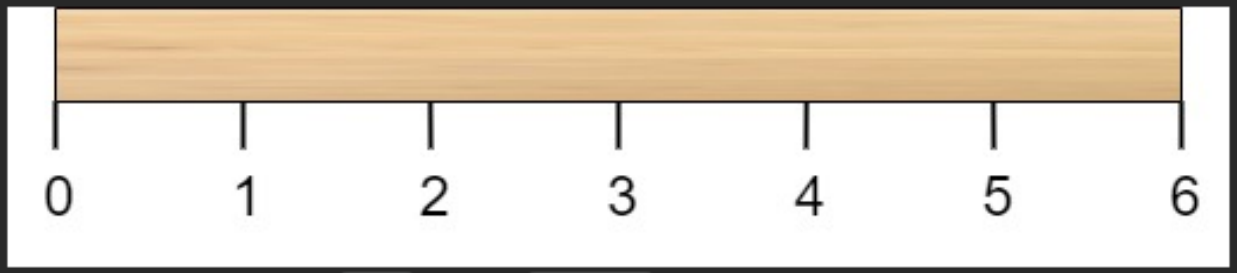
Minimum Cost to Cut the Stick

Problem Statement:

Given a wooden stick of length `n` units. The stick is labelled from `0` to `n`. For example, a stick of length **6** is labelled as follows:



Given an integer array `cuts` where `cuts[i]` denotes a position you should perform a cut at.

You should perform the cuts in order, you can change the order of the cuts as you wish.

The cost of one cut is the length of the stick to be cut, the total cost is the sum of costs of all cuts. When you cut a stick, it will be split into two smaller sticks (i.e. the sum of their lengths is the length of the stick before the cut). Please refer to the first example for a better explanation.

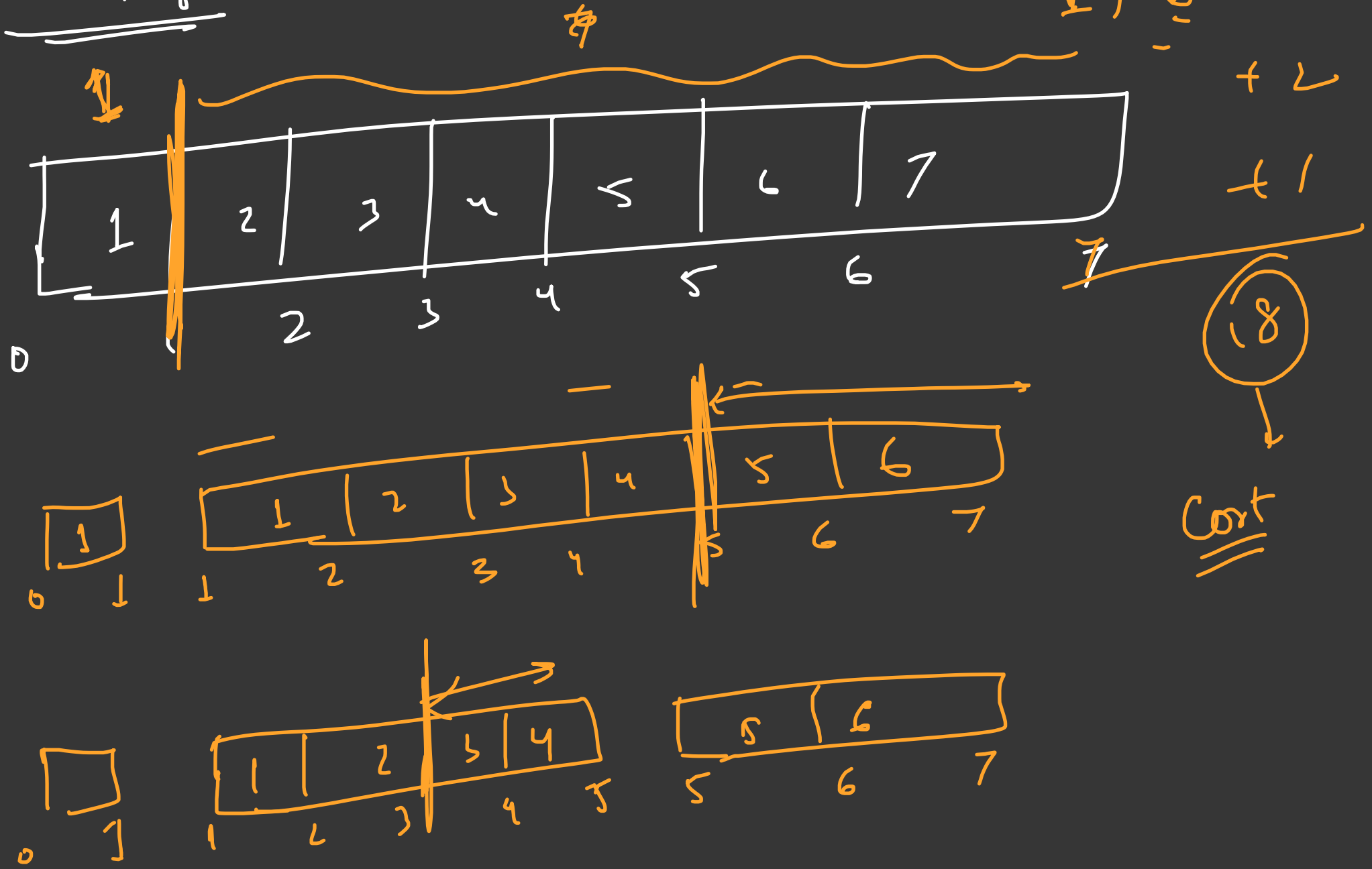
Return *the minimum total cost* of the cuts.

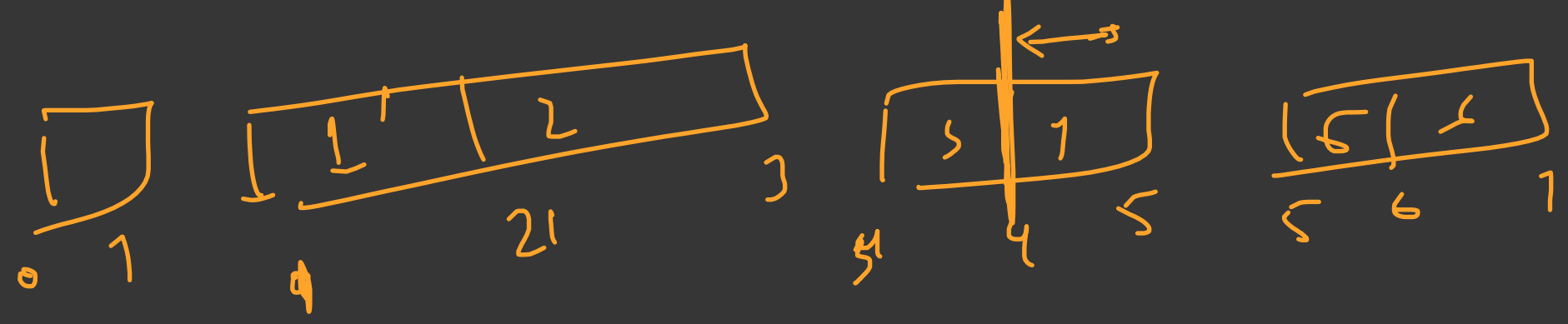
n = 7

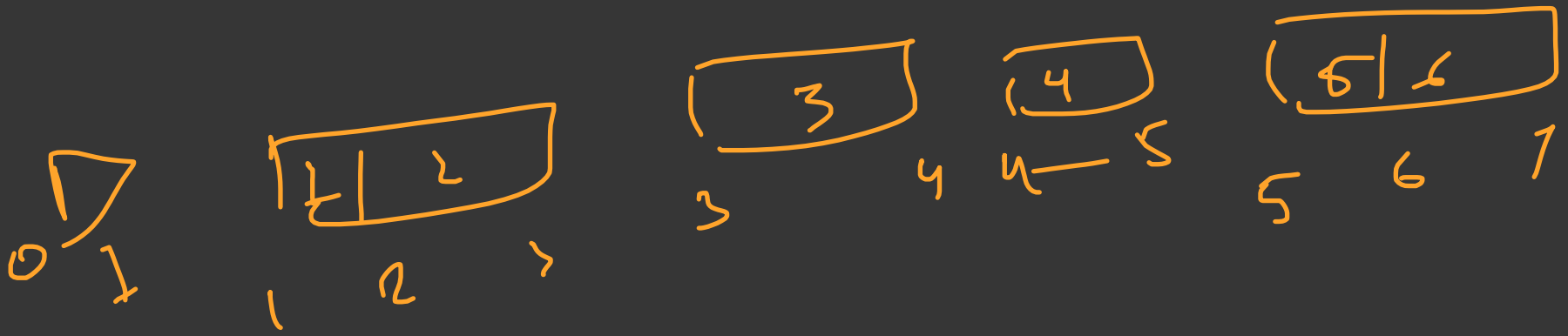
cuts = { 1, 5, 3, 4 }

index → any order

Stock Span 2

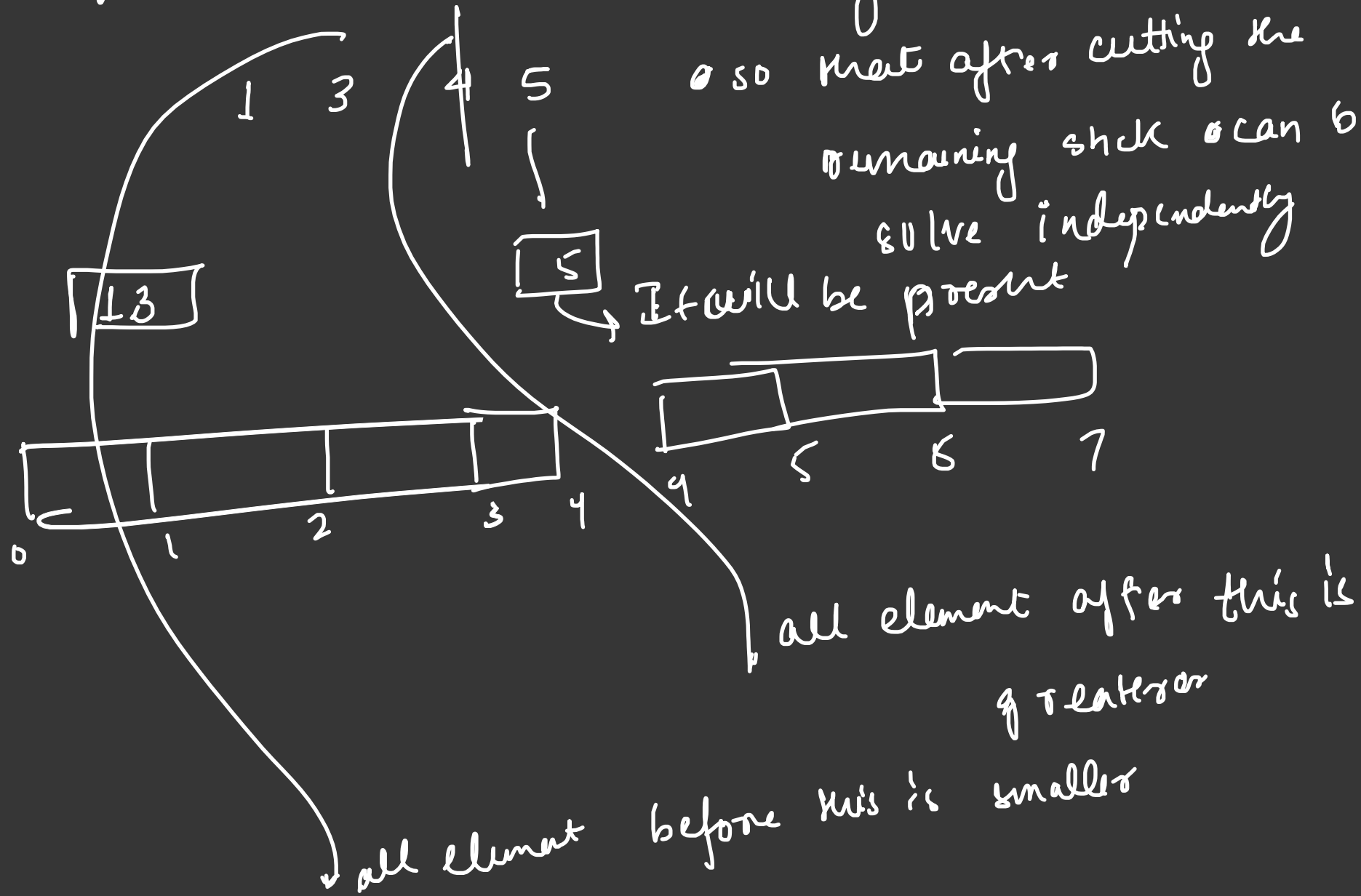






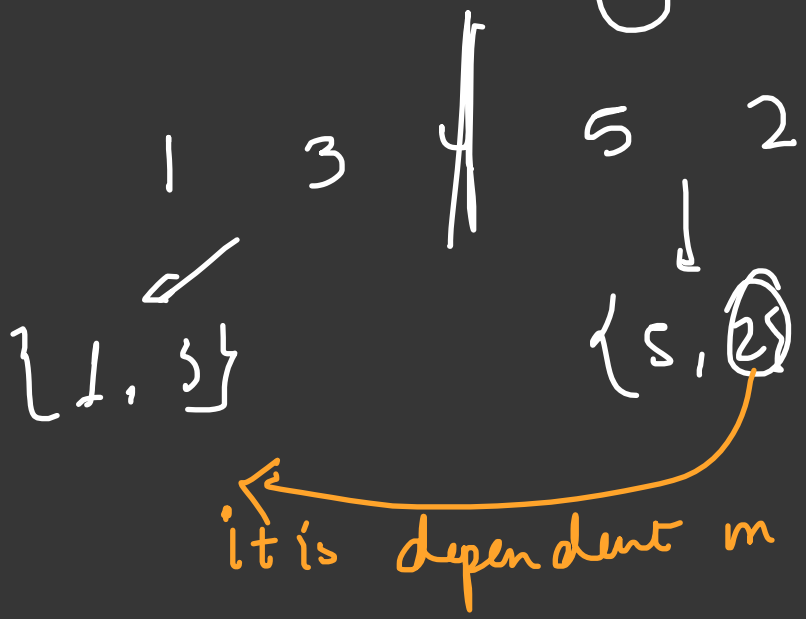
minimum Cost find

- First we have to sort the cut array.



- so that after cutting the remaining stick can be solve independently

If we not sort the array



it is dependent on index 2 which is on other side

Cost of one cut : Length of the stick in which element is present.

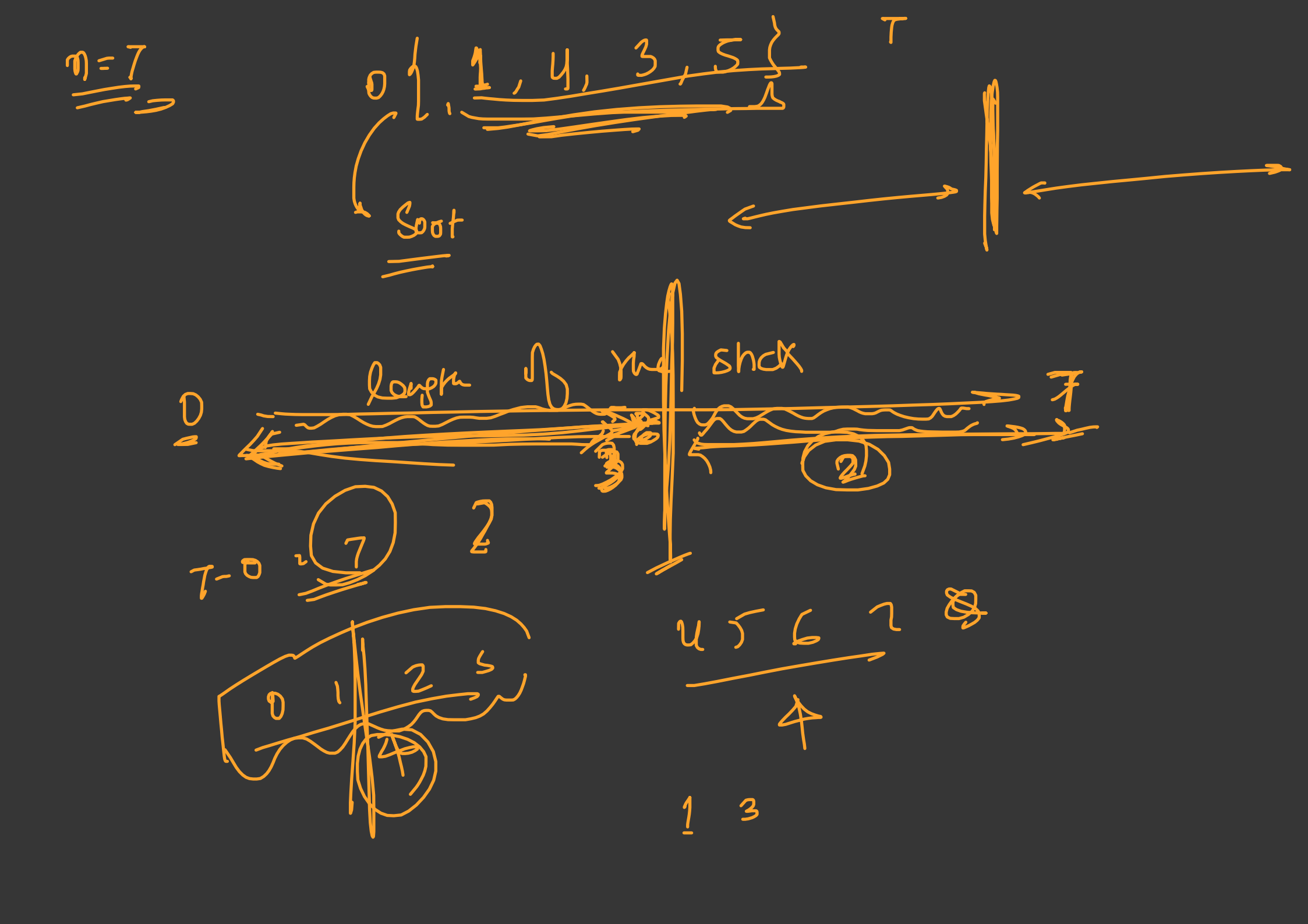
So, 0 { 1, 3, 4, 5 } 7

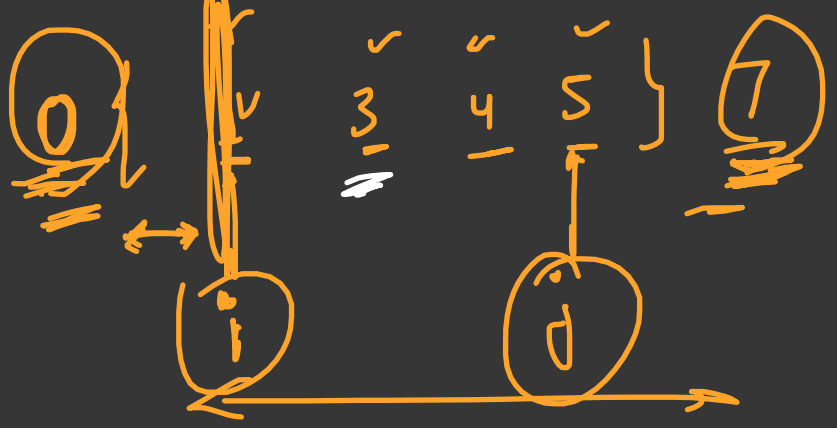
cost = arr[j+1] − arr[i−1], + f(i, k−1)

+ f(k+1, j)

Partition

insert 0 { 1, 3, 4, 5 } 7

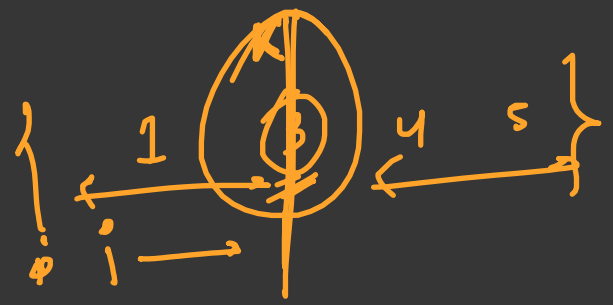

n = 7
0 { 1, 4, 3, 5 }
T
Soot
0
length
shock
7
2
T - 0 = 7
2
0 1 2 5
4
4 5 6 7 8
4
1 3

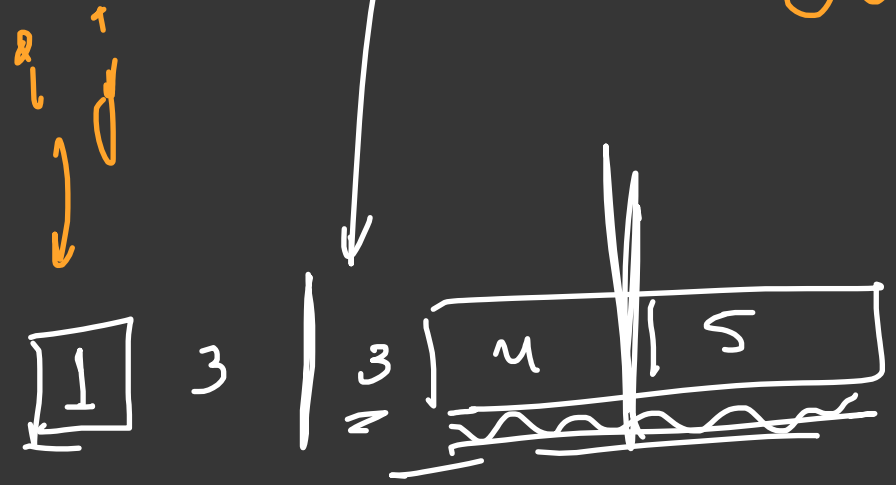


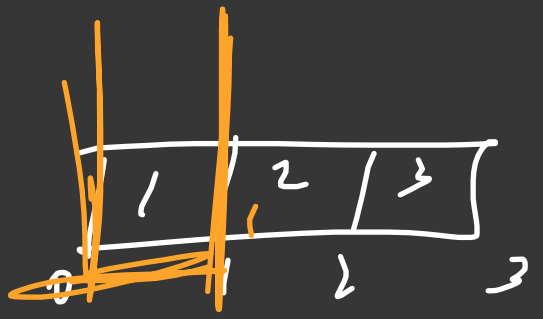
for (k = i ⟶ j)

Cost = arr[j+1] − arr[i−1] + f(i, k−1)

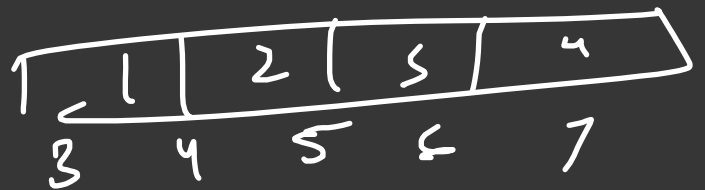
+ f(k+1, j)







7 − 3 = 4



Recursive function:

Sort array

0 ... i ... j

Plan

$f(i, j)$

Base case: if ($i \geq j$) return 0;

① Start control block and start with i and j

min = len,

$f_{or}(i \longrightarrow j)$

② Try out partition

Cost = arr[j+1] − arr[i−1] + $f(i, k-1)$ + $f(k+1, j)$

min = min(min, Cost)

return min