Combinationation Sum I

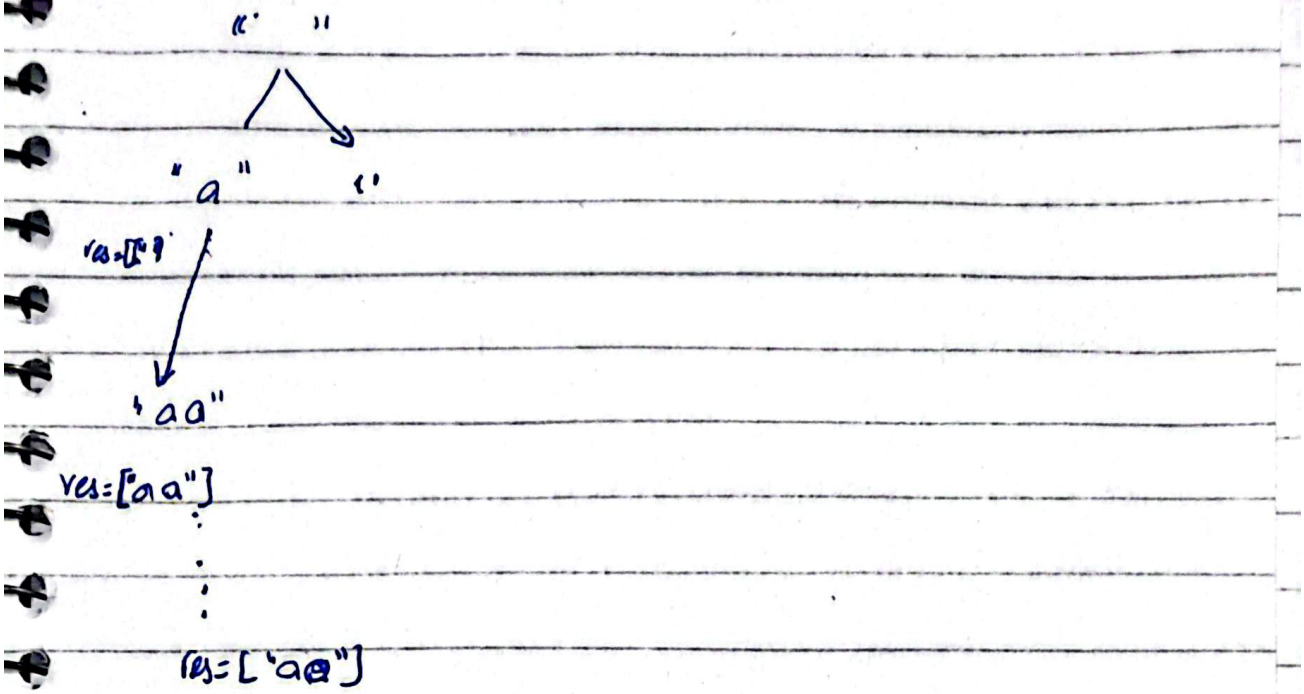
→ when ever there is a sub-sequence or combination of sub-sequence problem always jump to recursions.

→ to pick element from an array and them make same combination we always approach pick and no pick approach.

arr = [2, 3, 6, 7] , target = 7

first combination = [2, 2, 3]

-> So we have to decide if we are picking or not picking the element.

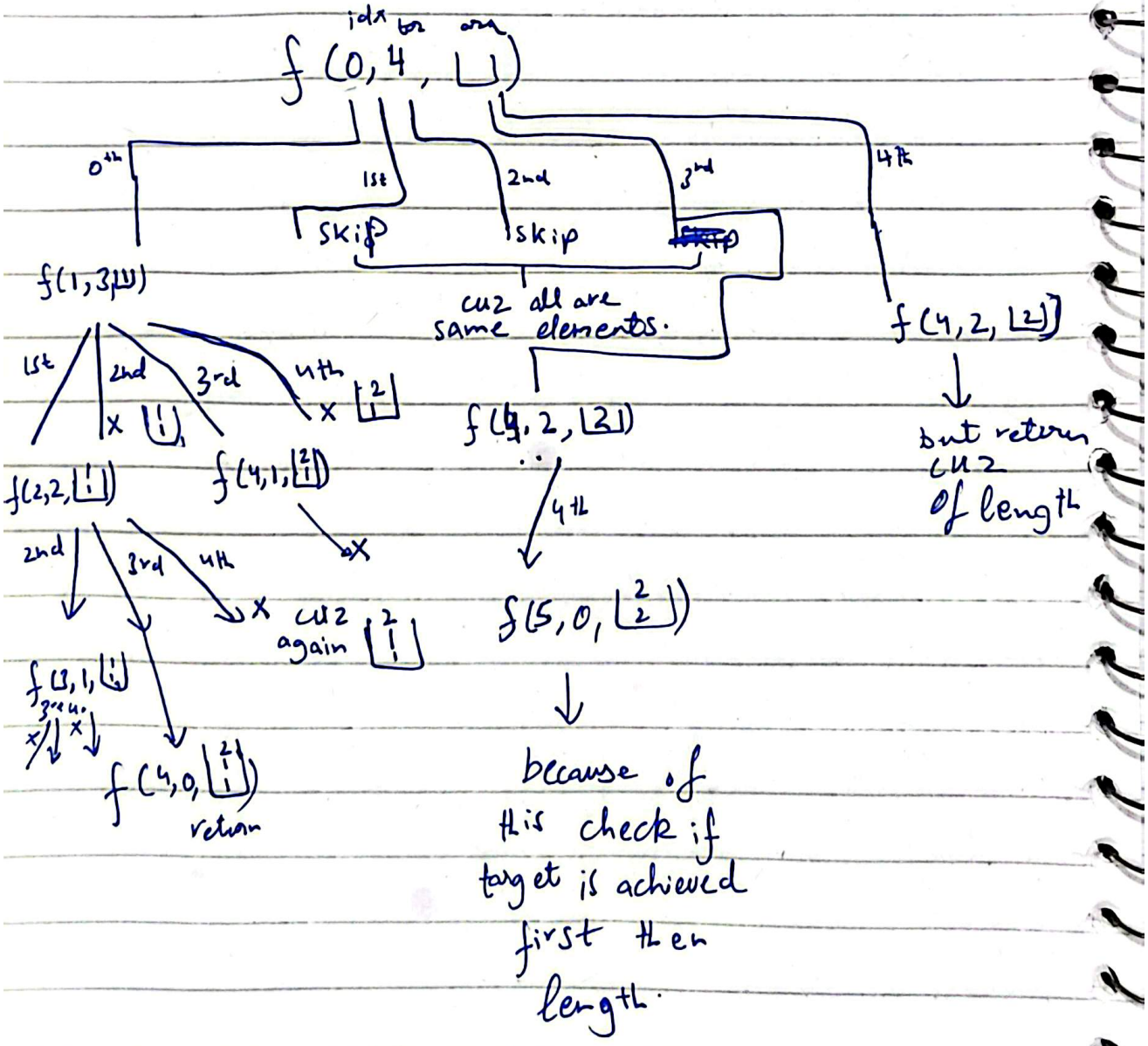

""
"a"
""
res=["a"]
"aa"
res=["aa"]
res=["aa"]

Combination al Sum II

(i) first sort the array:

arr = [1, 1, 1, 2, 2] target = 4

in this we can not repeat the index



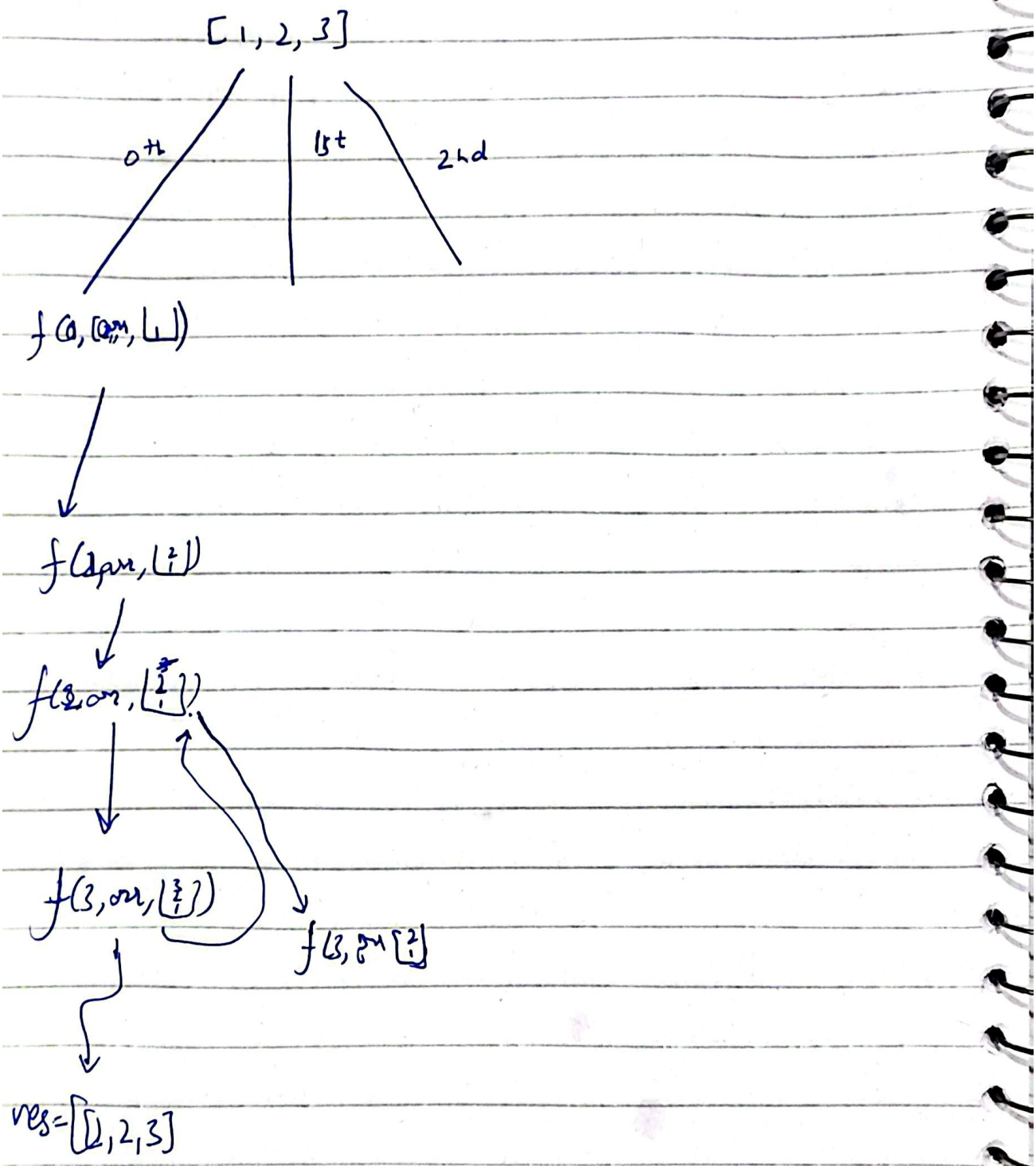
Wherever we are at 0 we try from 0 — n

when we are at 1 we try from 1 — n

→ that s' why on every recursion we are looping on every stage.
and start of that loop is increasing on every stage.
and then remove on every step what is added.

→ also if arr[i] > target is greater that target we break and return cuz arr is sorted and all the other will be greater.

→ If the last one is picked and it same like in CS 2 problem then we should just skip cuz it can make that same subset again.


[1, 2, 3]
0th
1st
2nd
res=[1,2,3]