Q:- Given a string, print all subsequenc

of the string.

"abc"

↳ "",

"a", "b", "c",

"ab", "bc", "ac"

"abc"

→ $\{1, 2, 3\}$ $N=3$

↳ How many subsets we will be having ??

2^3 subsets

for N elements, no of subsets $= 2^N$

How this formula came up! ??

$$\{\overset{\overset{2}{\uparrow}}{a_1}, \overset{\overset{2}{\uparrow}}{a_2}, \overset{\overset{2}{\uparrow}}{a_3}, \overset{\overset{2}{\uparrow}}{a_4} \cdots\cdots\cdots\cdots \overset{\overset{2}{\uparrow}}{a_n}\} \longrightarrow \underline{\underline{2^n}}$$

Choice → either to be included in the subset

→ or not to be included in the subset.

[1, 2, 3]

↓

[ ]   [1]   [2]   [3]

[1, 2]   [1, 3]   [2, 3]

[1, 2, 3]

"abc"

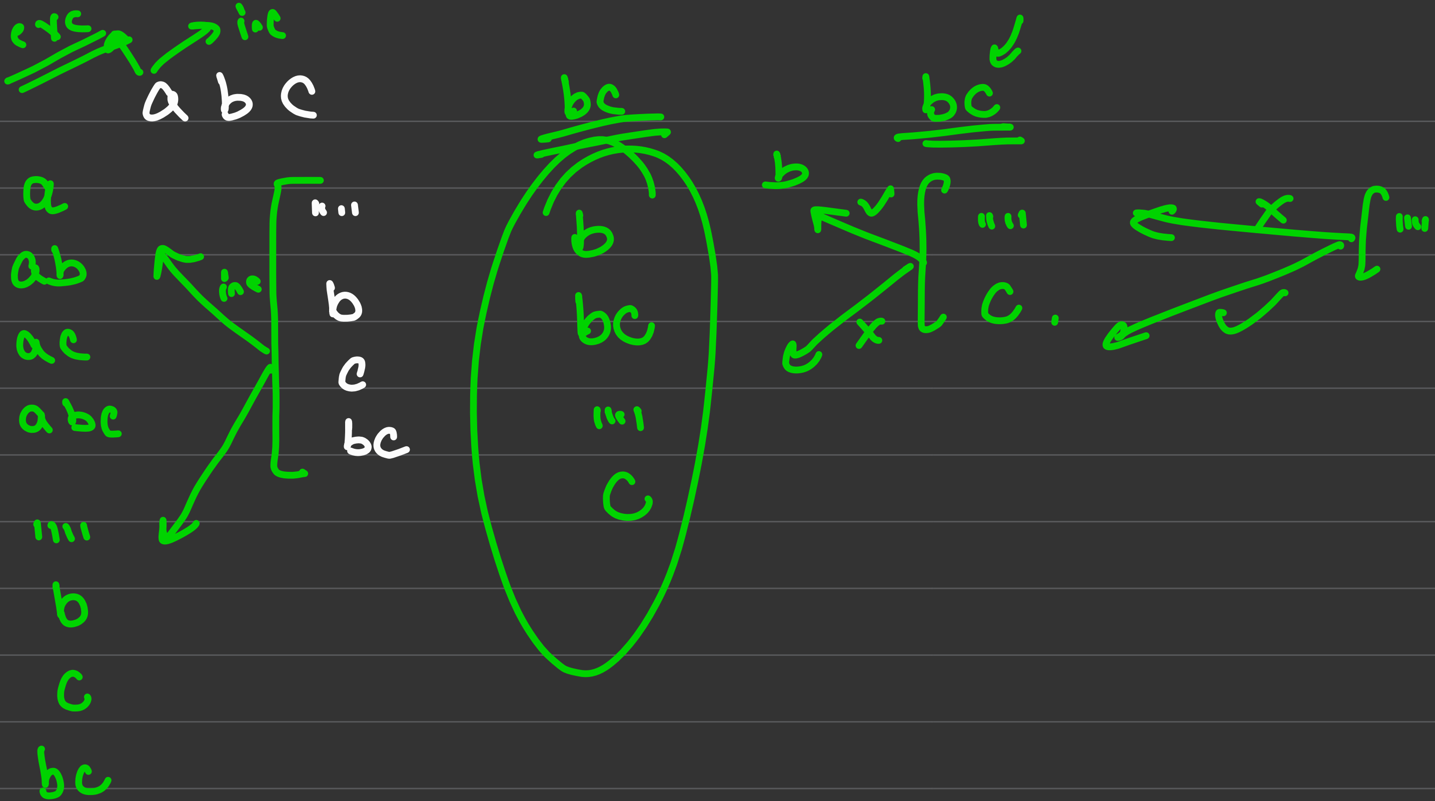
Base Case

if there is an empty string,

only one subsequence possible → ""

Recursion Assump

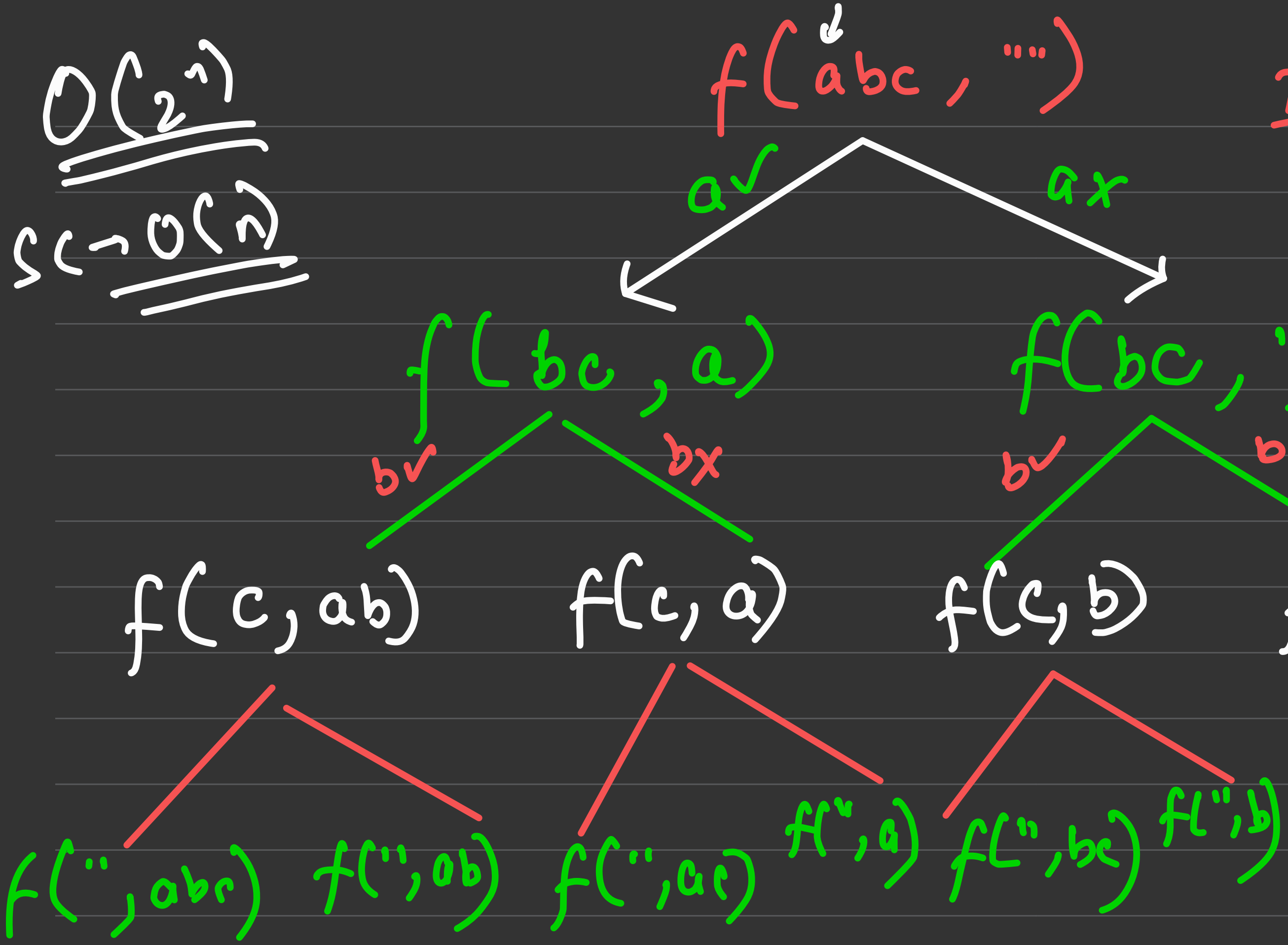
Self work


exc
inc
a b c
a
ab
ac
abc
....
b
c
bc
inc
....
b
c
bc
bc
b
bc
....
c
bc
✓
b
✓
....
x
c.
x
....
✓

$O(2^n)$

$SC \rightarrow O(n)$

f(str, output)



Ascii Subseq

ab →

""

"a"

"b"

"ab"

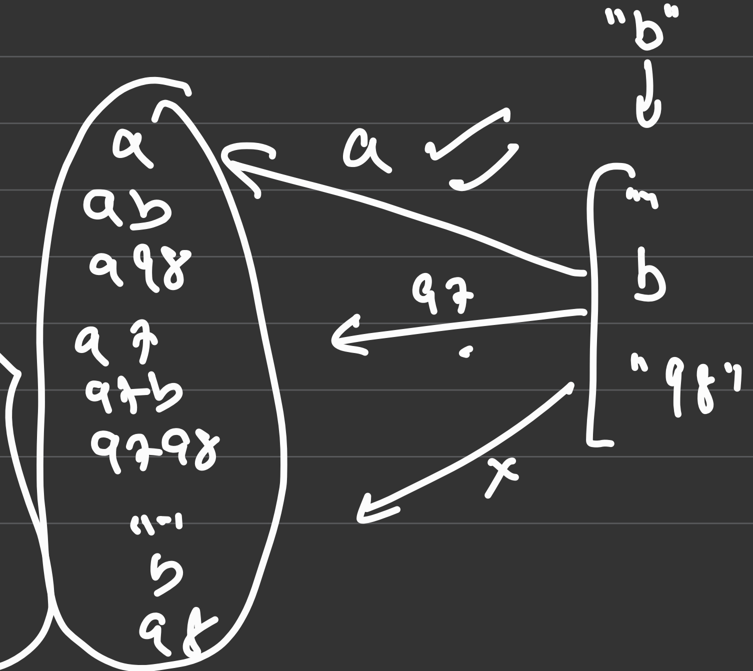
"97"

"98"

"97b"

"a98"

"9798"



Q:- Given a string, calculate all permutations of the string.

"abc"

↳

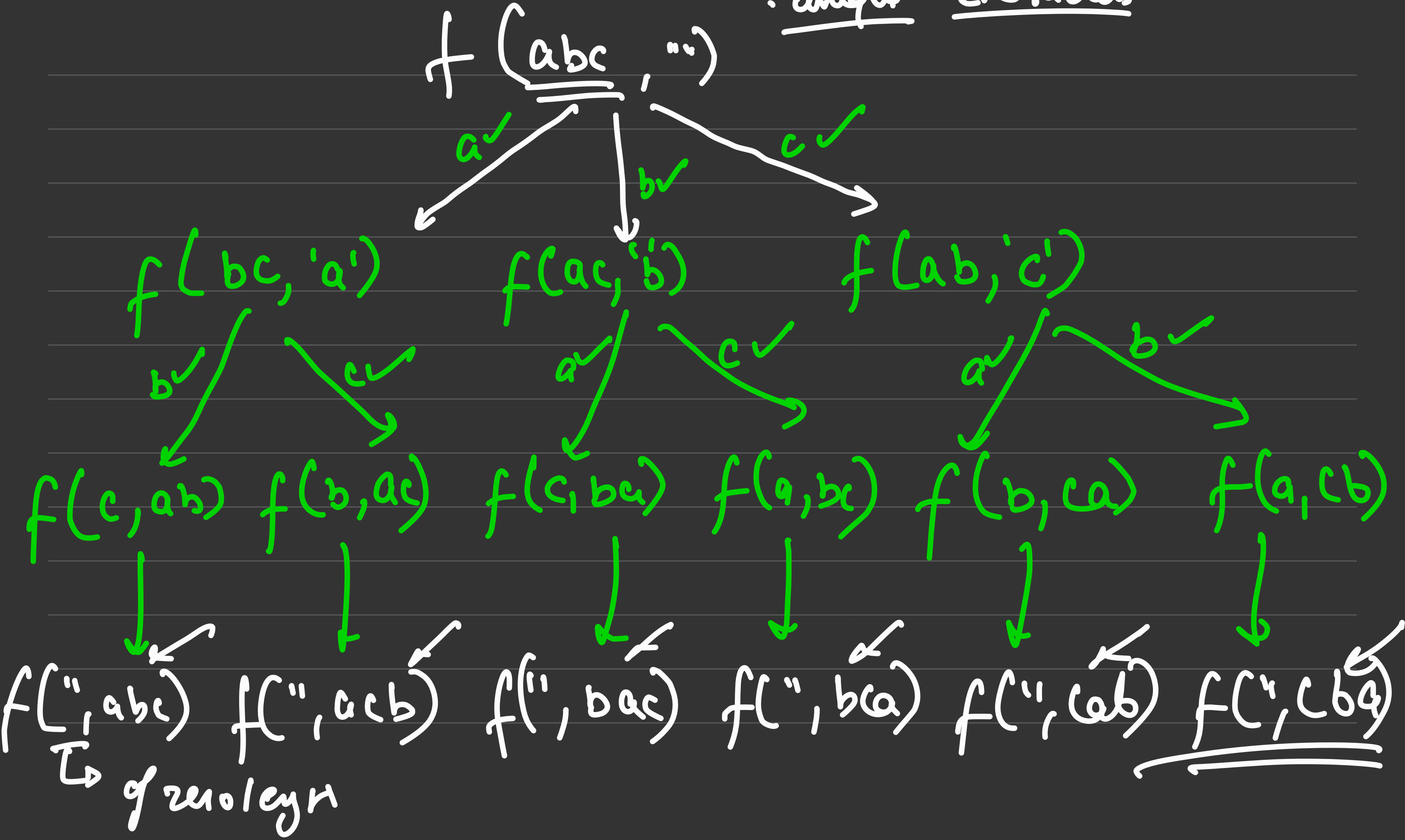
abc
acb
bac
bca
cab
cba

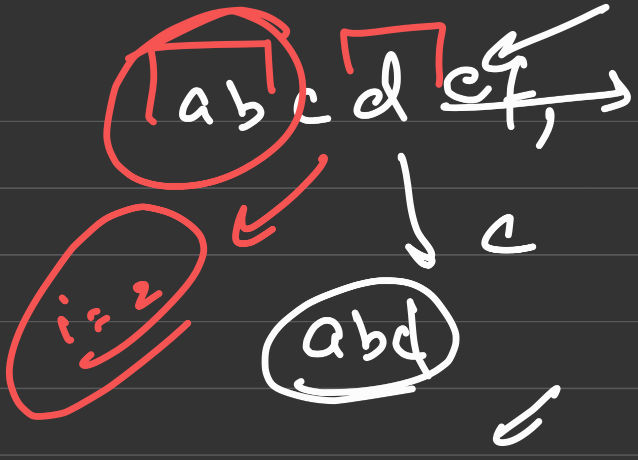
3!

Print

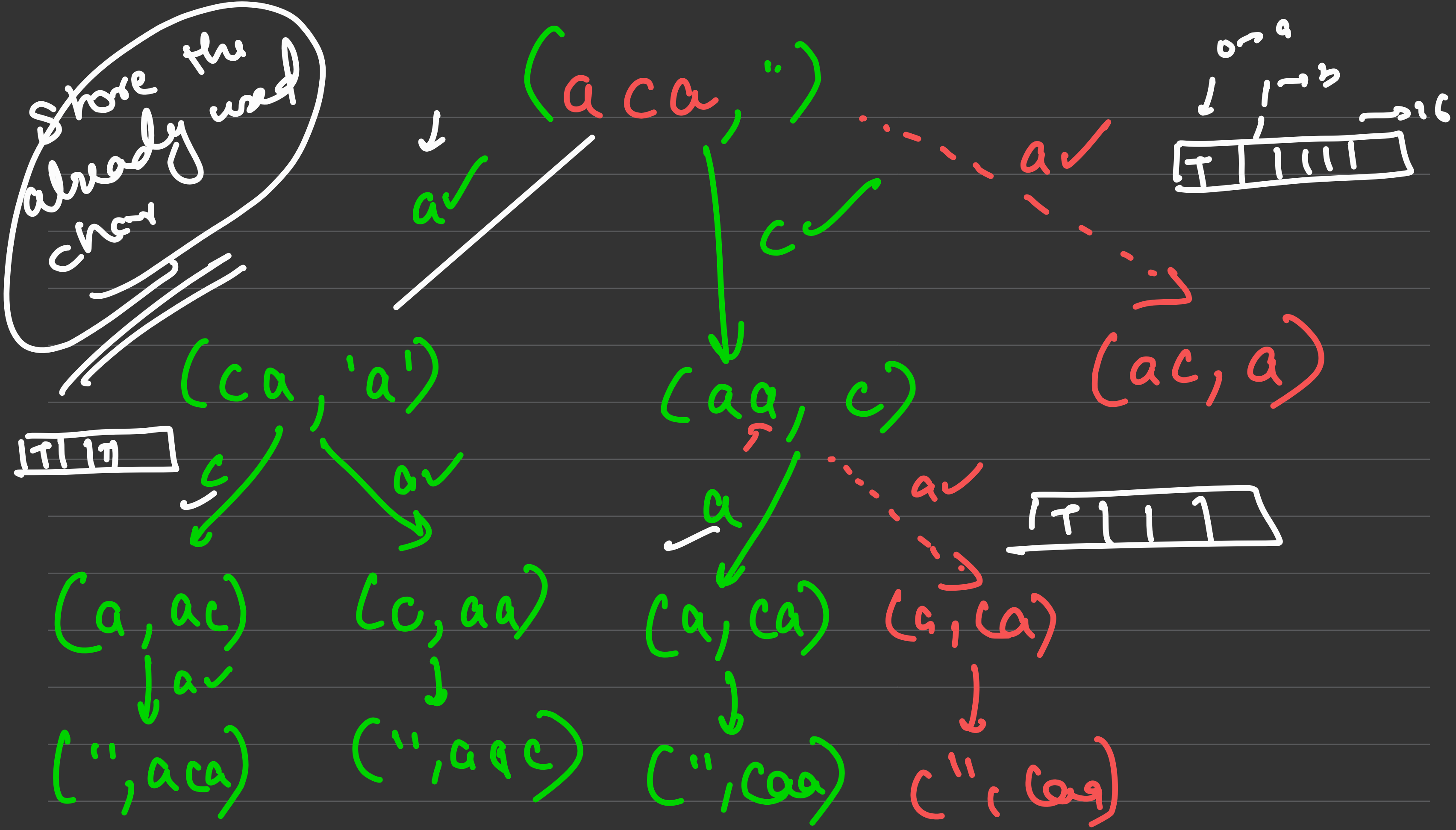
f(s)
↓
print permutations of s

· unique characters





substr(0, i) + substr(i+1);



Store the already used char

Q. Given a string of numbers, calculate all the codes of string.

→ "25114"

BEAAD ✓

YAAD ✓

YKD ✓

BEKD ✓

BEAN ✓

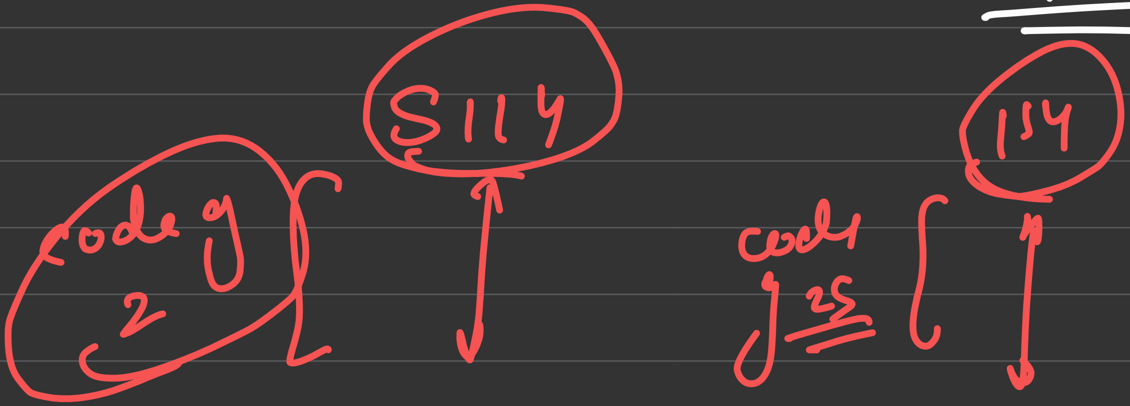
YAN ✓

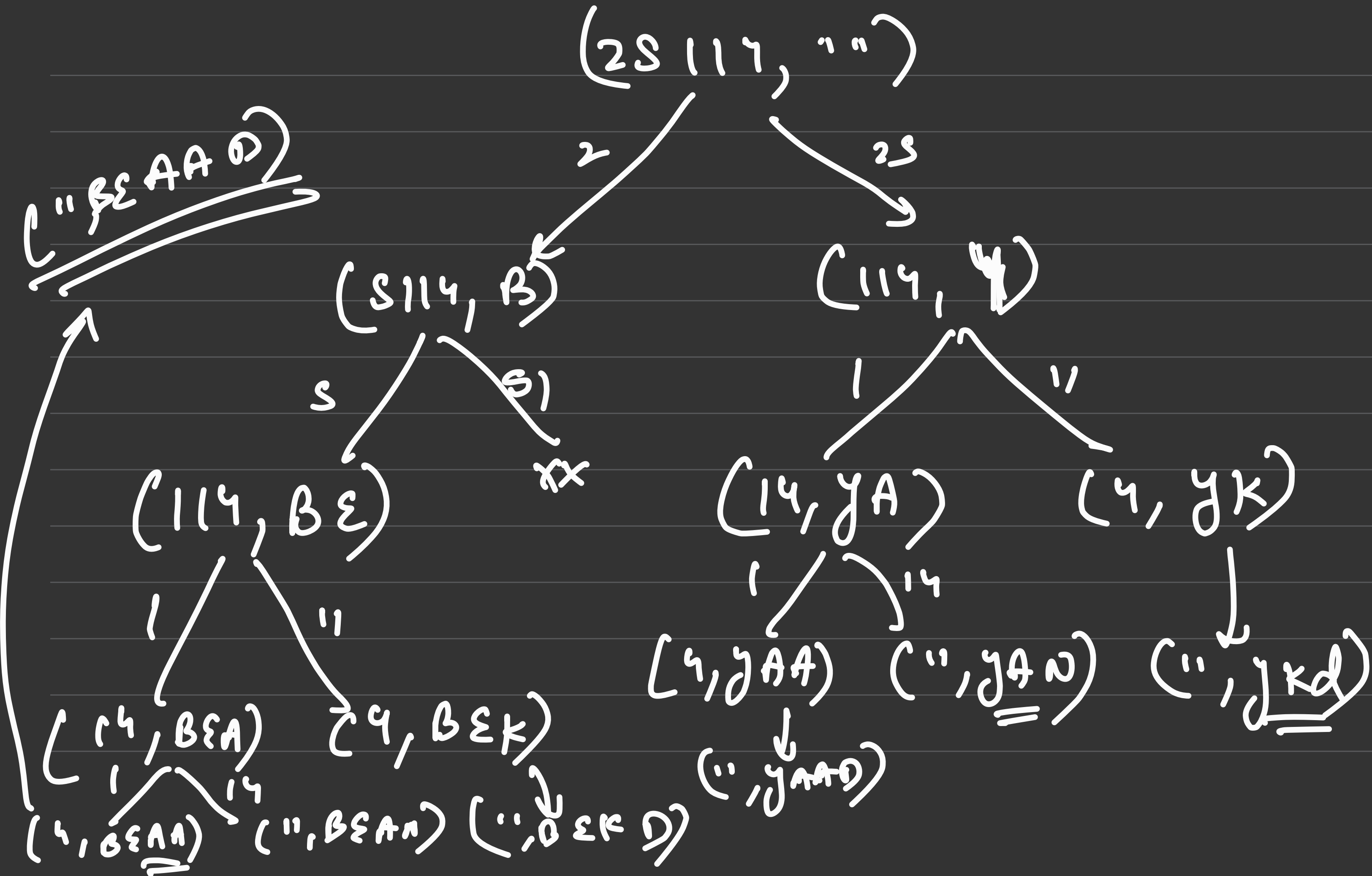
Print all the codes of string

→ 25114

Recursive assumption

go and calc codes of 5114 &
114




(25114, "")
2
25
("BEAAD")
(5114, B)
(114, Y)
5
51
XX
1
11
(114, BE)
(14, YA)
(4, YK)
1
11
1
14
(14, BEA)
(4, BEK)
(4, YAA)
("", YAN)
("", YKD)
1
14
("", YAAD)
(4, BEAA)
("", BEAN)
("", BEKD)