word k-selection-3 :

String : aabbac

k=3

- aaa
- aab
- aac
- abb
- abc
- bbc

k=3

characters → level

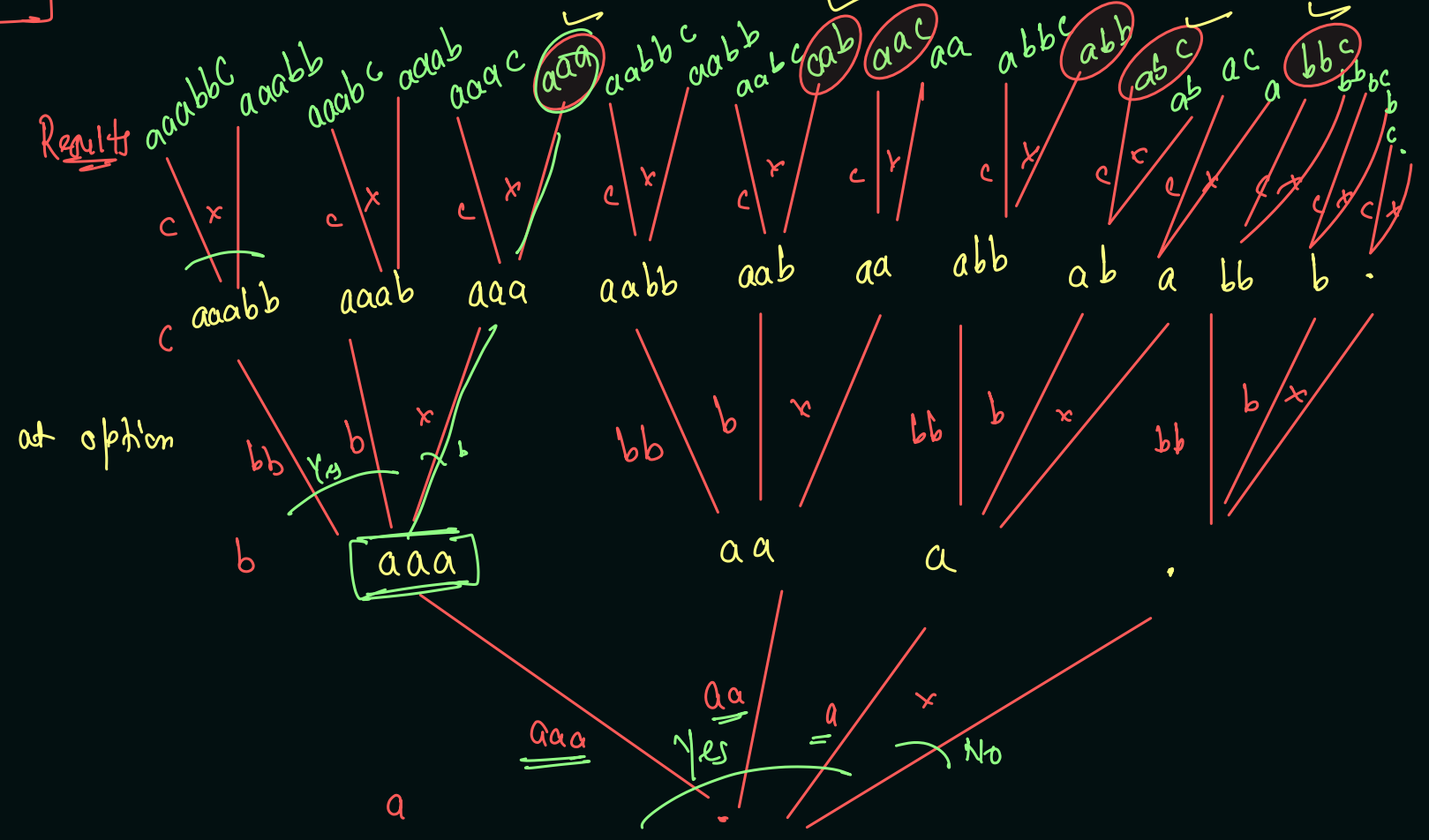
occurrence of character at option

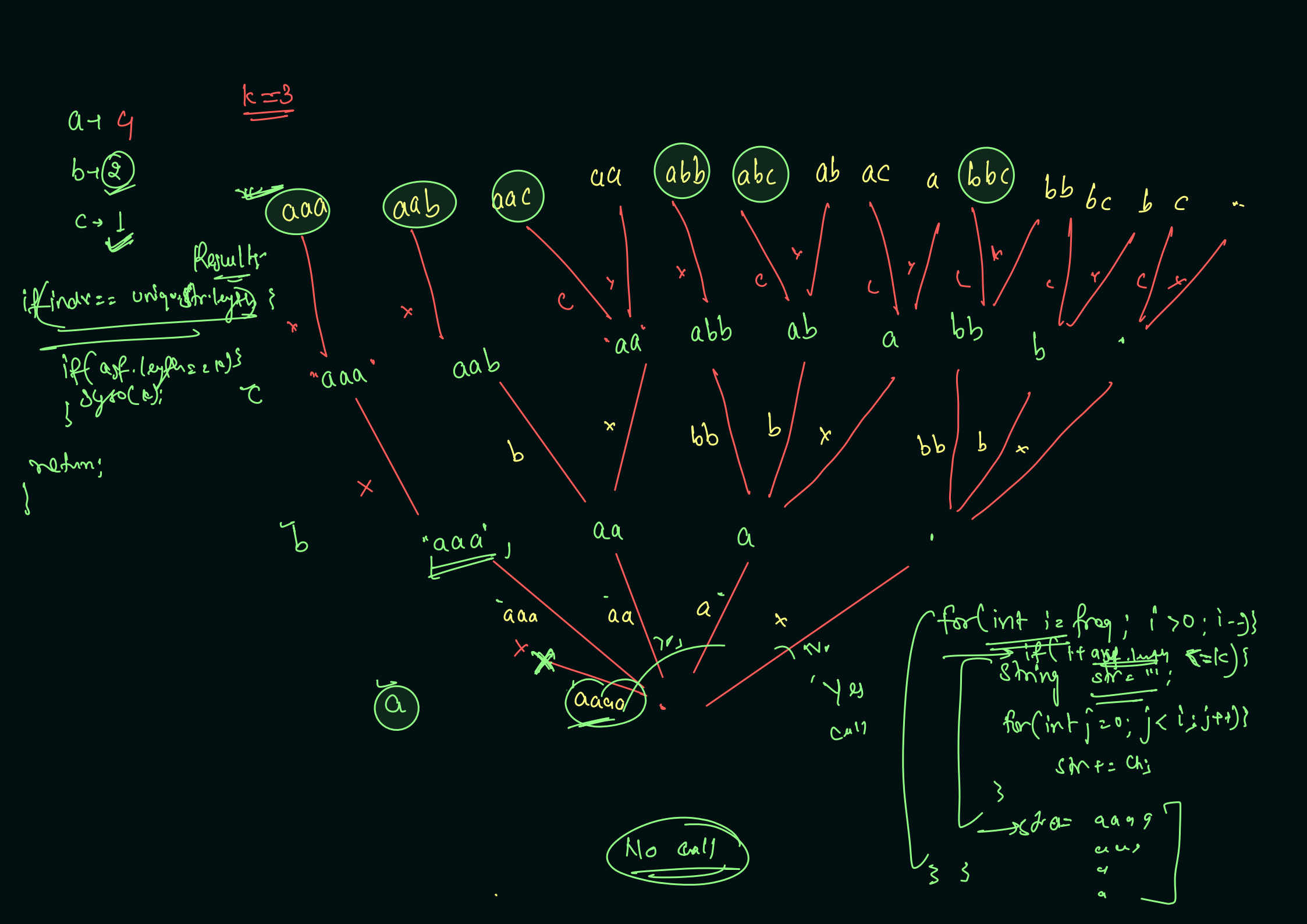
aaa
aab
aac
abb
abc
bbc

String → aabbac

unique string → abc

a → 2
b → 1
[c → 1]




k=3
a→4
b→2
c→1
Result:
if(indx == uniqueStr.length){
if(asf.length() == n){
System.out.println(asf);
}
return;
}
aaa
aab
aac
aa
abb
abc
ab
ac
a
bbc
bb
bc
b
c
"aaa"
aab
aa
abb
ab
a
bb
b
"aaa"
aa
a
aaa
aa
a
aaaa
No call
for(int i=freq; i>0; i--){
if(ifasf.length() <= n){
String str = "";
for(int j=0; j<i; j++){
str += ch;
}
Ja= aaaa
aaa
aa
a
}
}

word - k - selection - 4.

String → aab bac → aaa, aab, aac, abb, abc, bbc

level → Spot

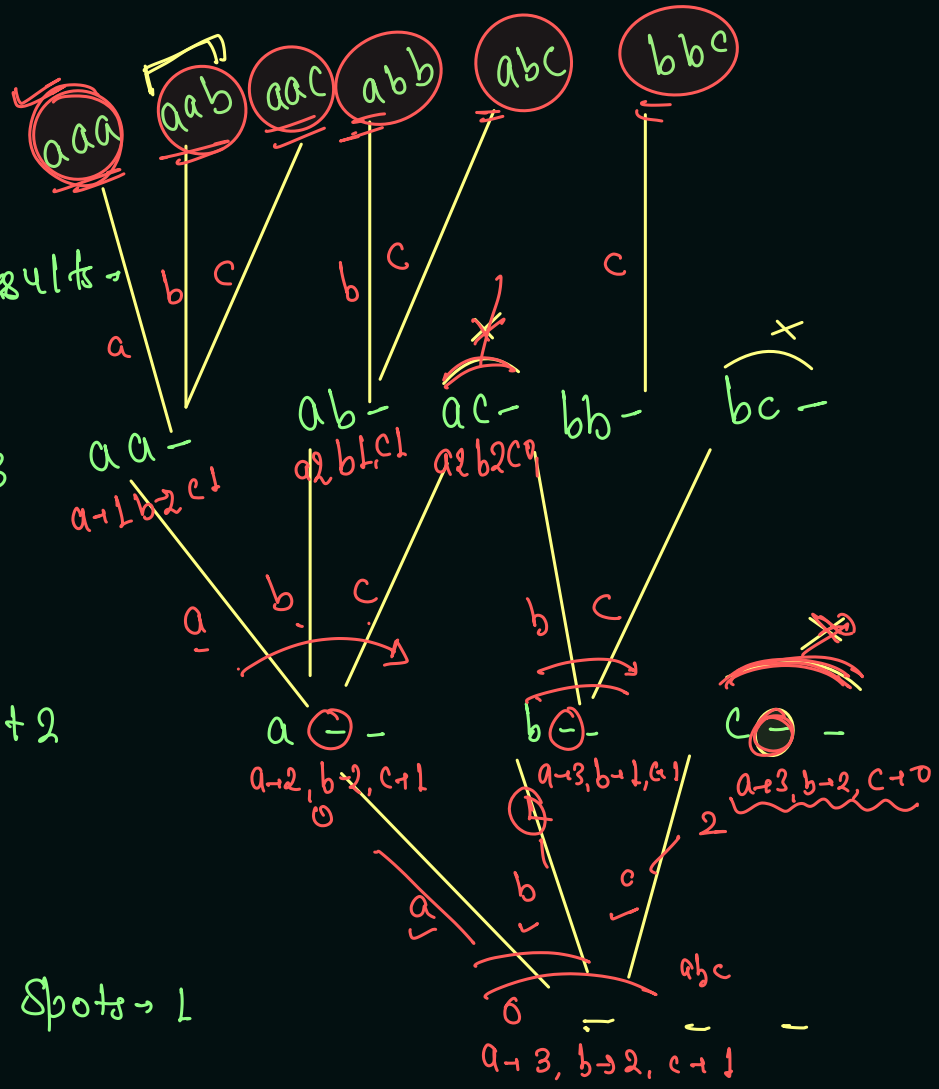
options → Characters.

k = 3.

Spot

unique string = abc

Results →



unique

words - k - length words : given a string, you have to select k characters (may be repeated) and print all possible selections.

what not to do?

$a_0\ a_1\ a_2$

0 1 2 →	$a_0 a_1 a_2$ = aaa	we have to avoid these kind of repetitions
0 2 1 →	$a_0 a_2 a_1$ = aaa	
1 0 2 →	$a_1 a_0 a_2$ = aaa	
1 2 0 →	$a_1 a_2 a_0$ = aaa	
2 0 1 →	$a_2 a_0 a_1$ = aaa	
2 1 0 →	$a_2 a_1 a_0$ = aaa	

String = aabbac

Combinations

aaa

aab

aac

abb

abc

bbc

$\frac{3!}{1!\,1!\,1!}$

aabbccc = $\frac{7!}{2! \times 2! \times 3!}$ =

Permute these combinations

$\frac{3!}{1!} = 1$

$\frac{3!}{2! \times 1!} = 3$

$\frac{3!}{2!} = 3$

$\frac{3!}{2!} = 3$

$3! = 6$

$\frac{3!}{2! \times 1!} = 3$

aaa, aab, aba, baa, aac, aca, caa, abb, bab, bba, abc, acb, bac, cab, bca, cba, bbc, bcb, cbb

19

String = aabbac

Unique string = abc

$a_0 a_0 b_0 \rightarrow aab$

$a_0 a_1 b_0 \rightarrow aab$

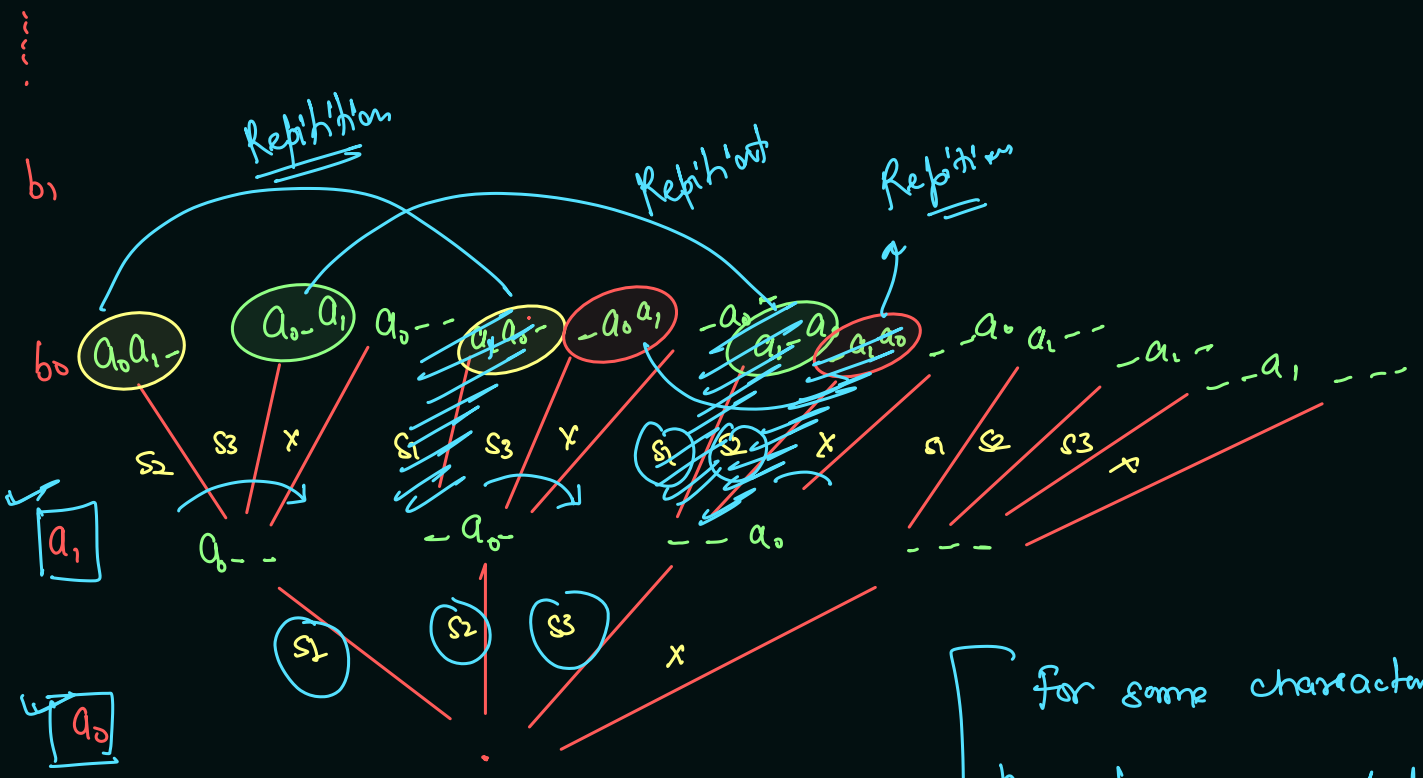
k = 2

Characts are at level

spots are option.

3-spots

↳ S1, S2, & S3



For some character we have to make sorted order of spots at next level,

Although we are solving permutation still it have possibility of repetition.

aabbac

$k = 3$

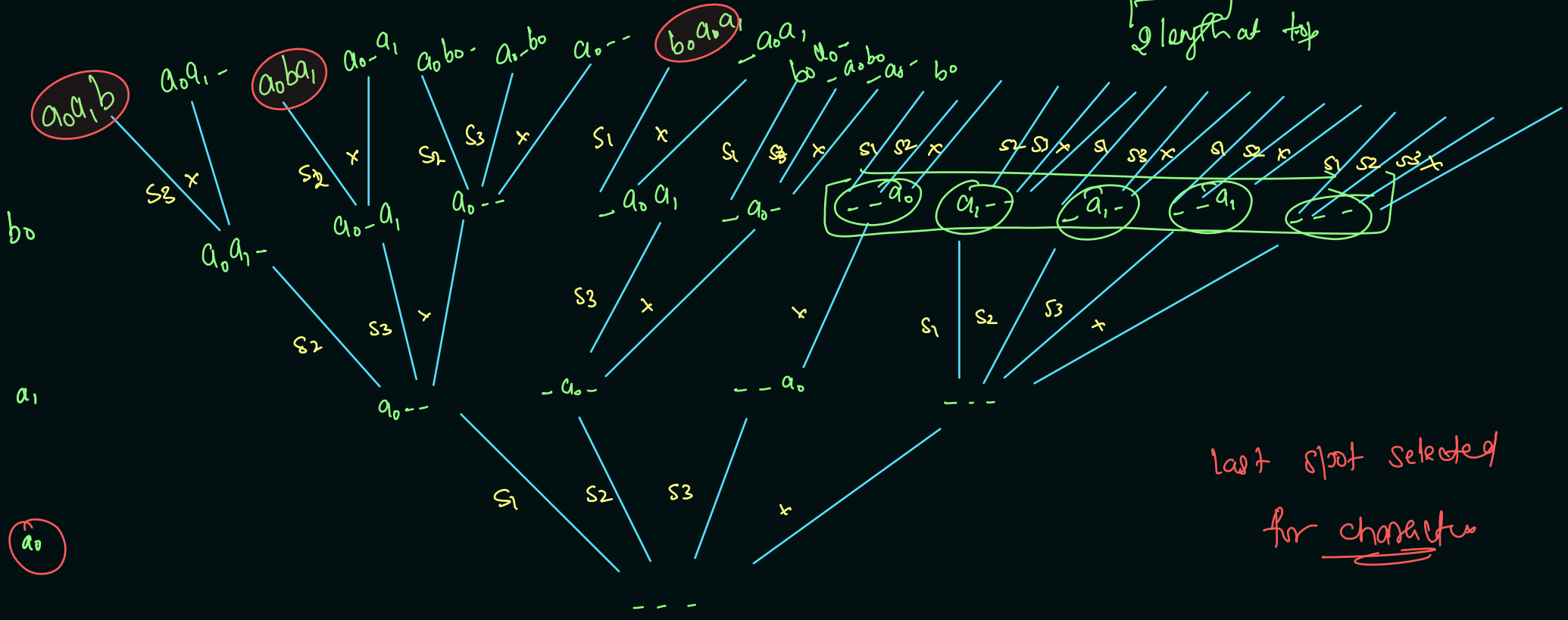
Result

$a_0 a_1 b_0$
$a_0 b_0 a_1$
$b_0 a_0 a_1$

String → $a_0 a_1 b_0$

$k = 3$ → Slot length

max length at top



Last slot selected for character

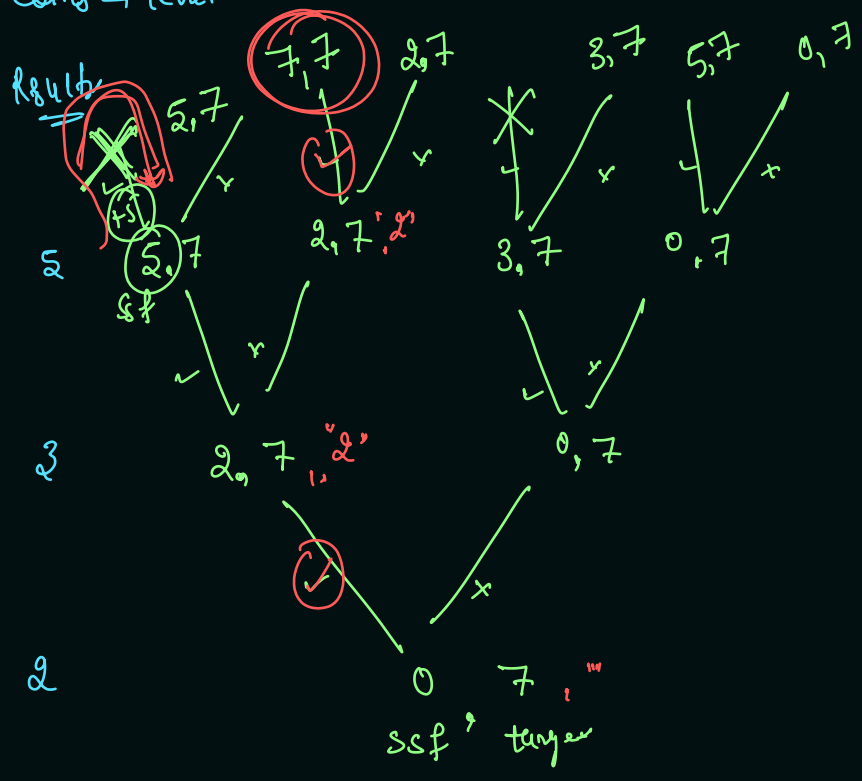
Coin change combination -1

coins → {2, 3, 5} [10, 15, ------] target = 7

(Limited no. of coins)

→ [2, 5] → combination

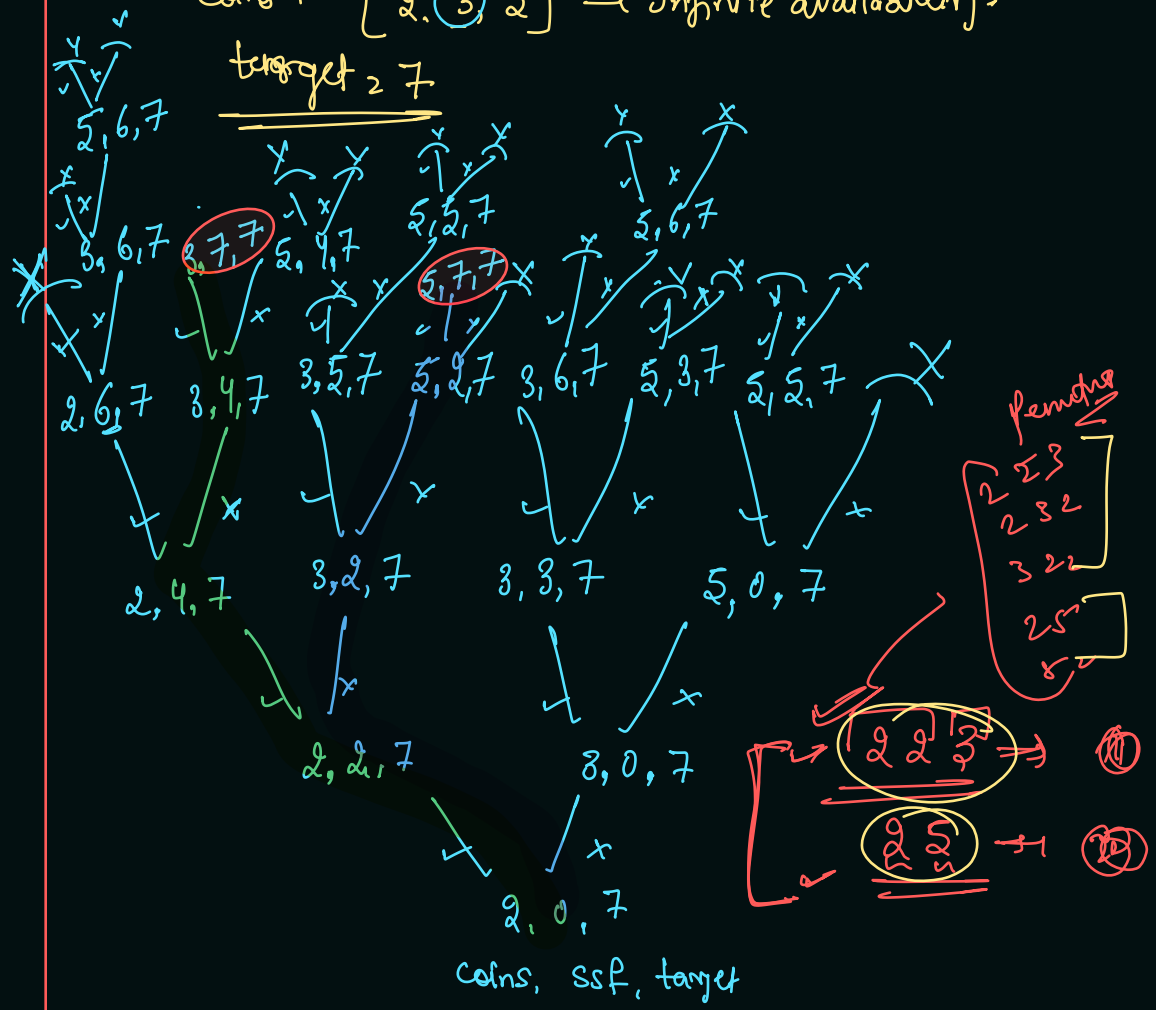
coins → level



Coin change combination - 2

coins : [2, (3), 5] → Infinite availability.

target = 7

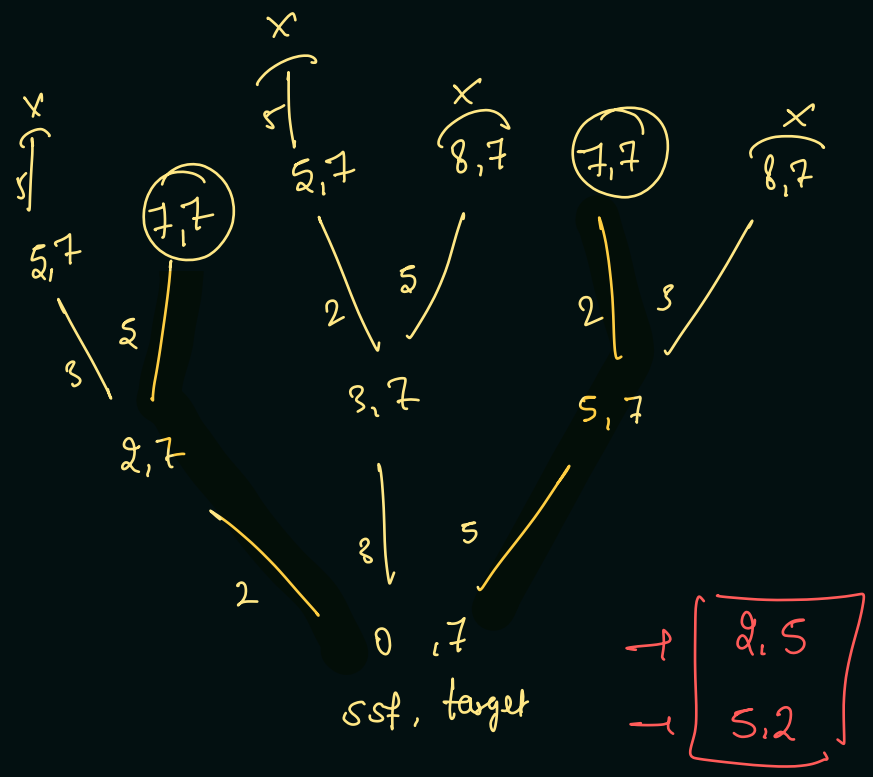


Coin change permutation - 1

Finite coins → 2, 3, 5

o/p → 2, 5
2, 2

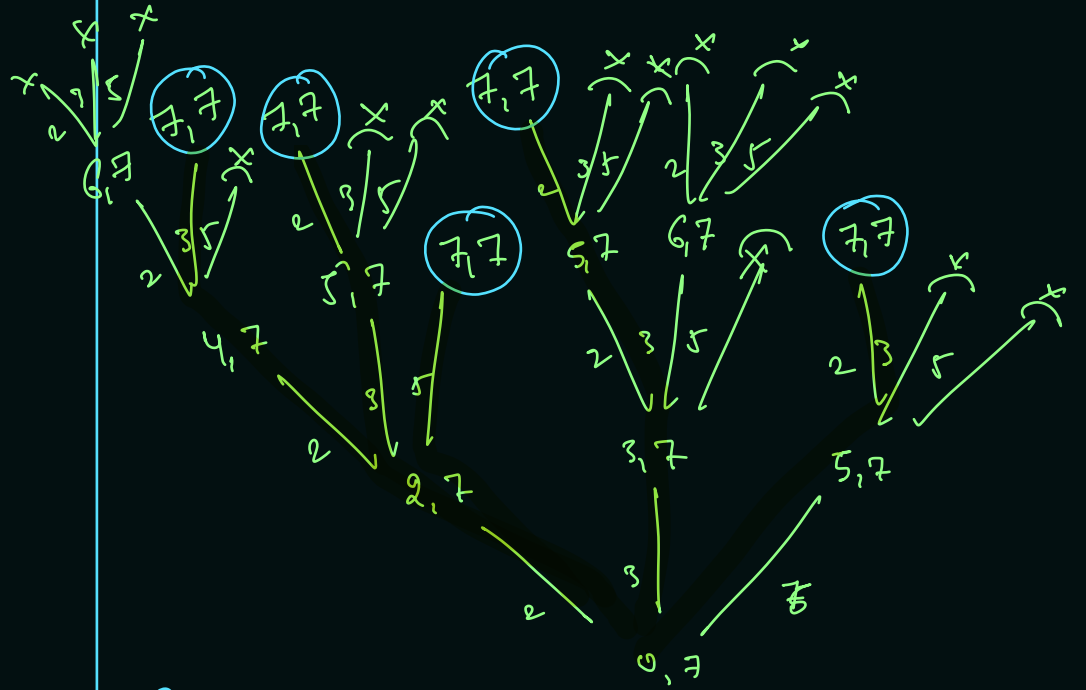
target = 7



Coin change permutation - 2

Infinite coins → 2, 3, 5 [Assumption]

target = 7



① 2-2-3
② 2-3-2
③ 2-5
④ 3-2-2
⑤ 5-2

L1 → Basics of programming
→ Recursion → 1day
→ TnS → 1 day
→ SnS → 1day.
→ Linked List → 1 day
→ Trees → 1dy
→ Graph → 1dy
→ Heap & Hash → 1dy.
→ DP → 2day.

9 days.

Revision. [Not practice]

L2 →
- R & B → 2 days
- Bits → 1 day
- DP → 5 day
- Heap n Hash map. → 2day
- LL → 1 day
- Trees → 2 dy — BST + AVL
- Graph → 2 day.
- Trie → 1 day + 1dy = 2dy
- AnS → 1 dy + 1day = 2dy
- SS → 2 dy

16 days

OOPs → 2 day.
Theoratical Part