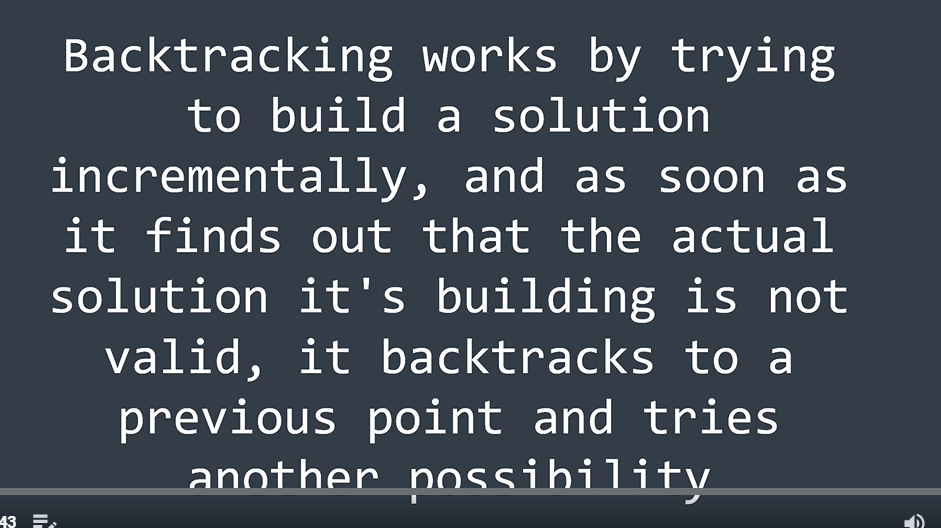
<https://www.udemy.com/course/learn-recursion/learn/lecture/23974312#overview>





Once we know, we cannot continue further, we go back to the previous point, which still has other possibilities.

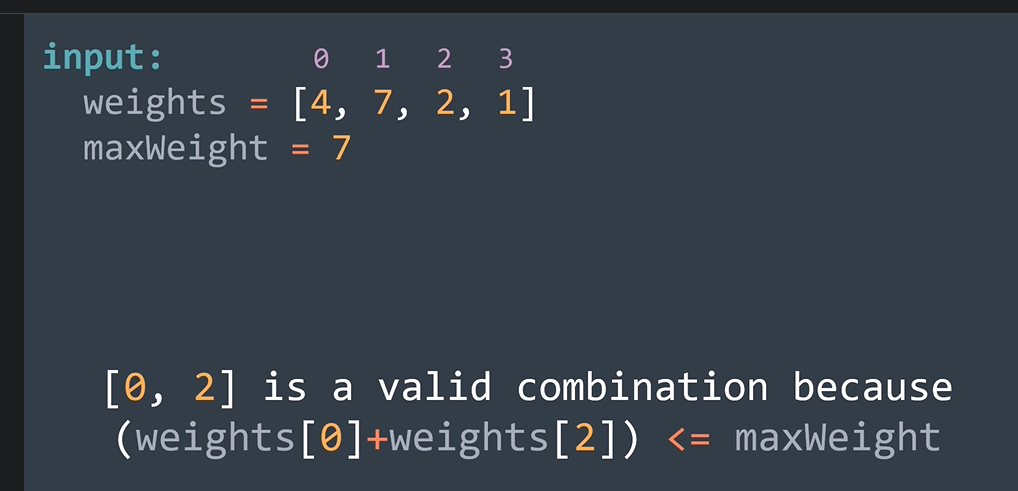
We again go ahead, and if blocked again,, we again backtrack to the point where we can explore other possibilities..

We keep on doing this until we reach the end..

Looking at the problem:

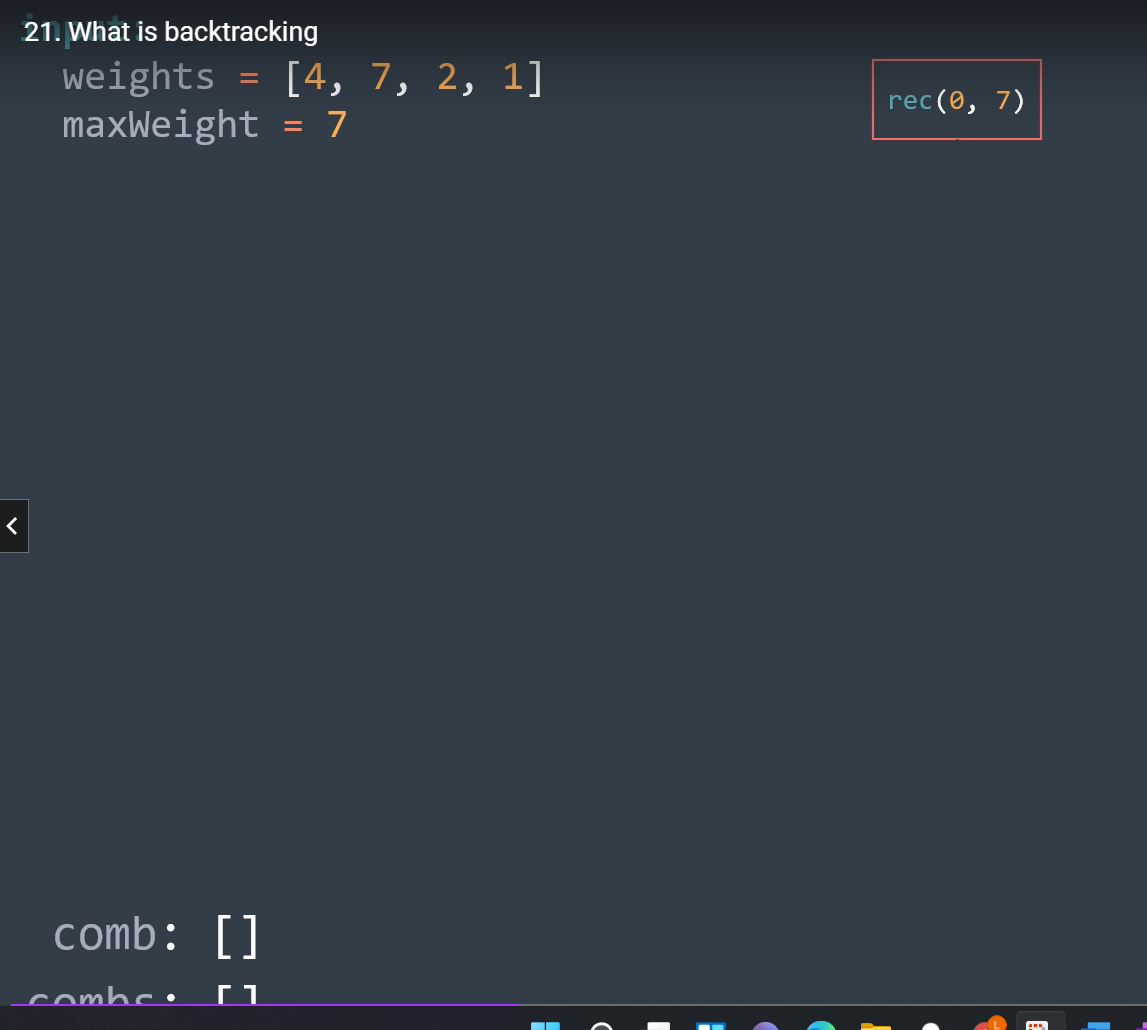
Knapsack, which can carry some maximum weight.. and we have weighrs.. total sum should not exceed max. weight.





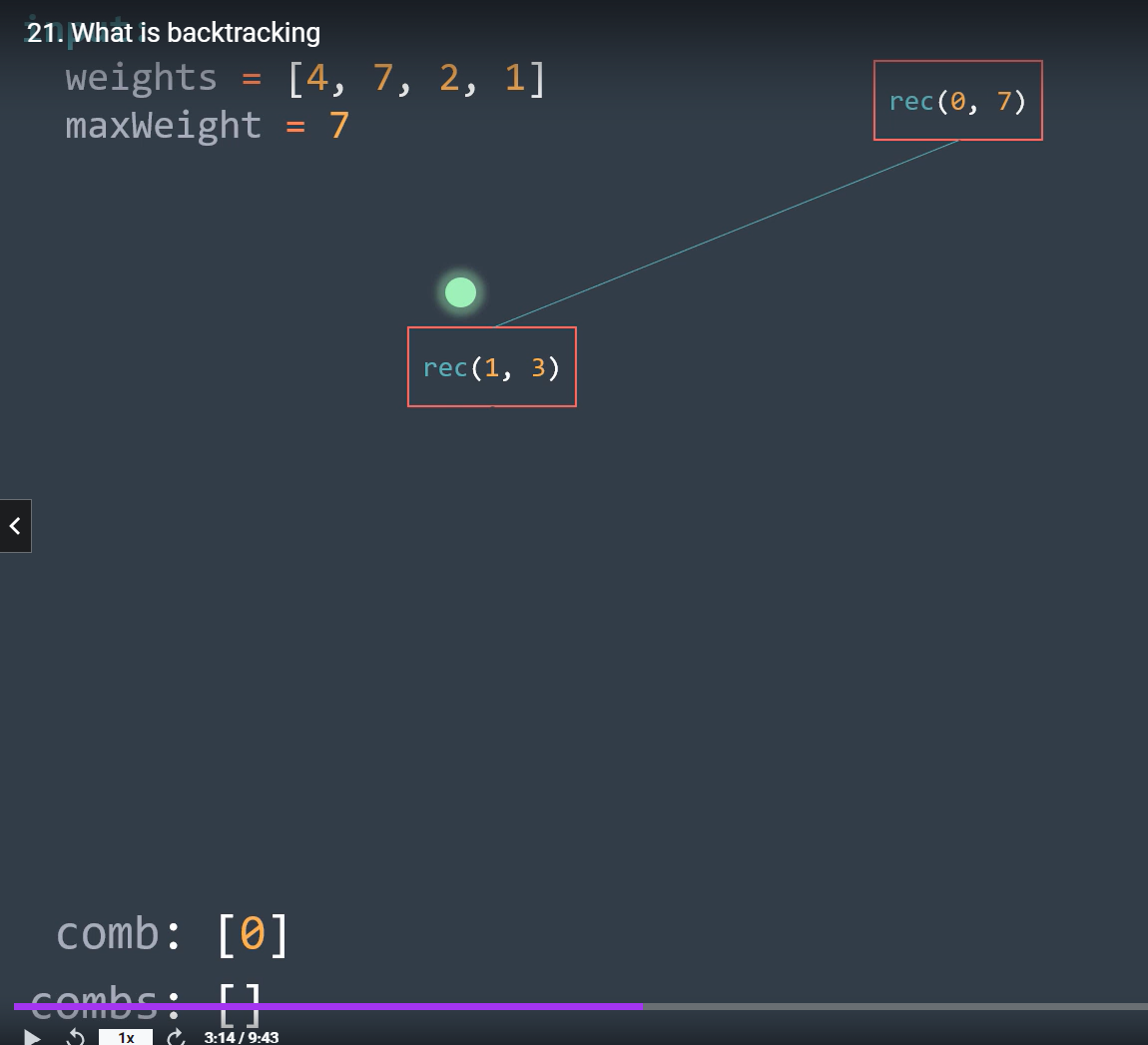
We want all the valid combinations..

Use the backtracking tp solve this..

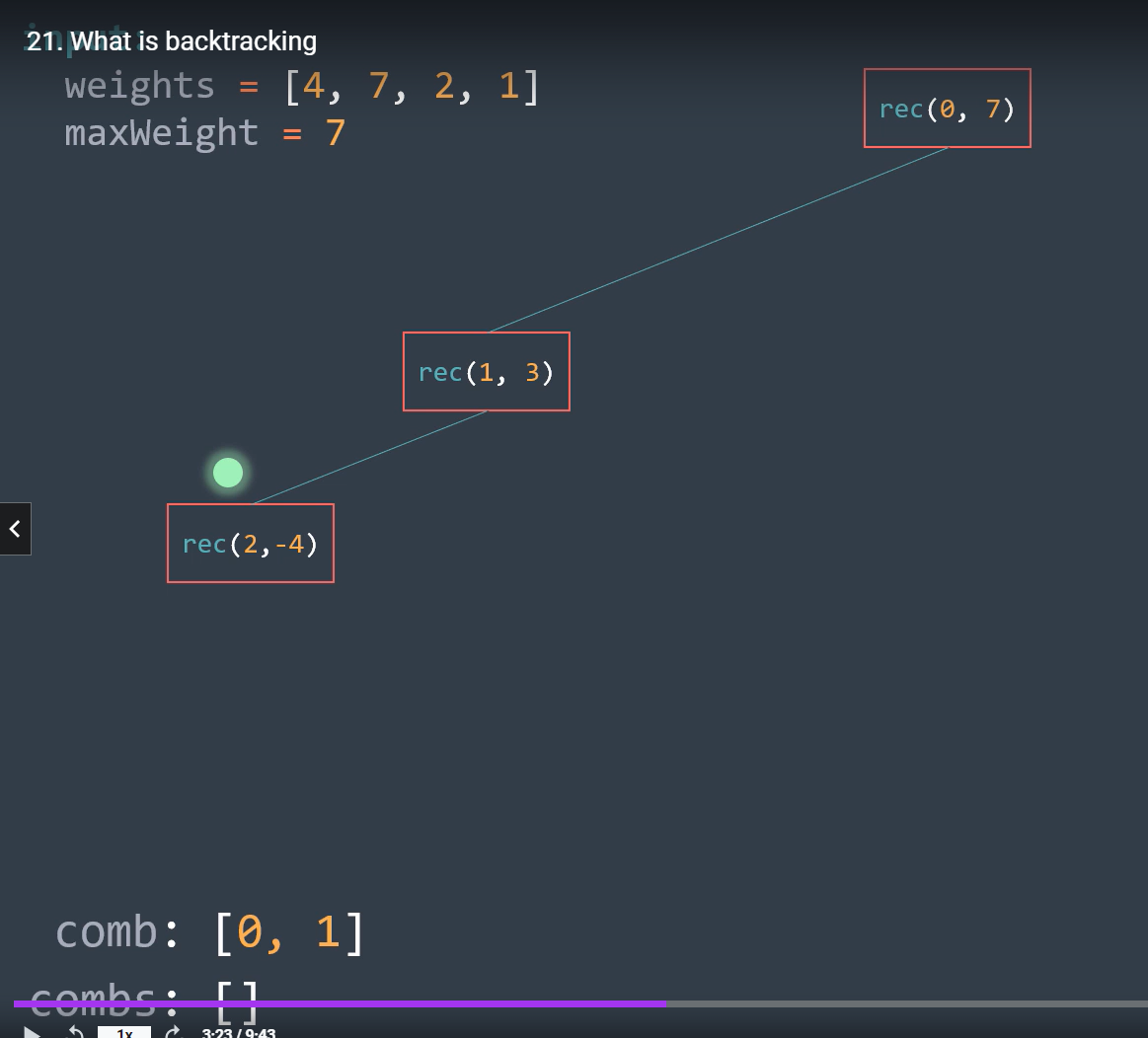


Going to left means we have taken the element, and going to right means we have not taken the element.

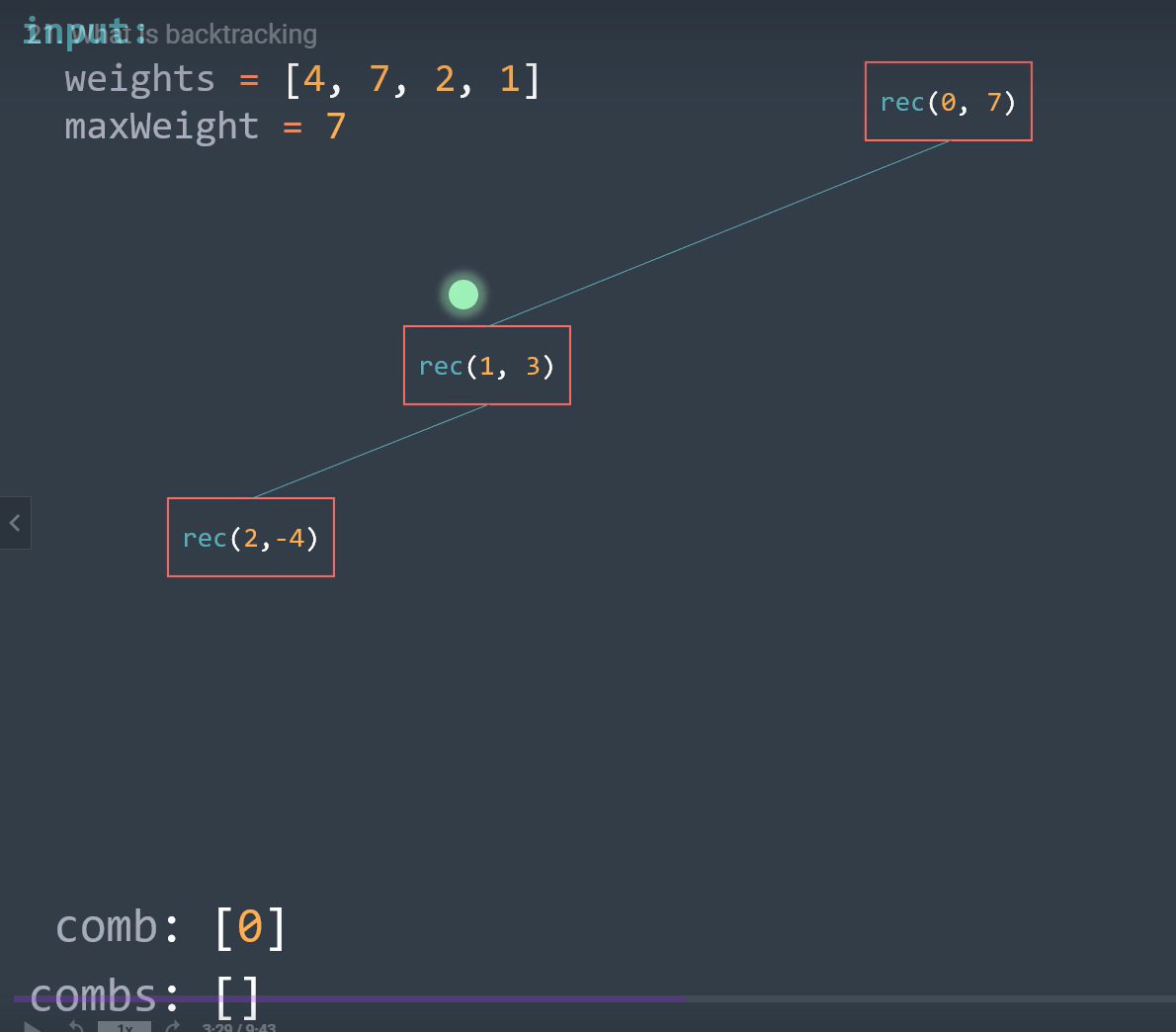
Comb array contains the current combination, and combs will contain the valid solution.



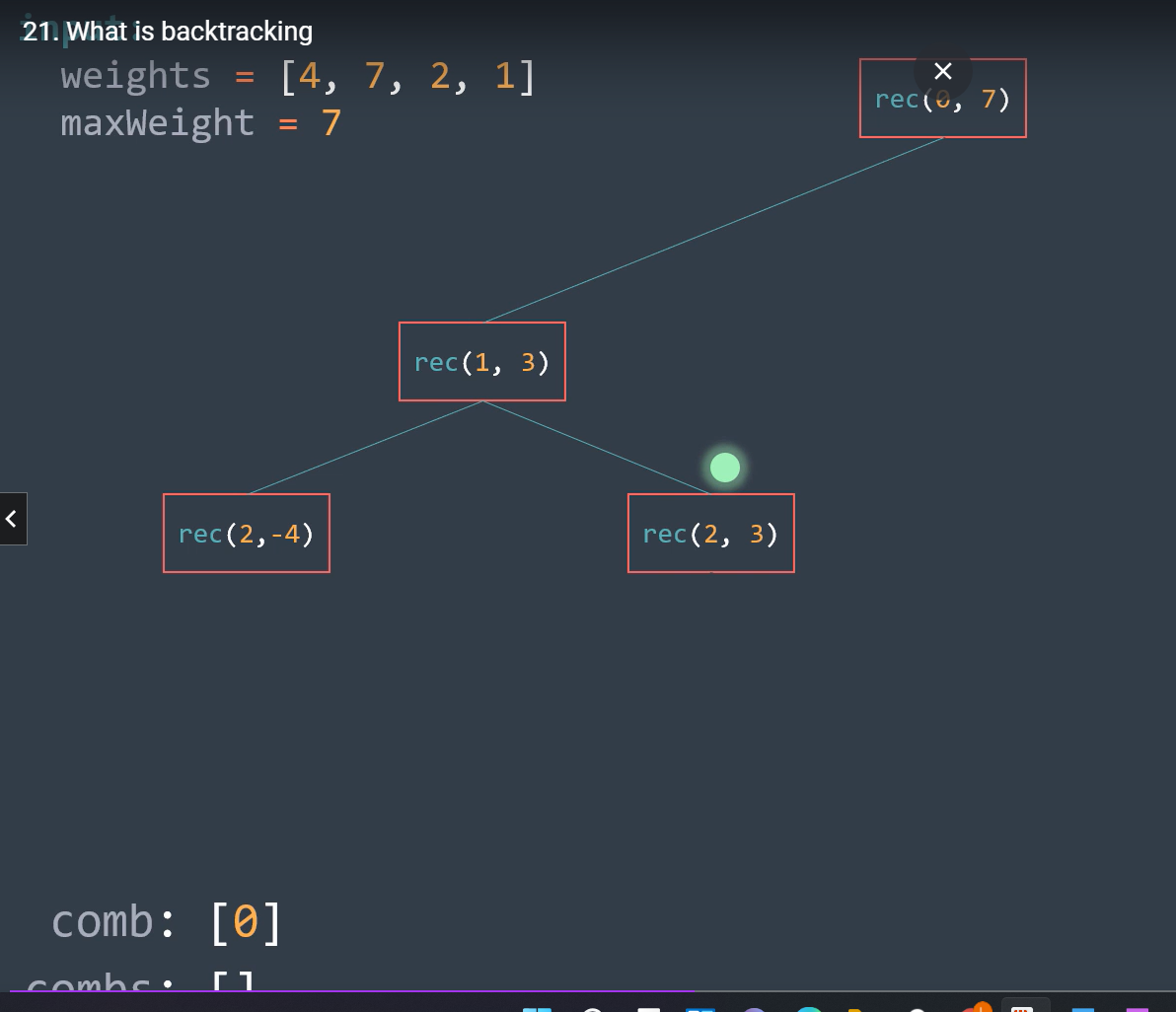
1st aparameter represents the index, 2nd represents the remaining weight..



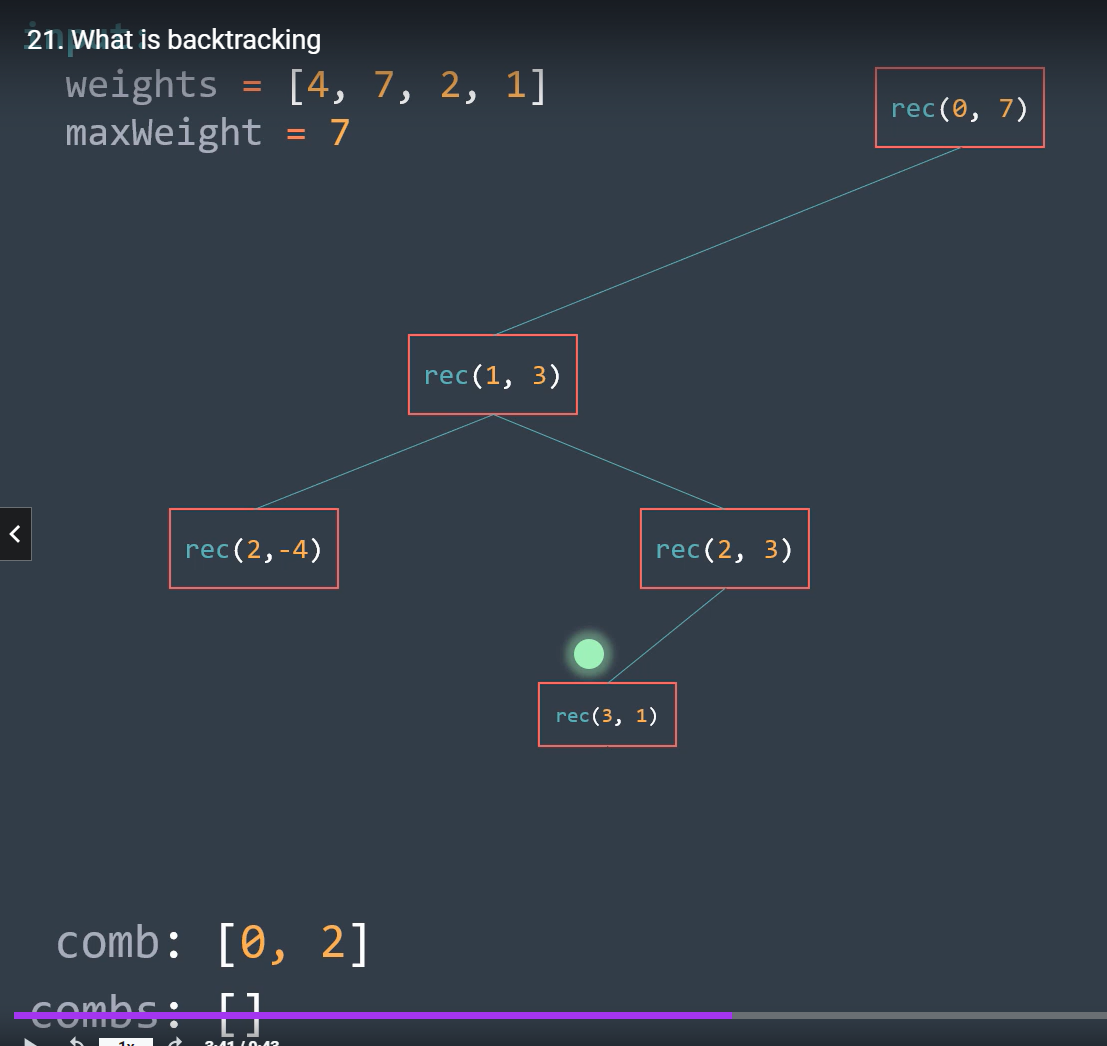
After subtracting max weight will become negative, this combination is not valid, we do not want to return anything.. **so we will backtrack.. and we will just remove or pop out this** current element from the comb array. (the element 7 at index 1 will have to be removed).



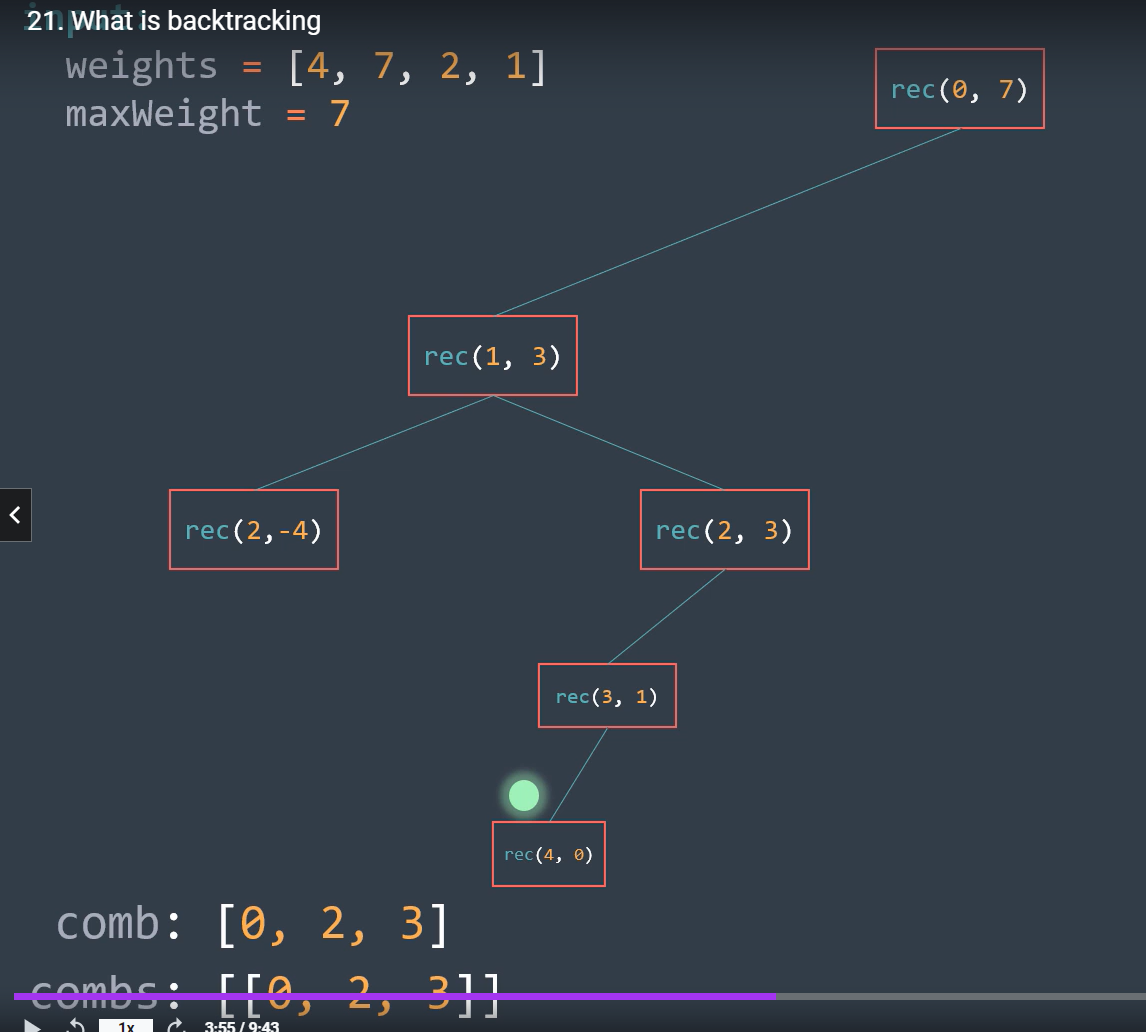
We did backtracking above, and removed or popped out the weight at index 1.



Now what we have done above is we have not taken the element at index 1 and have moved to index 2.

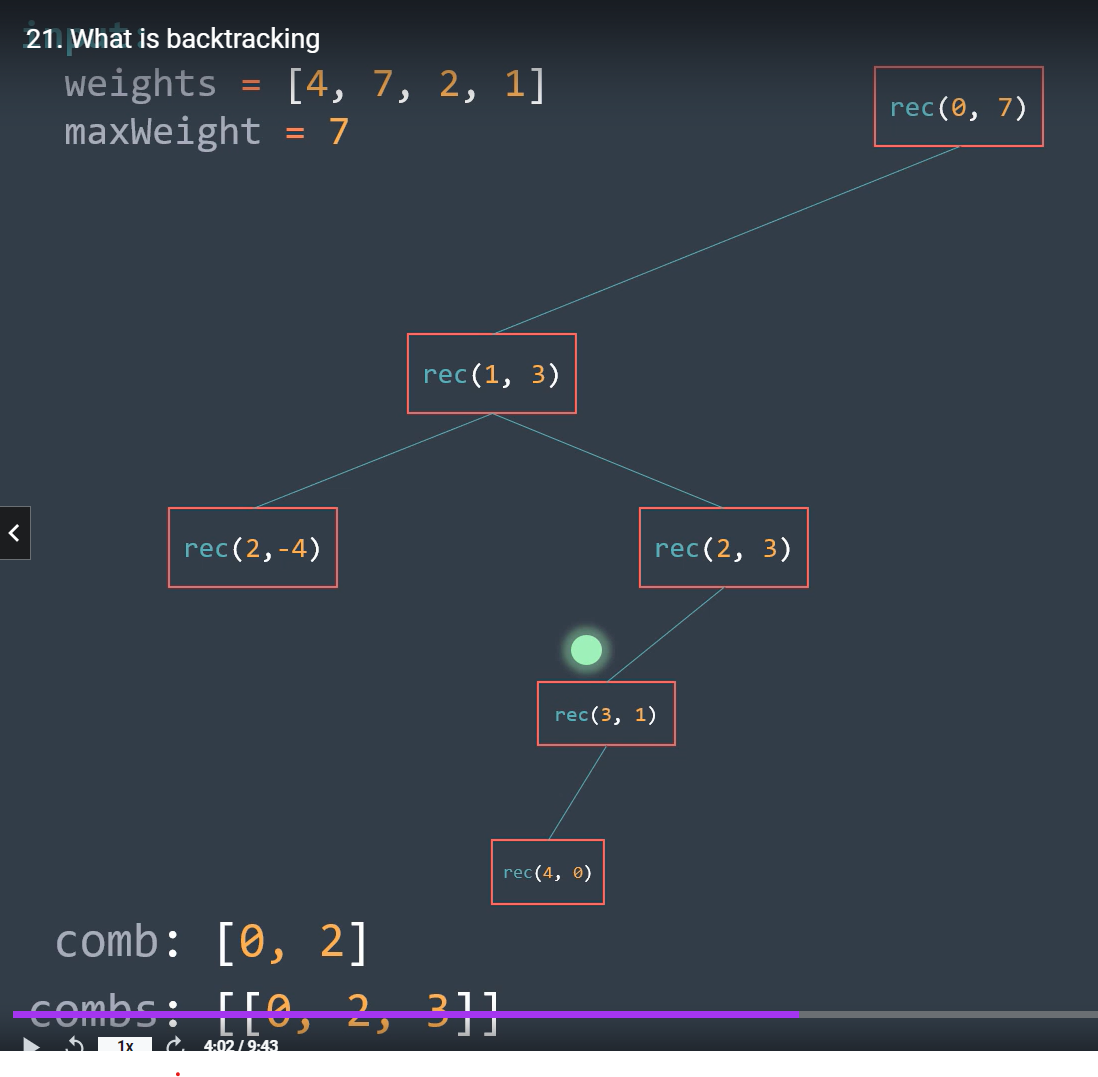


Now after we moved to index 2, let us take item two.. as done above..

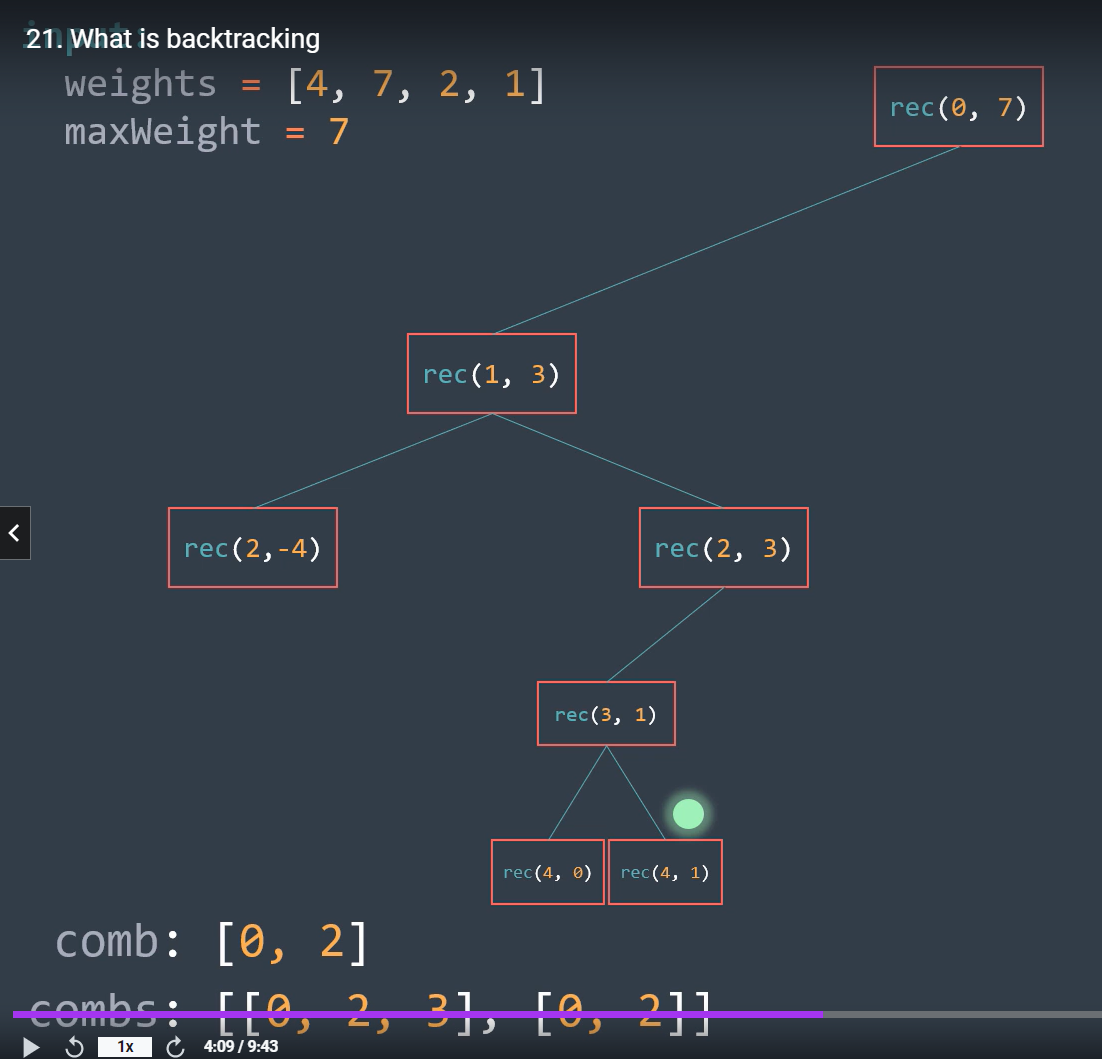


We took next item… now we have traversed all the elements.. what do we do? As w have traversed all the elements, we can put this combination in the combs array and we can backtrack from here.. to the last position.. (or last index which had value 2)..

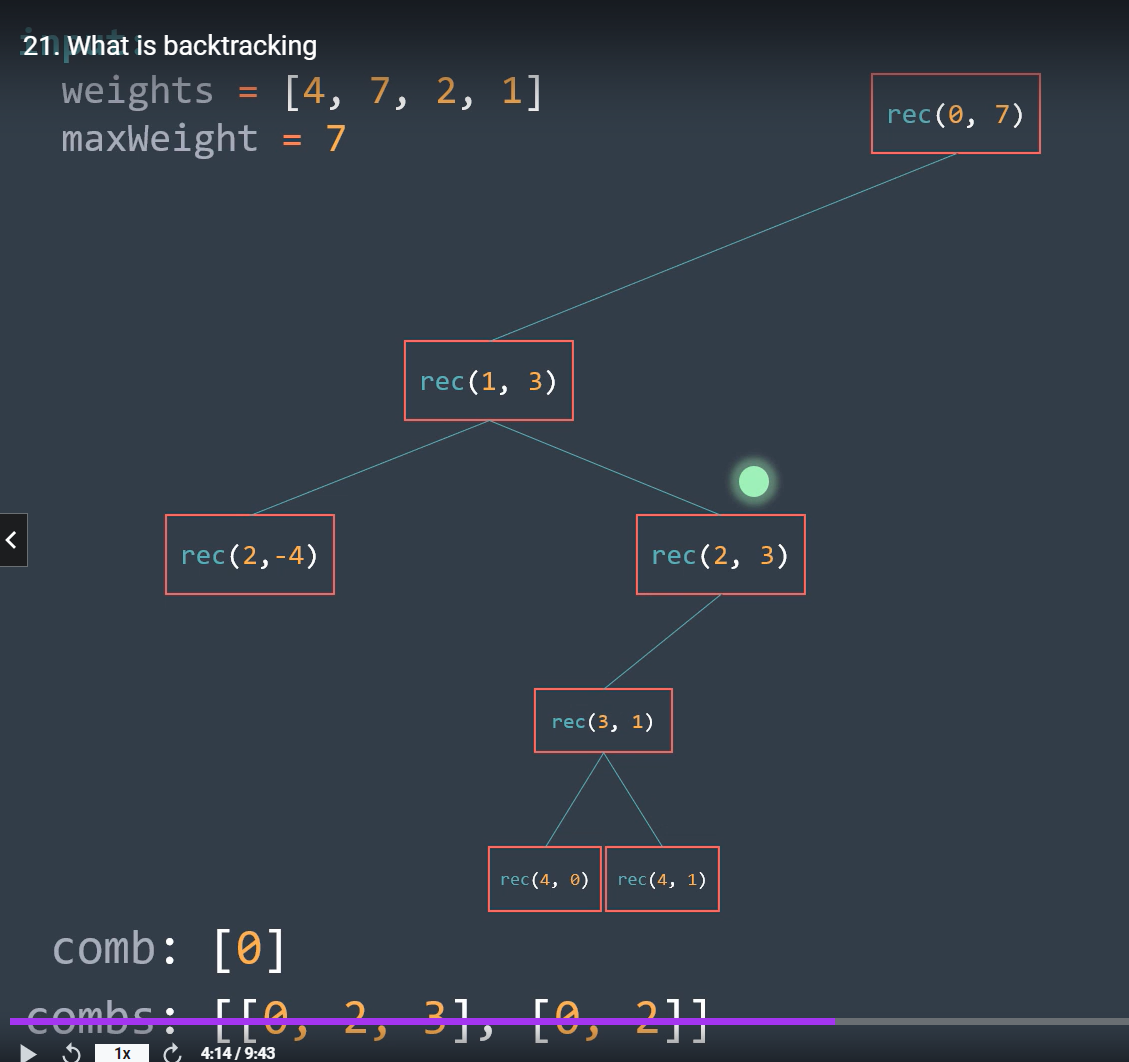
**How do I backtrack from (4,0) in terms of code? Just pop it off.. pop off element at indx 4/**



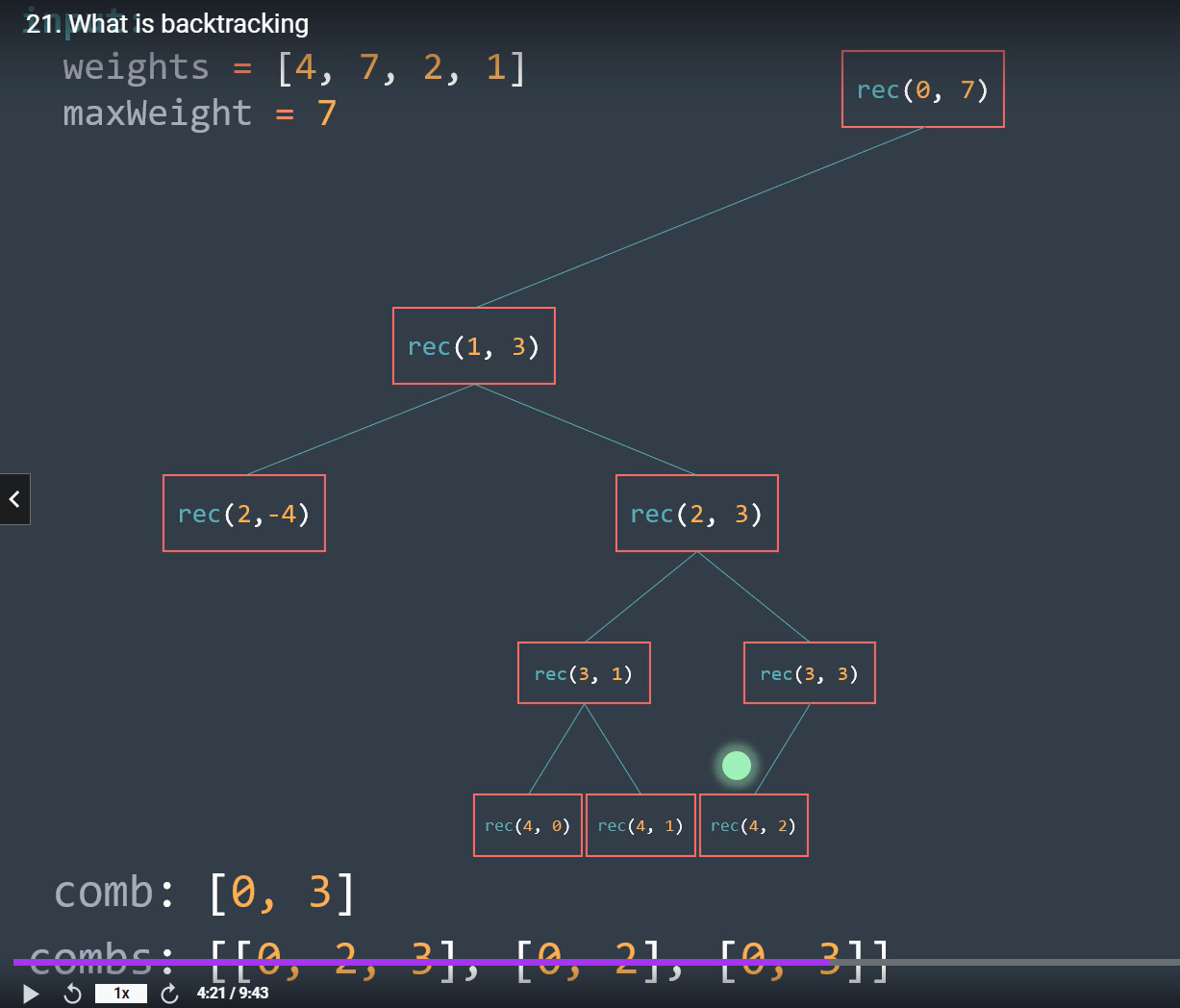
As we want to ffind all of the solution, we backtrack to the index 2.. (btw 3 means we have just considered index 2 as weight remains 1, so we say we are at index 2)..



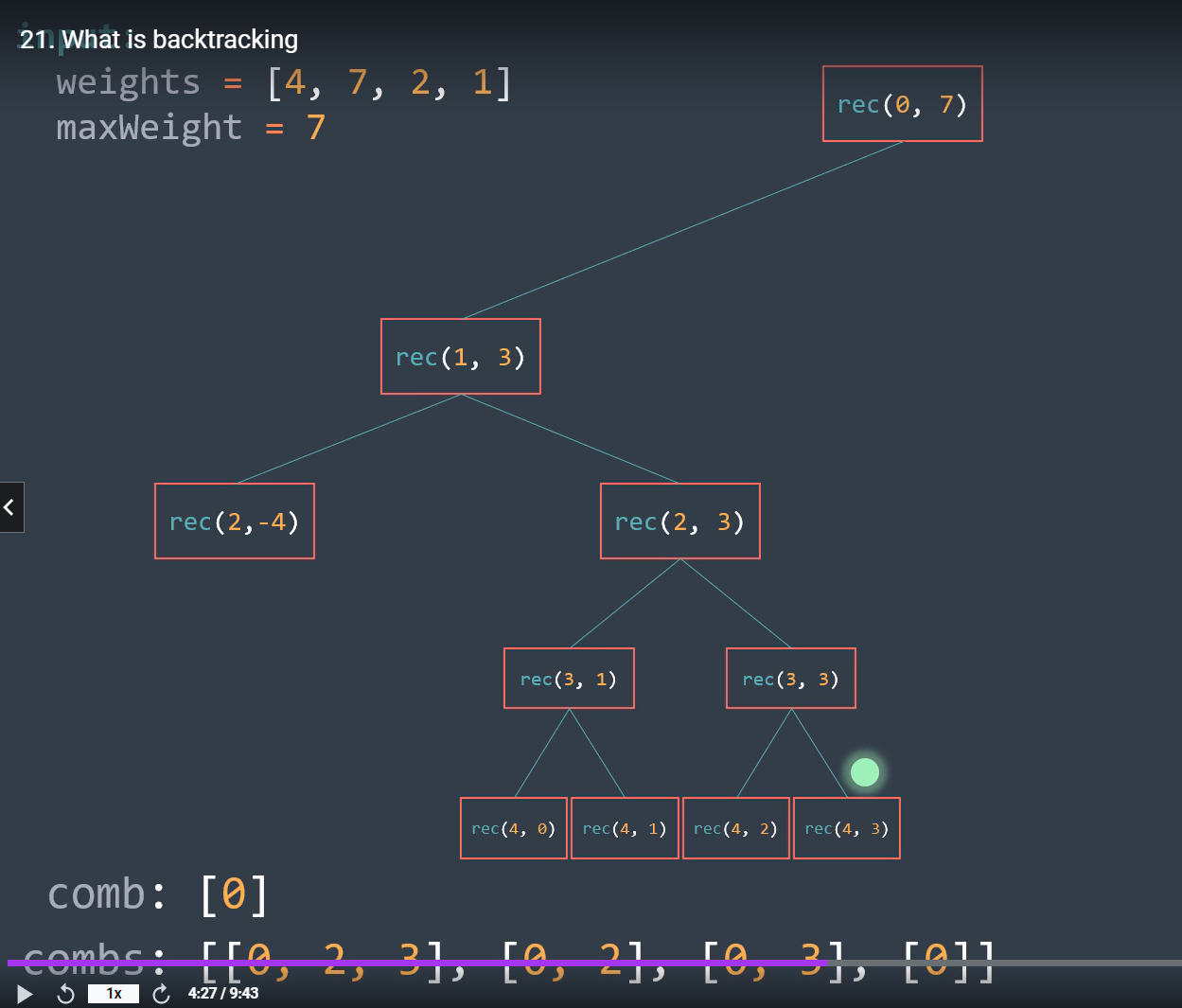
We choose another call by not consireing the next element.. that is just move by index but do not subtract.. from the weight..



Now we backtrack twice as we have tried both index 2 and 3, so we backtrack (means pop the elements at index 2 and 3 from comb).



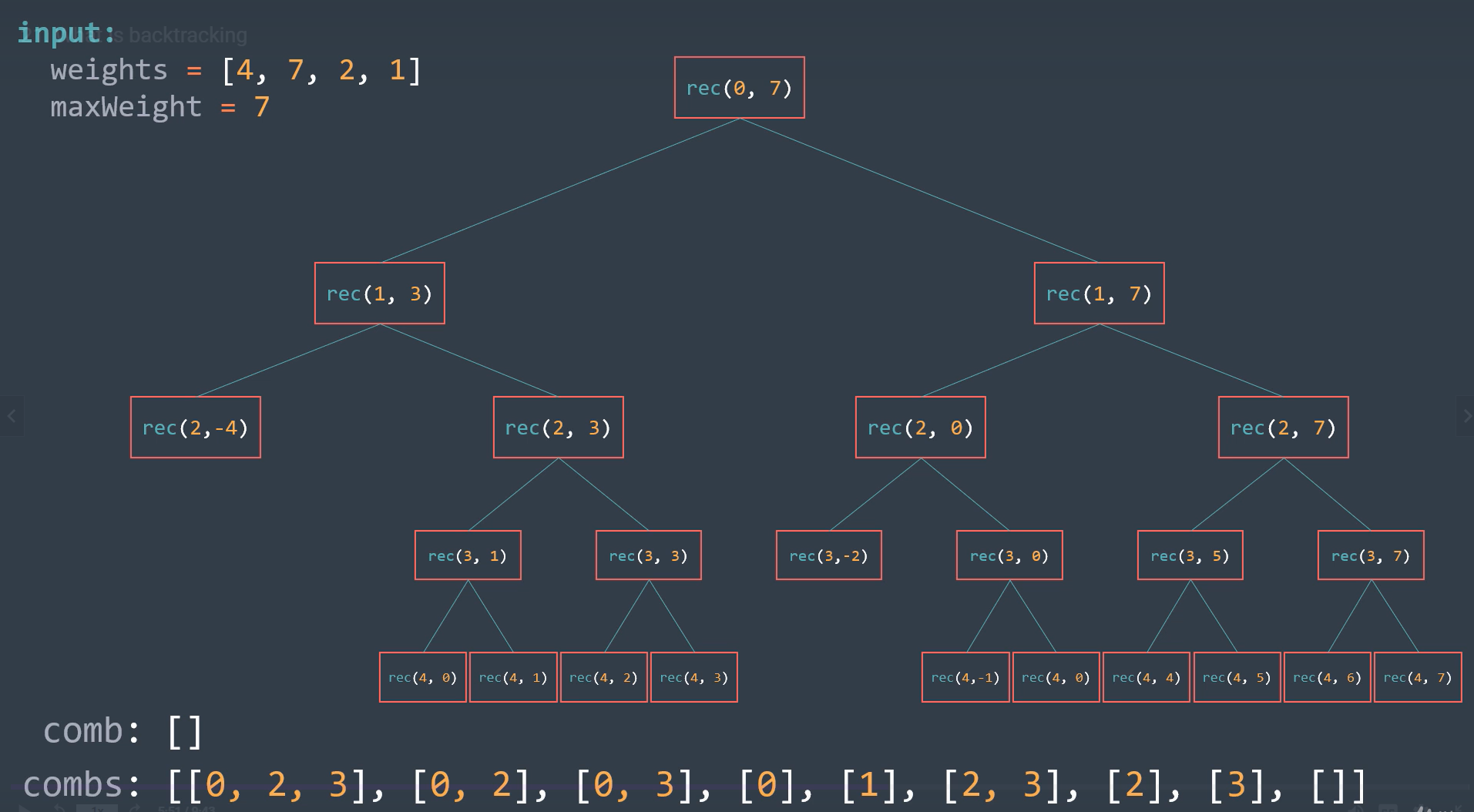
We try by not taking item at index 1 or index 2. And we take the item at index 3 and take that combination, as it is valid…



Max weight remains positive and we take the combination..

Interesting about this is once you figure out 1st triangle.. i.e complete left hand side and right hand side upto (rec(2,3))… all others are like just a copy of these..

#===============🡺



Similarly RHS can be completed of this tree..

At the end when we backtrack to the root.. comb will be empty and combs will have all the valid combinations..

#--===🡺

