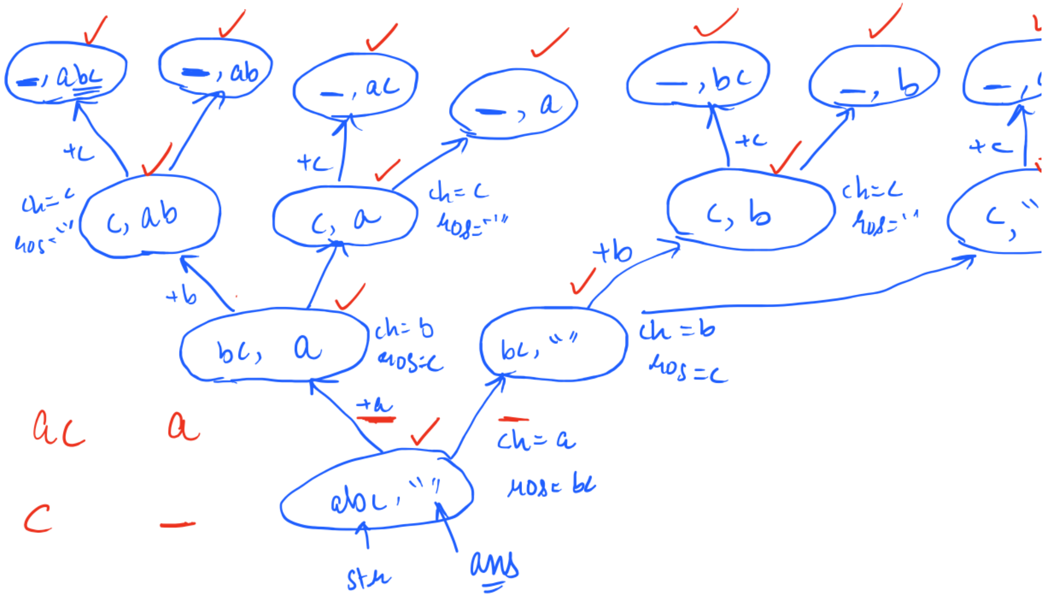
↓ Do not take 'a'

```
void subsequences(String s, String ans){
     if(s.length()==0){
       System.out.print(ans+" ");
       return;
     }
     // s -> abc
     char ch = s.charAt(0); //a
     String ros = s.substring(1); // bc

     // take a in the ans
     subsequences(ros, ans+ch); ①

     // do not take a in the ans
     subsequences(ros,ans); ②
}
```

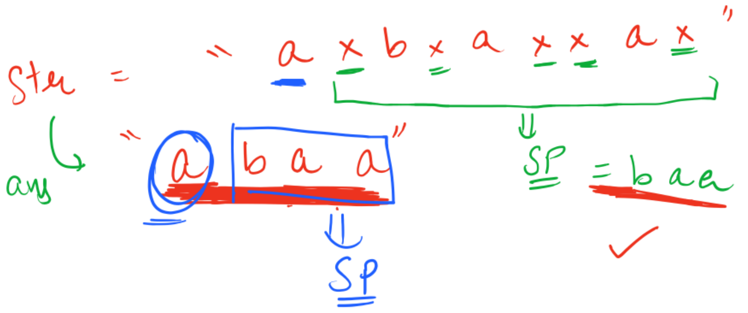
B.C



abc ab ac a

bc b c _

2) No x



Easy

BP: get ans from 0 to n-1 → SW+SP

SP: get ans from 1 to n-1

Is the ans after SW dependent on the ans of SP?

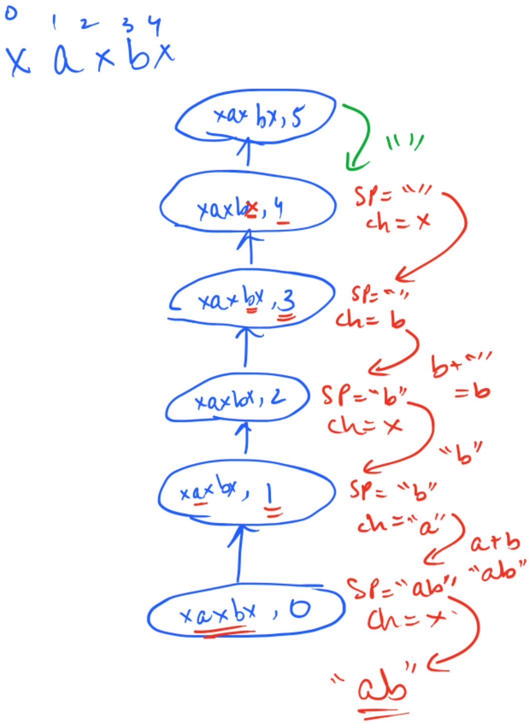
SW: add ch before SP → if not x

BC : invalid index

```
String removeX(String s, int idx){
    if(idx==s.length()){
      return "";
    }
    // SP
    String SP = removeX(s, idx+1);

    // SW
    char ch = s.charAt(idx);
    if(ch=='x'){
      return SP;
    }else{
      return ch+SP;
    }
  }
```

BC

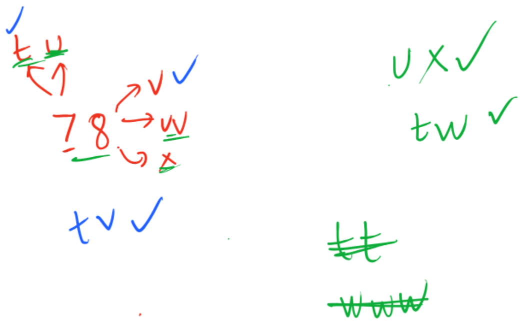


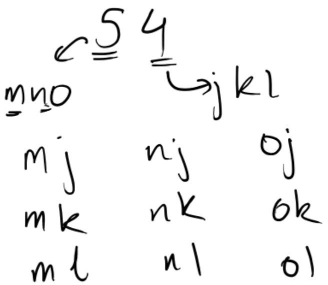
Keypad Combination

NOKIA 1100

0 -> .;
1 -> abc
2 -> def
3 -> ghi
4 -> jkl
5 -> mno
6 -> pqrs
7 -

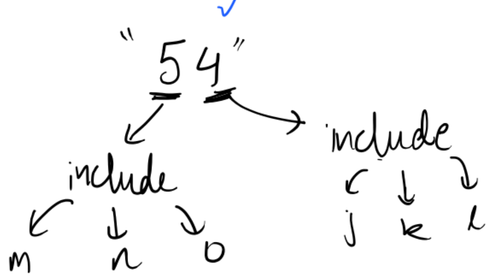
7 -> tu
8 -> vwx
9 -> yz

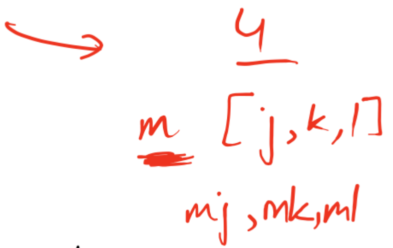




2
↳ include
↳ exclude

3 ↳ include m
↳ include n
↳ include o





BP → Find the ans of 54

SP → find the ans of 4

SW → Choose any one of 5's letters
↳ mno

BC → Invalid Index

```
static void printKPC(String ques) {
    KPC(ques,0,"");
  }
  static void KPC(String s, int idx, String ans){
    if(idx==s.length()){
      System.out.println(ans);
      return;
    }
    // SW
    int num = s.charAt(idx) - '0';
   String letters  = table[num]; // mno
    // choose 1 character each time
    for(int i=0;i<letters.length();i++){
      char ch = letters.charAt(i);
      // SP
      KPC(s,idx+1,ans+ch);
    }
  }
```

2

5

4

mno

→ m ✓

→ n ✓

→ o

+ n

j

k

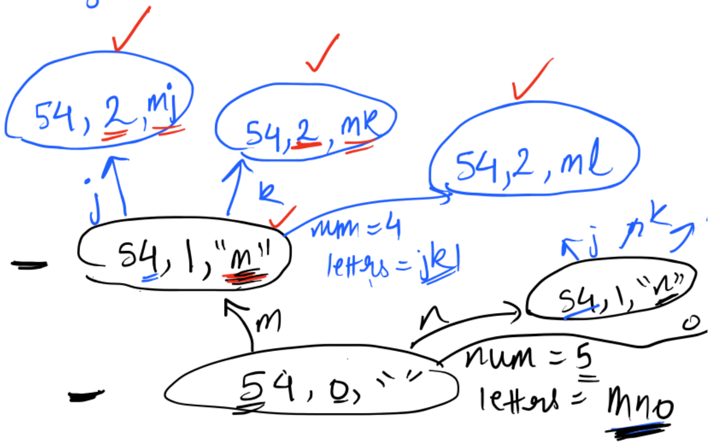
l

m , n , o

mj mk ml ✓

nj nk nl ✓

oj ok ol ✓



Print Stair Path

1

2

3

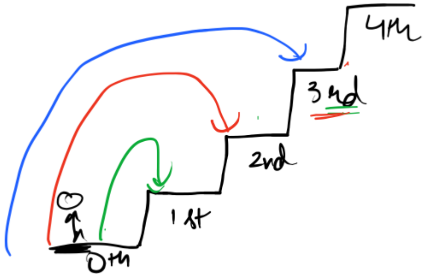
+ve Base

1111 ✓

-ve Base Case

5 ✗

✗



1 1 1 ✓

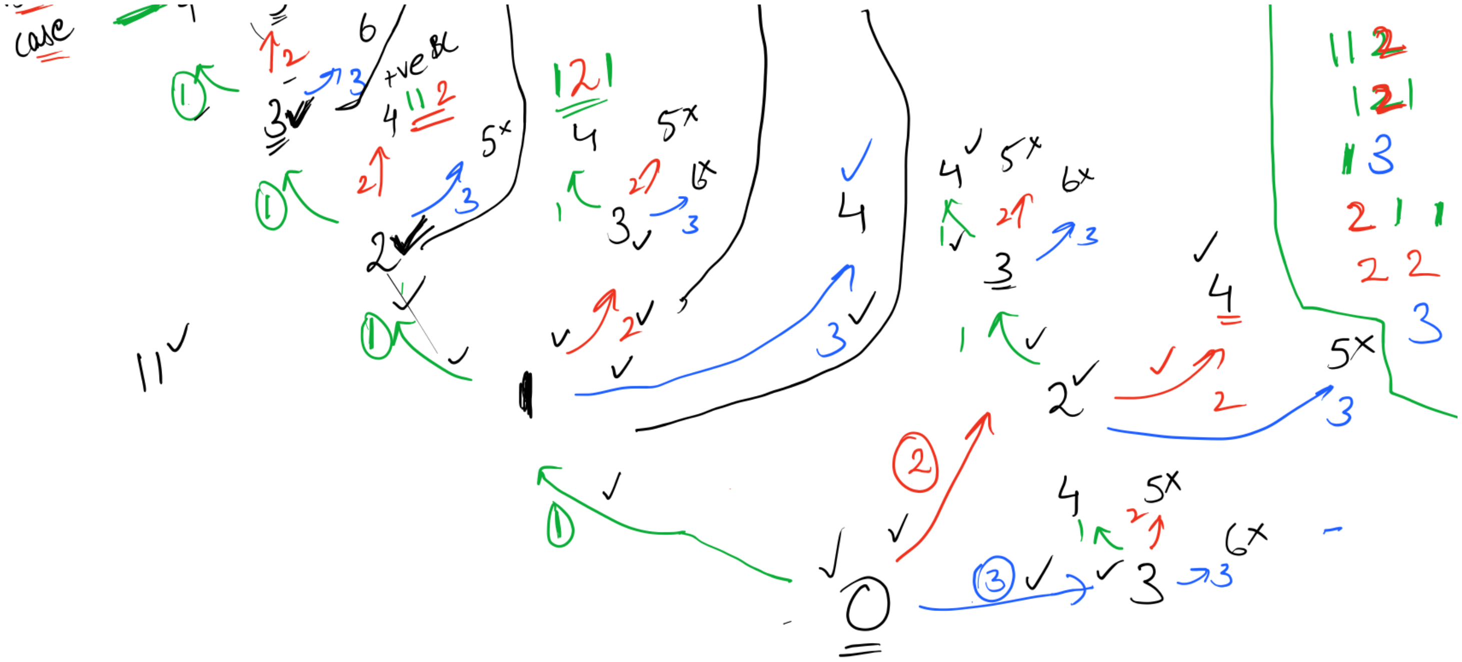
1 2 ✓ ✓

2 1 ✓ ✓

~~2 2~~ ✗

3 ✓

| | | | |


case
+ve BC
1 1 2
1 2 1
11
0
1
2
3
4
5×
6×
1 1 2
1 2 1
1 3
2 1 1
2 2
3