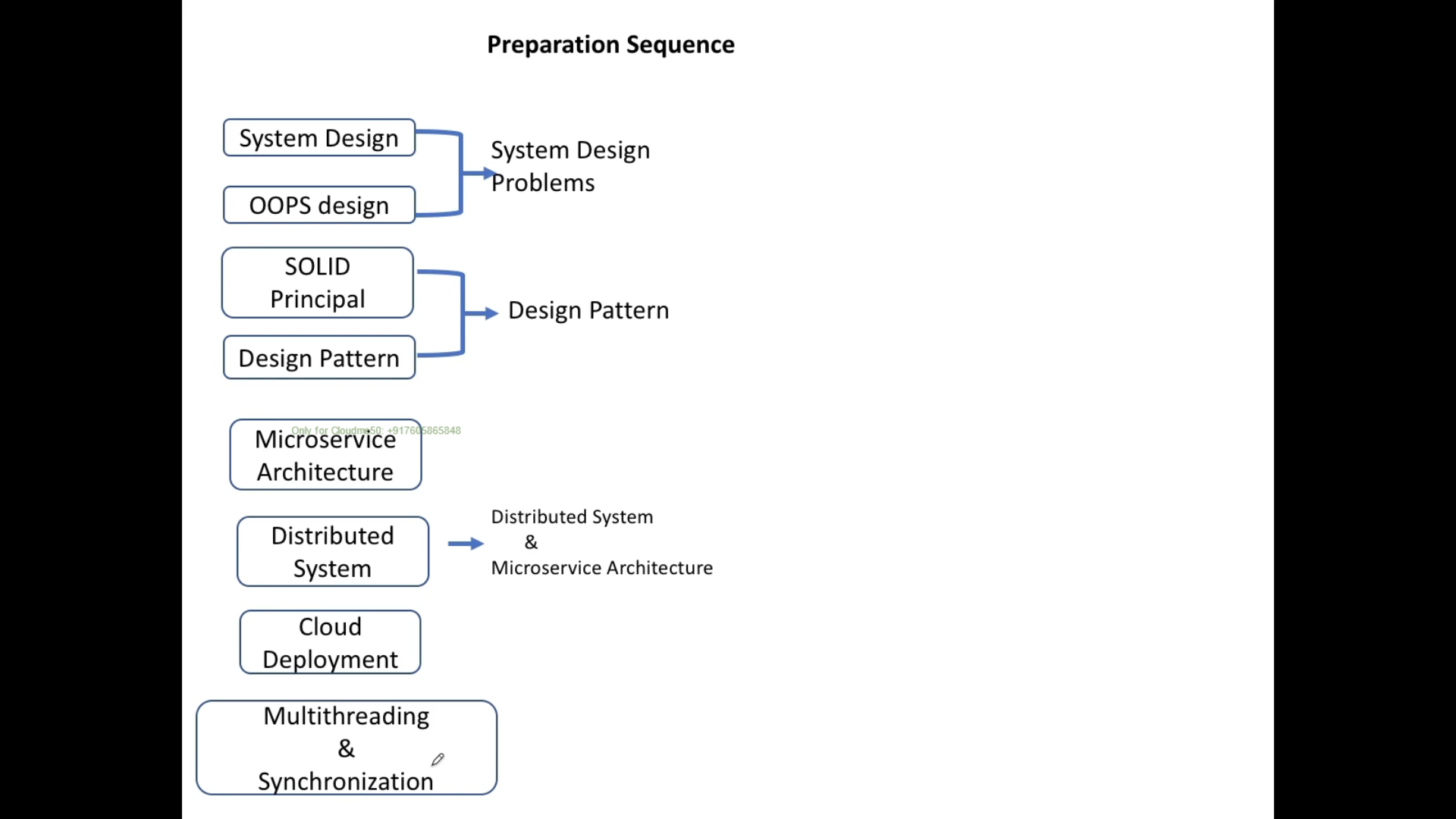
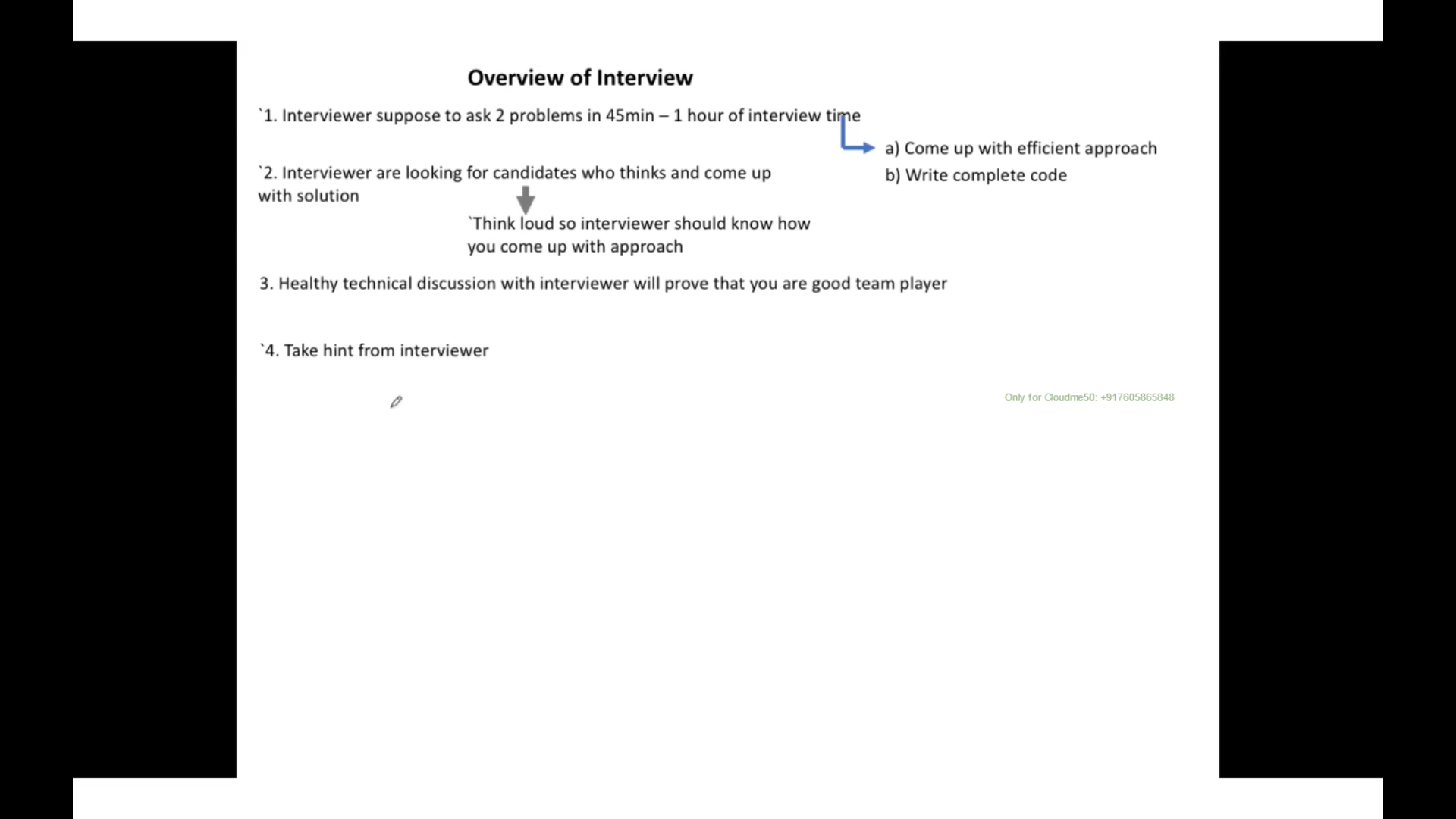
# Learning Sequence



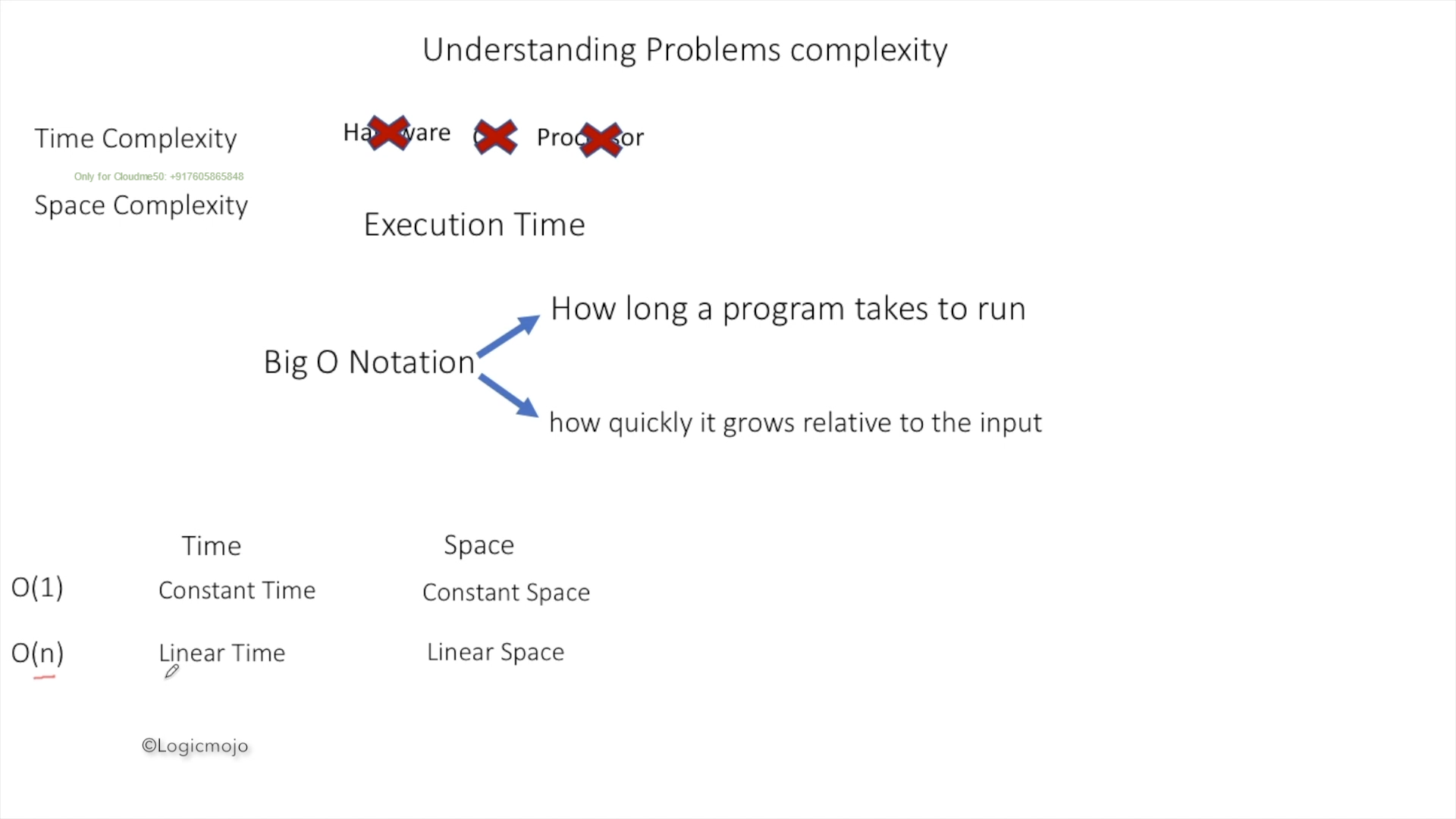


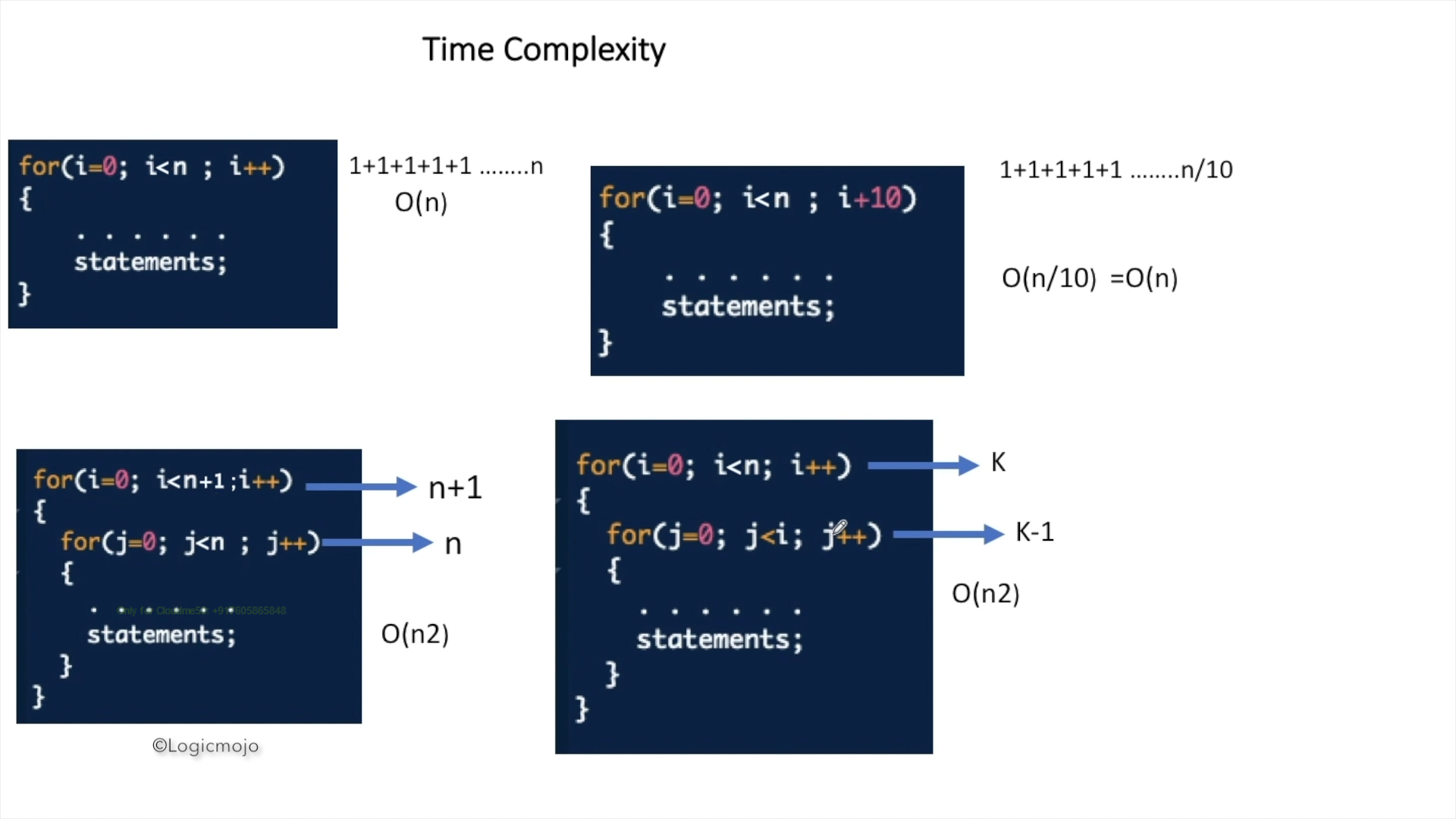


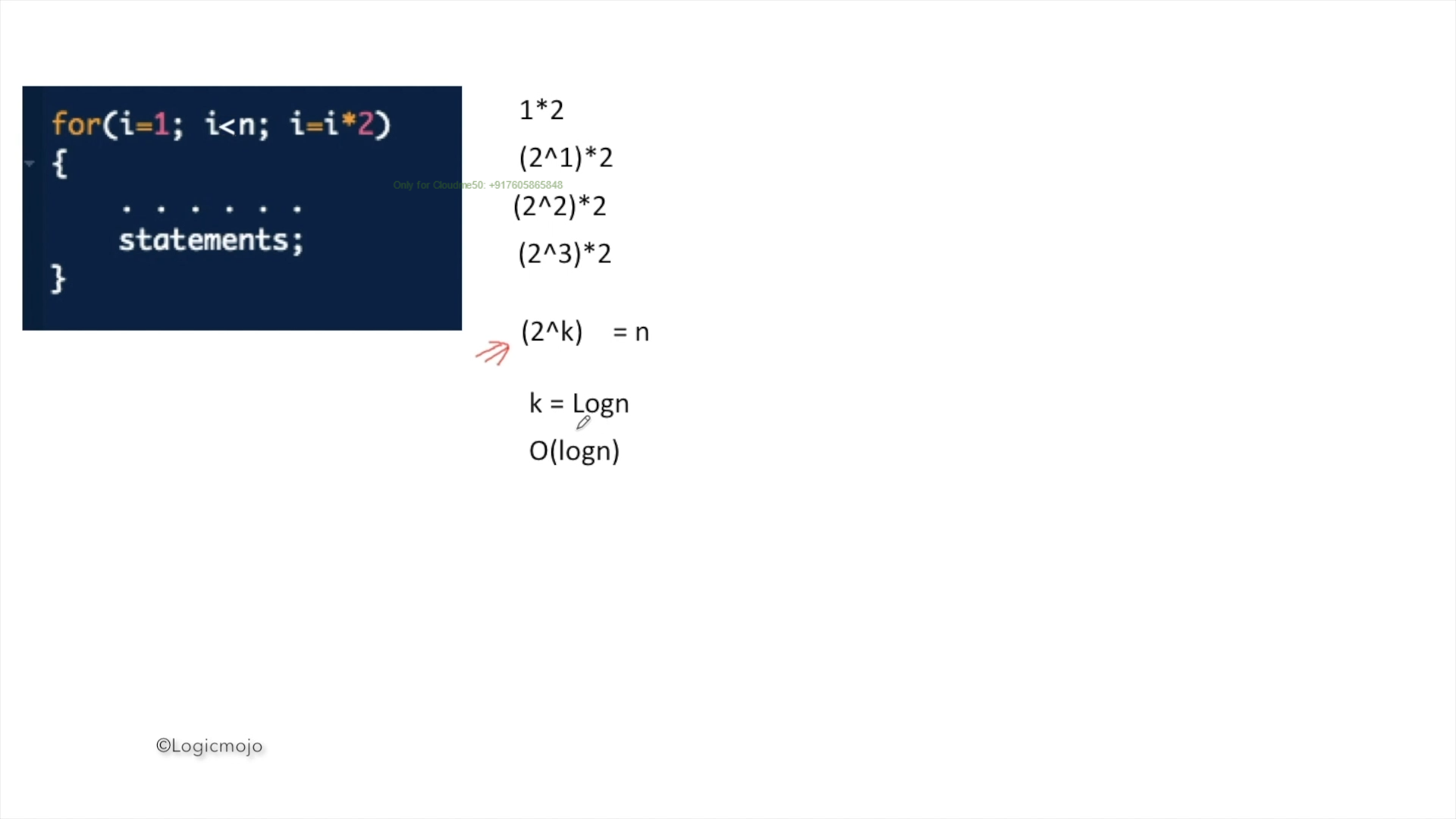
# **Space and Time Complexity of Algorithms by Using Big O Notation**

## **Time Complexity**

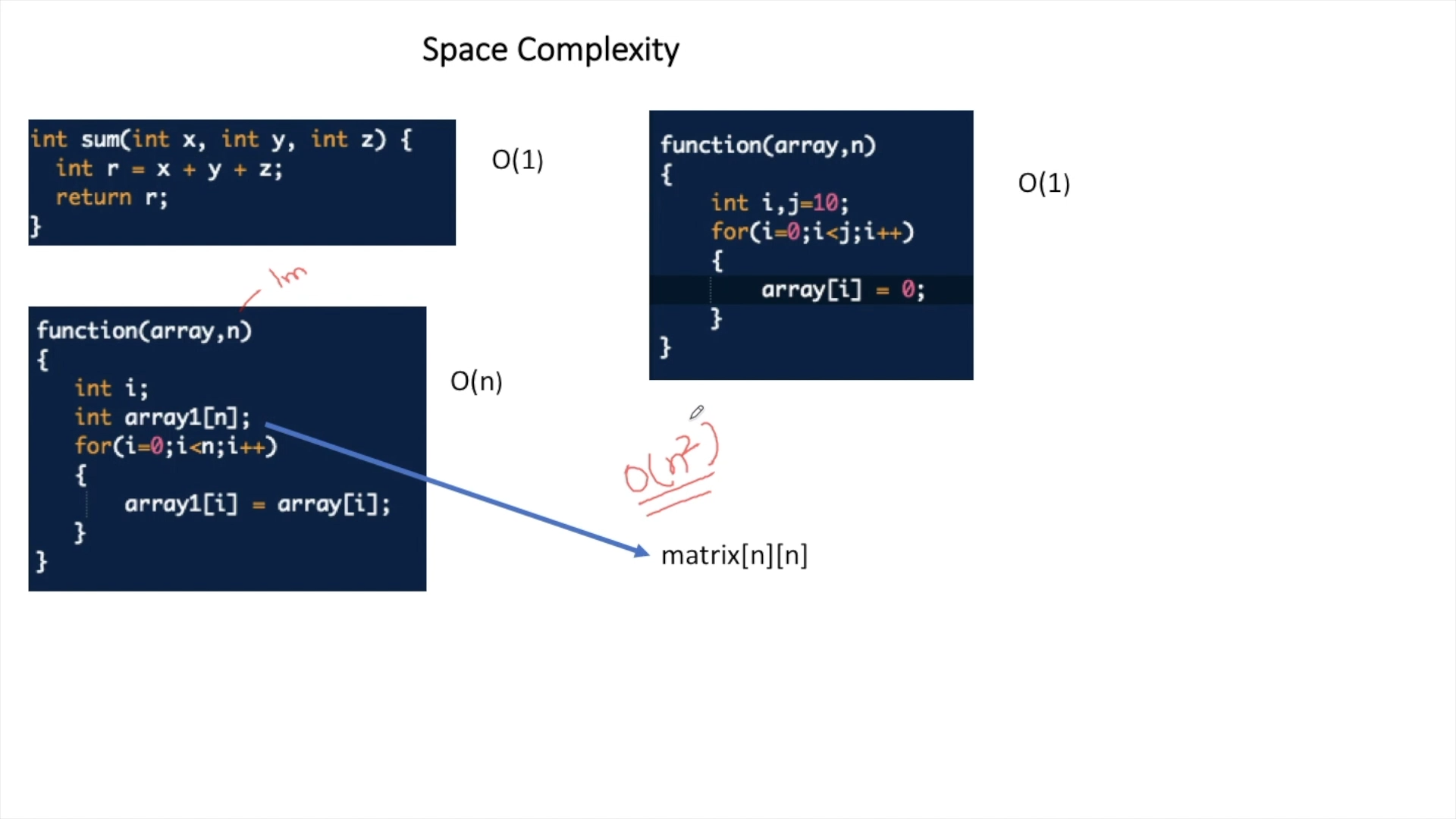
* + - One for loop – O(n)
    - Two nested for loop – O(n2)
    - Three nested for loop – O(n3)





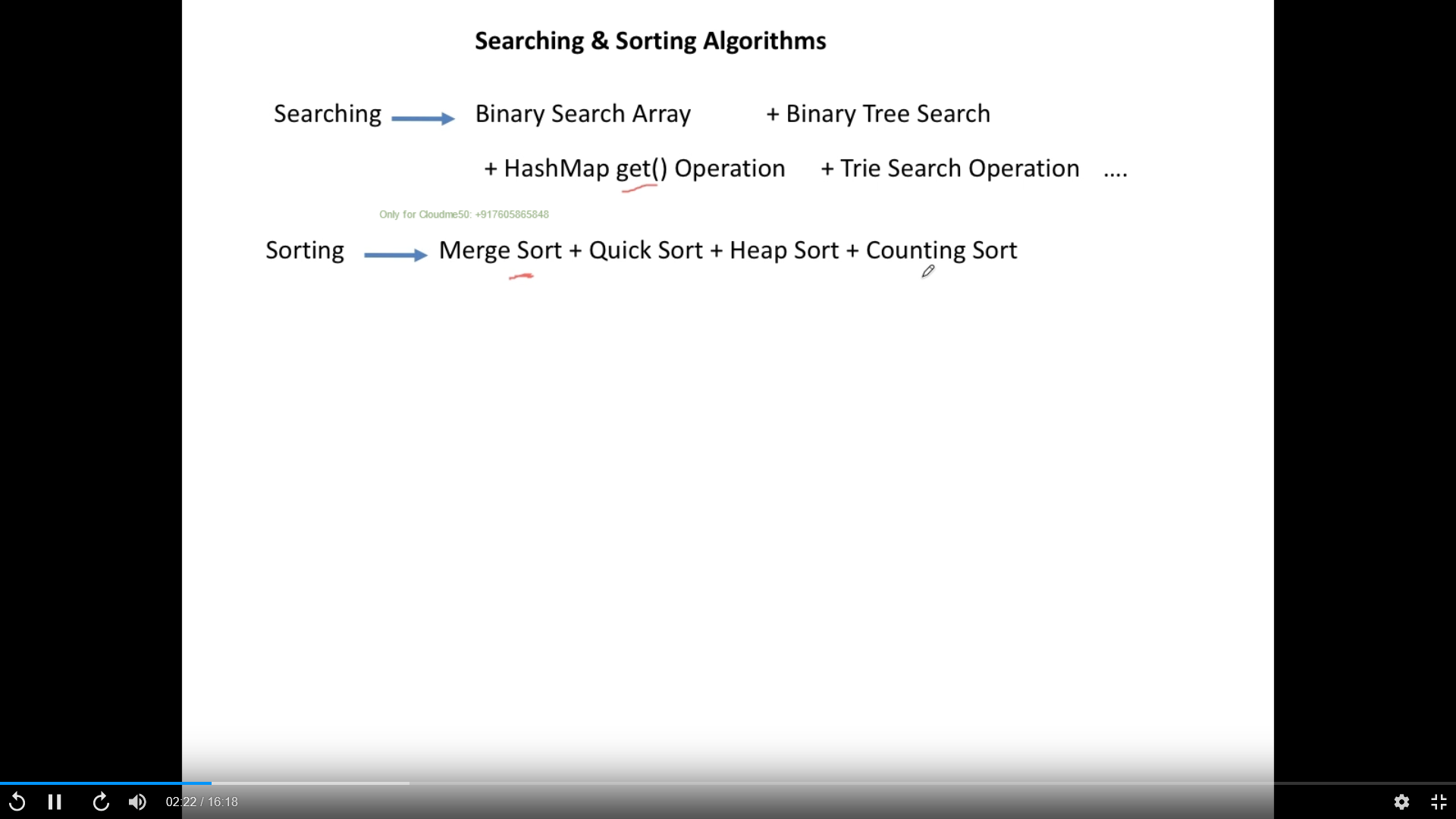


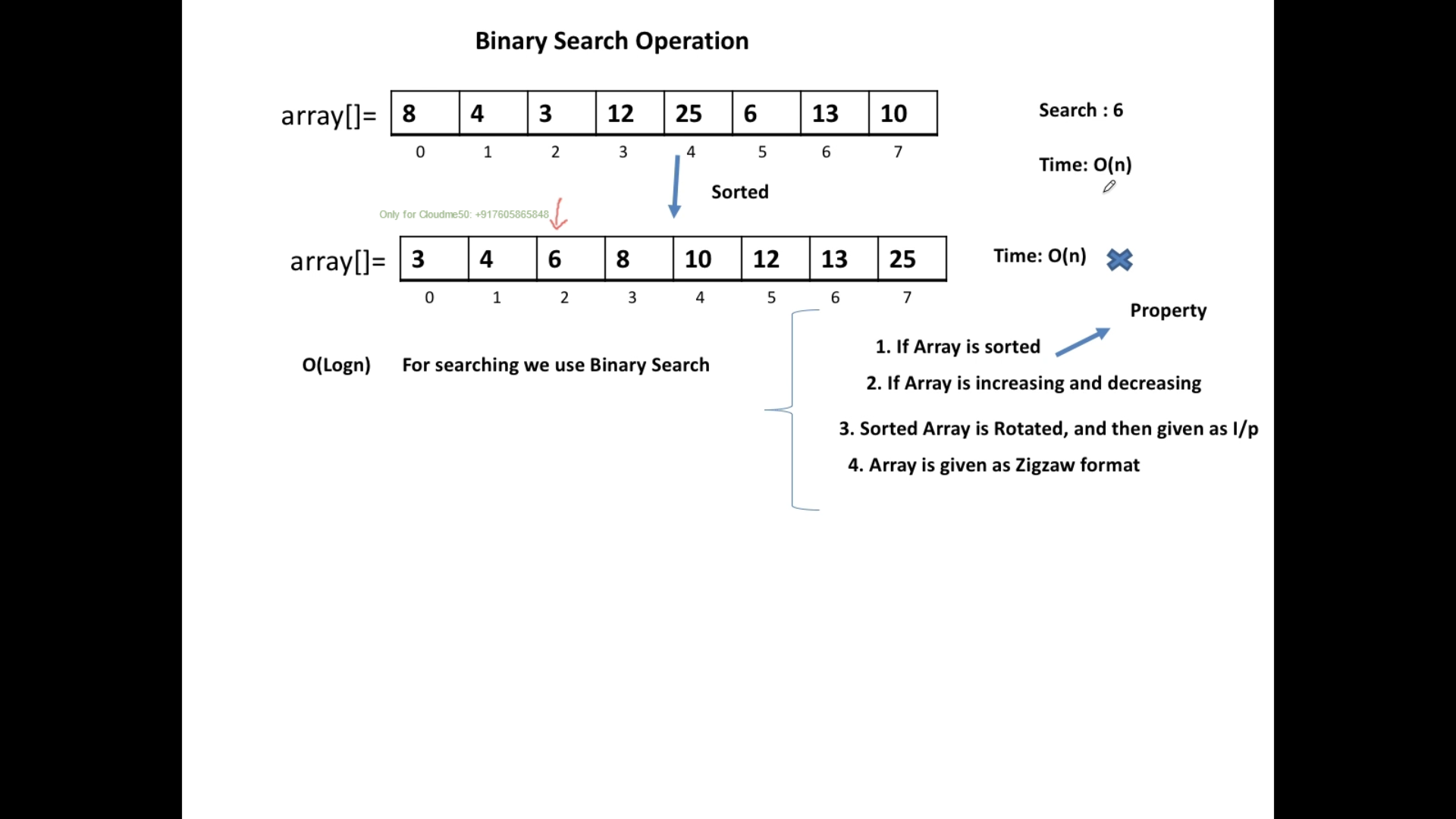
## **2.2 Space Complexity**

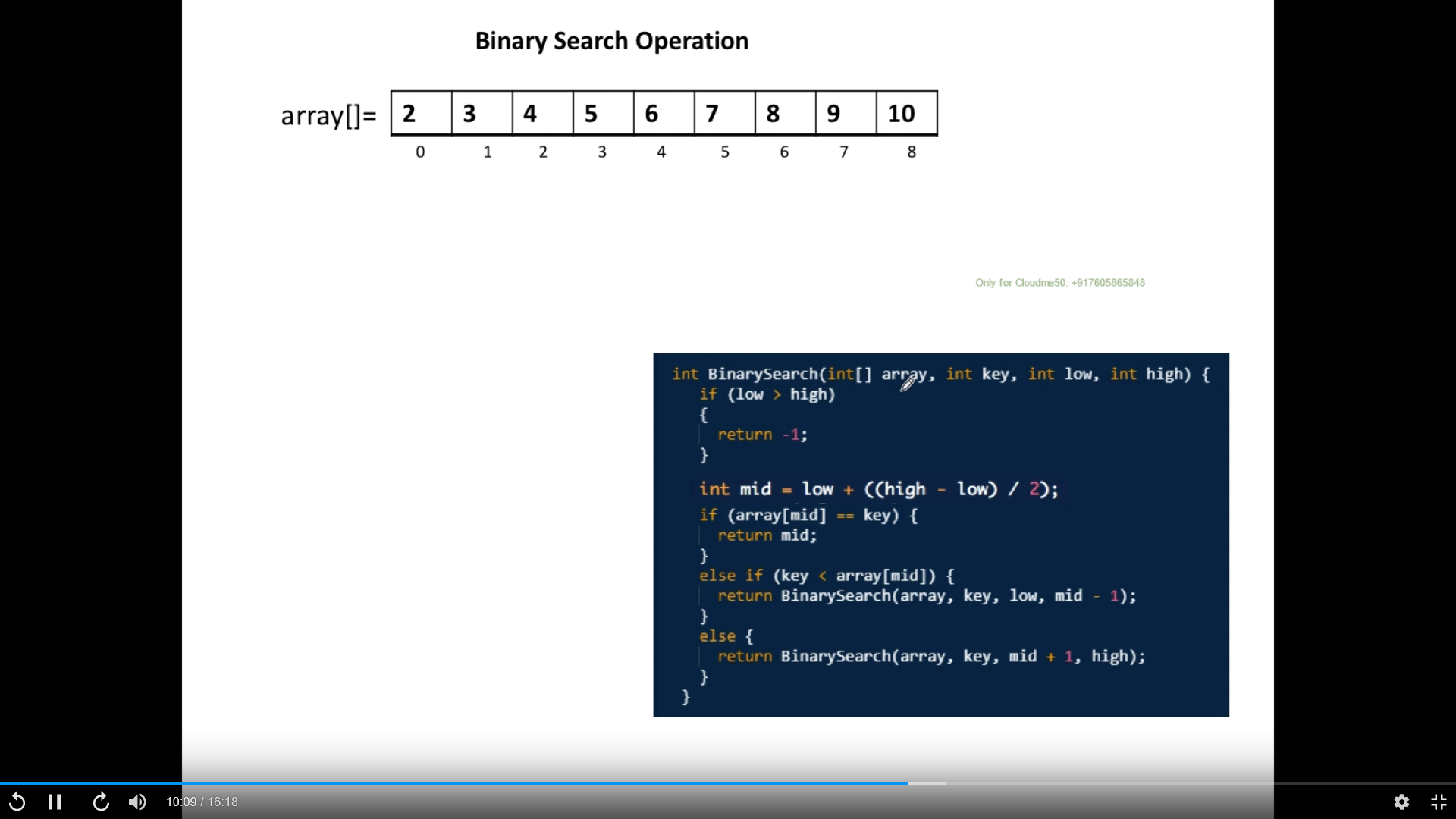


# Search and Sorting

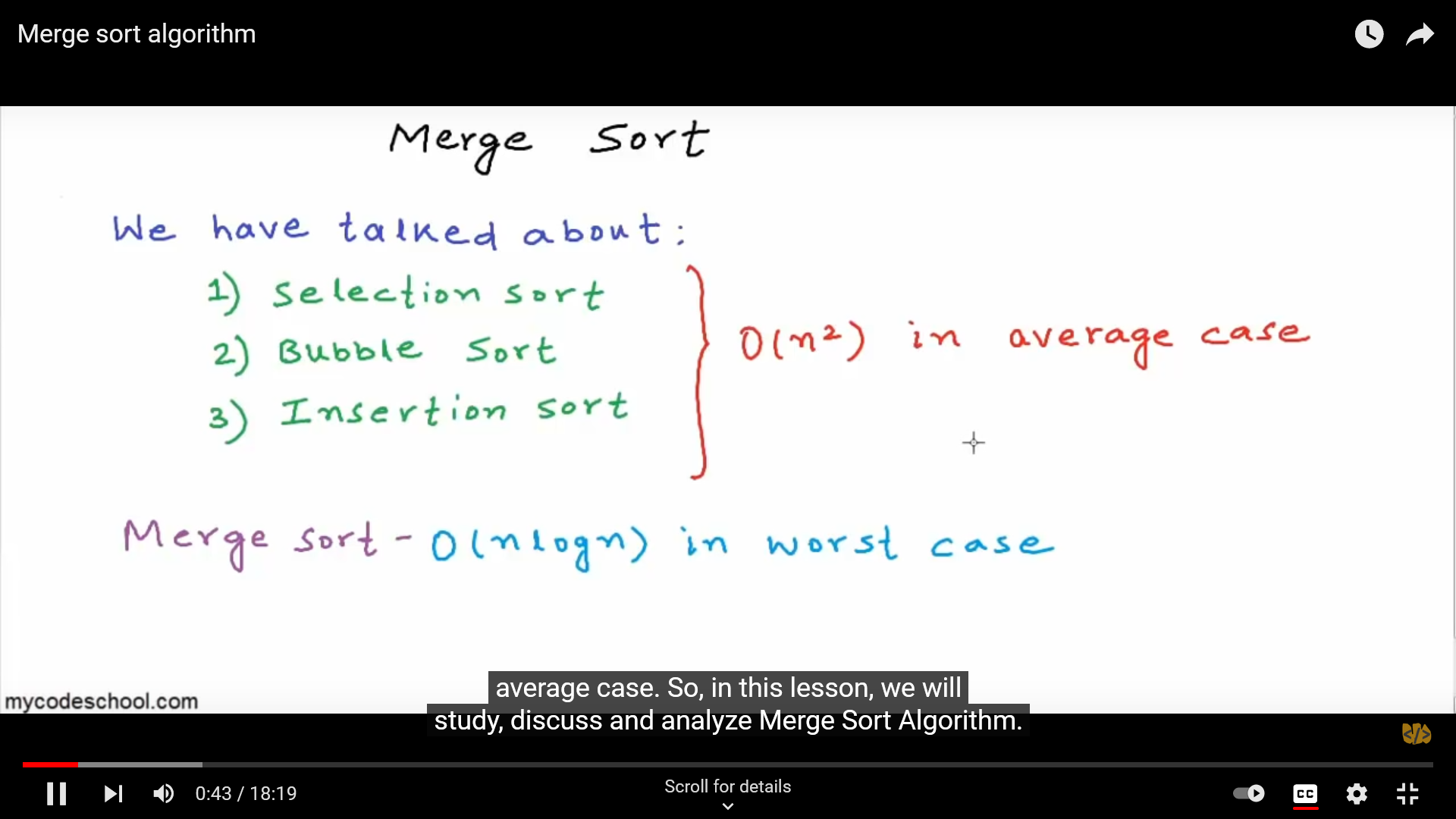
* Mostly Merge sort used 97 % following by Heap sort and Counting sort
* Binary search has O(logN) time

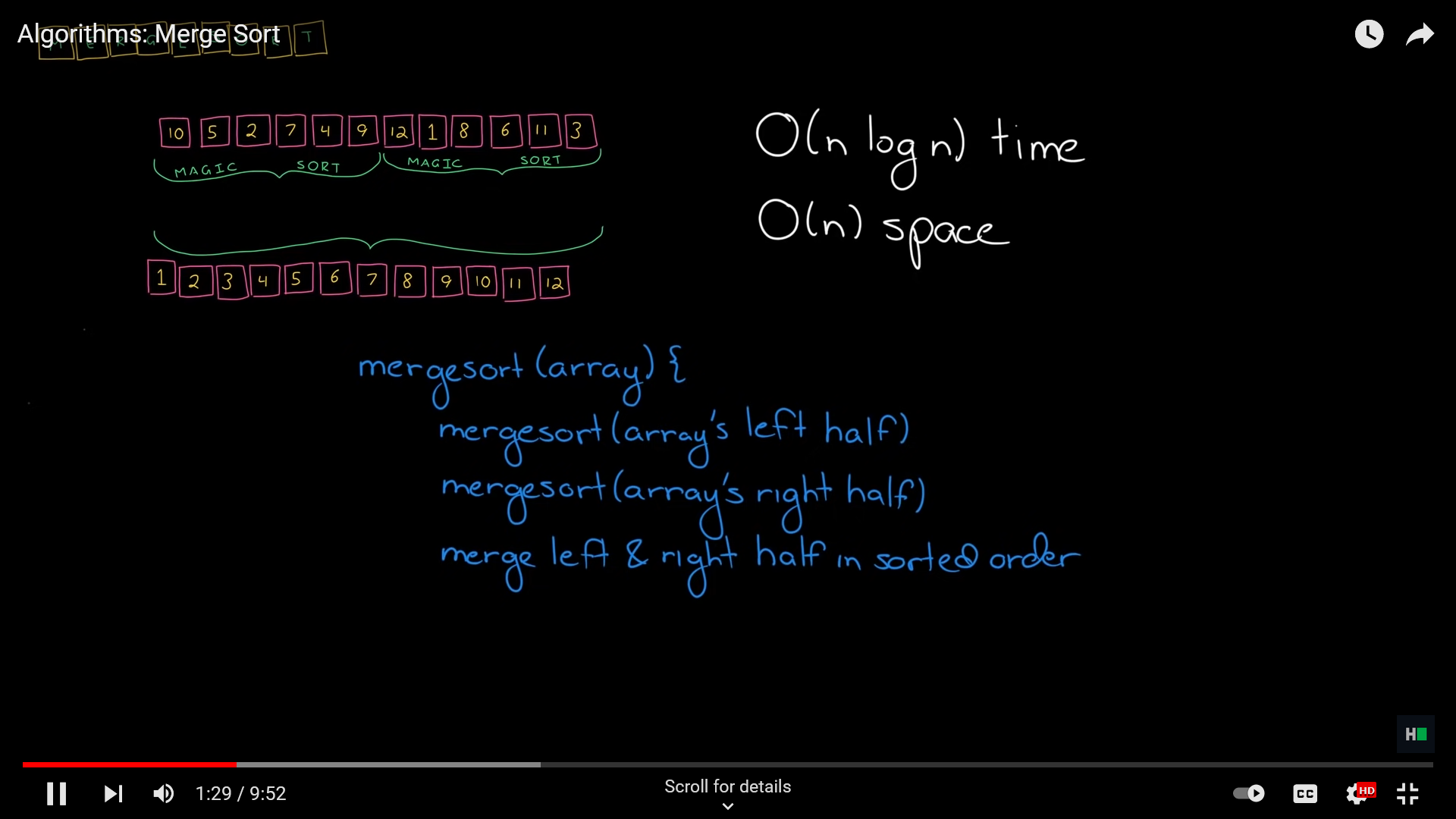


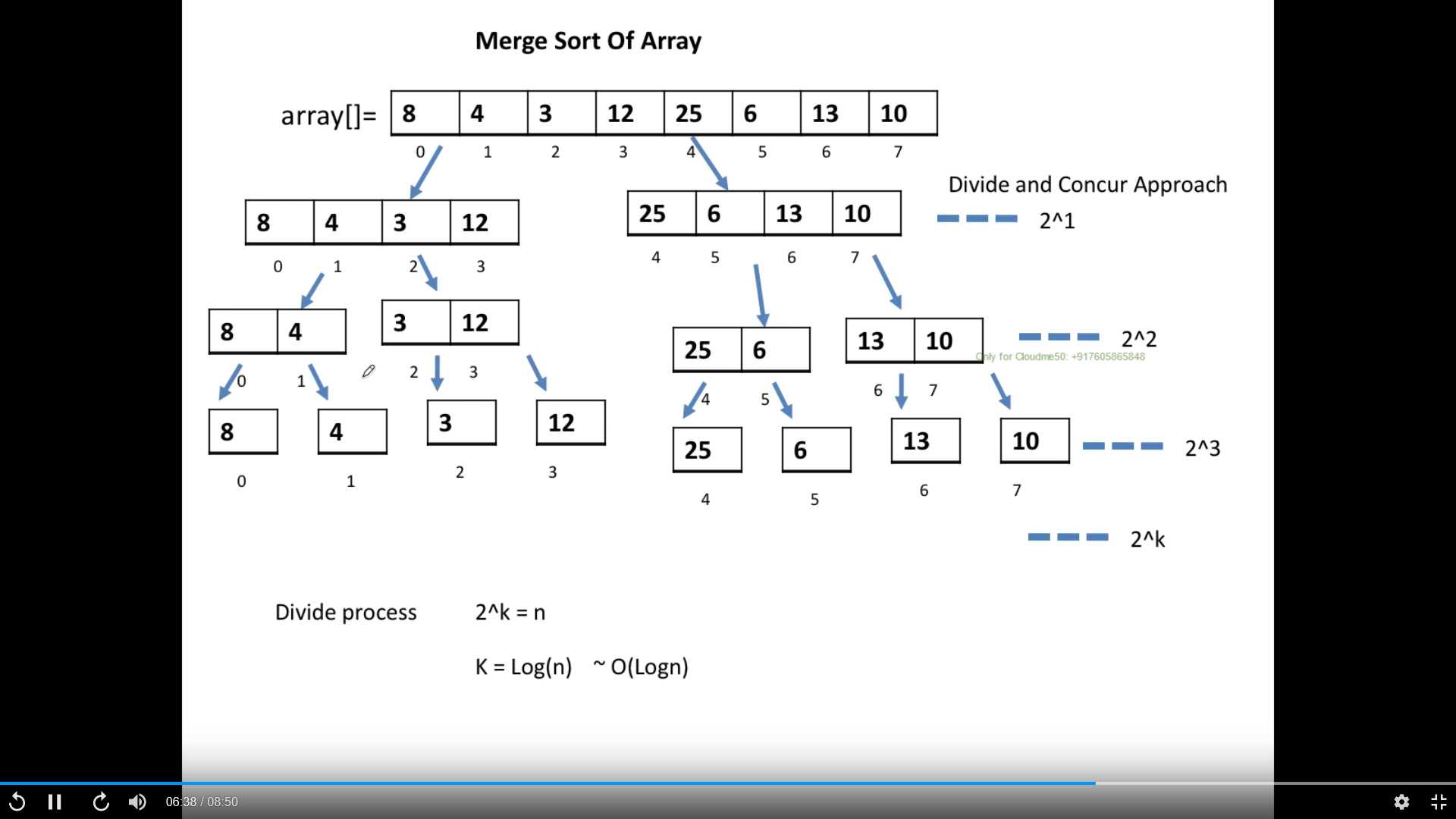


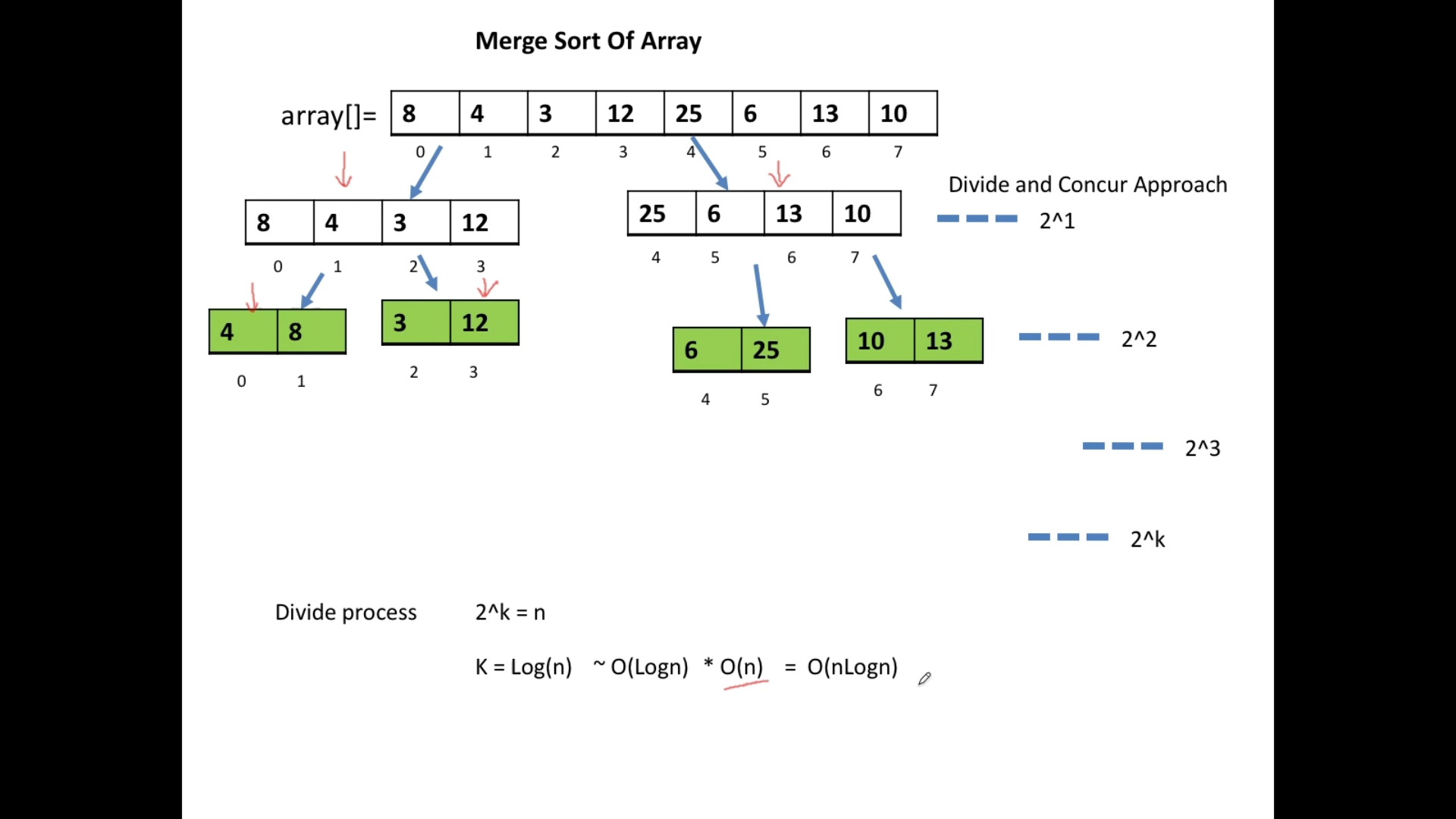


## Sorting Algorithm –

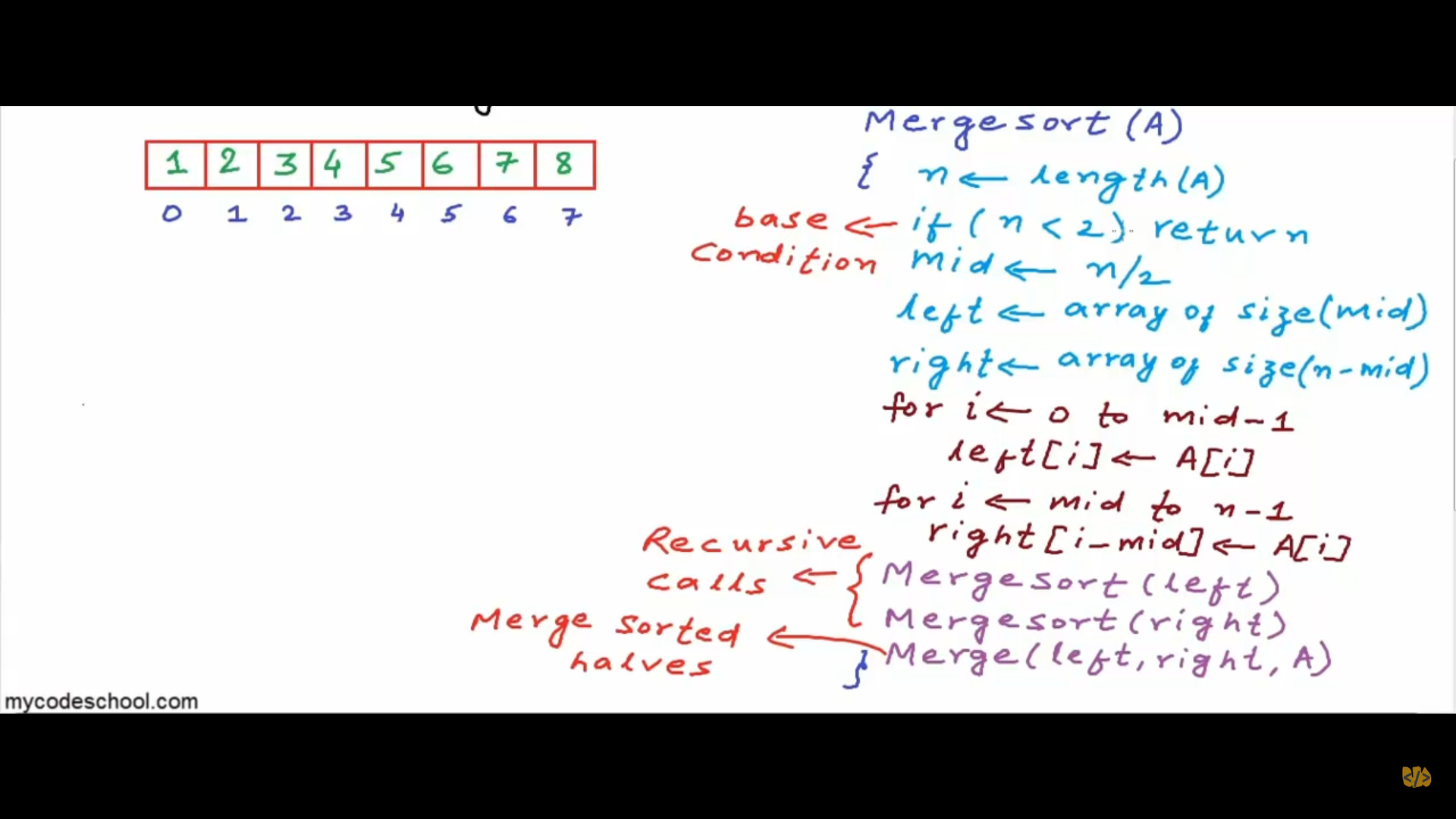
* merge sort used 98 %
  + Divide and concur approach
    - Keep diving an array until it has only one element
* 

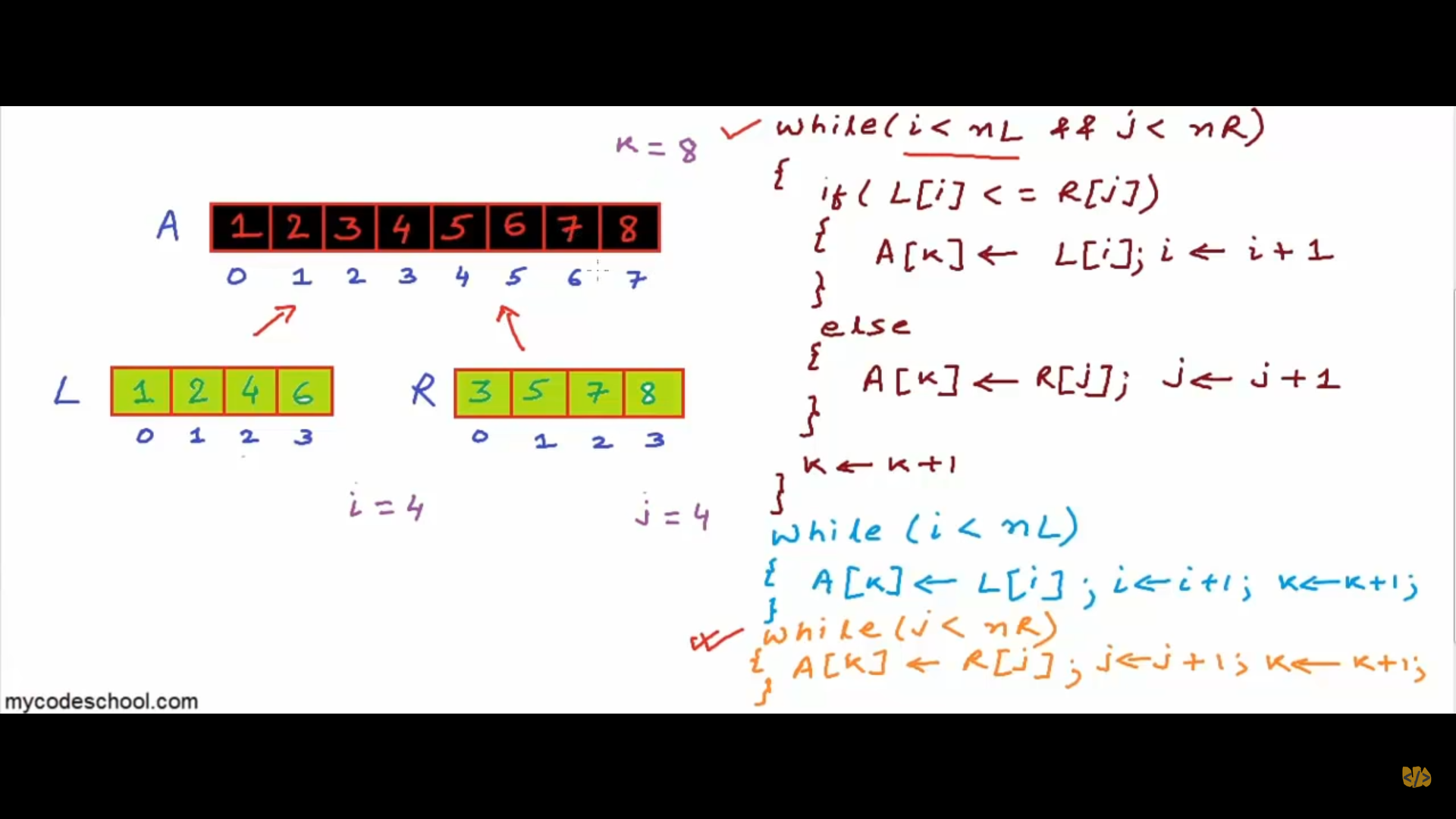






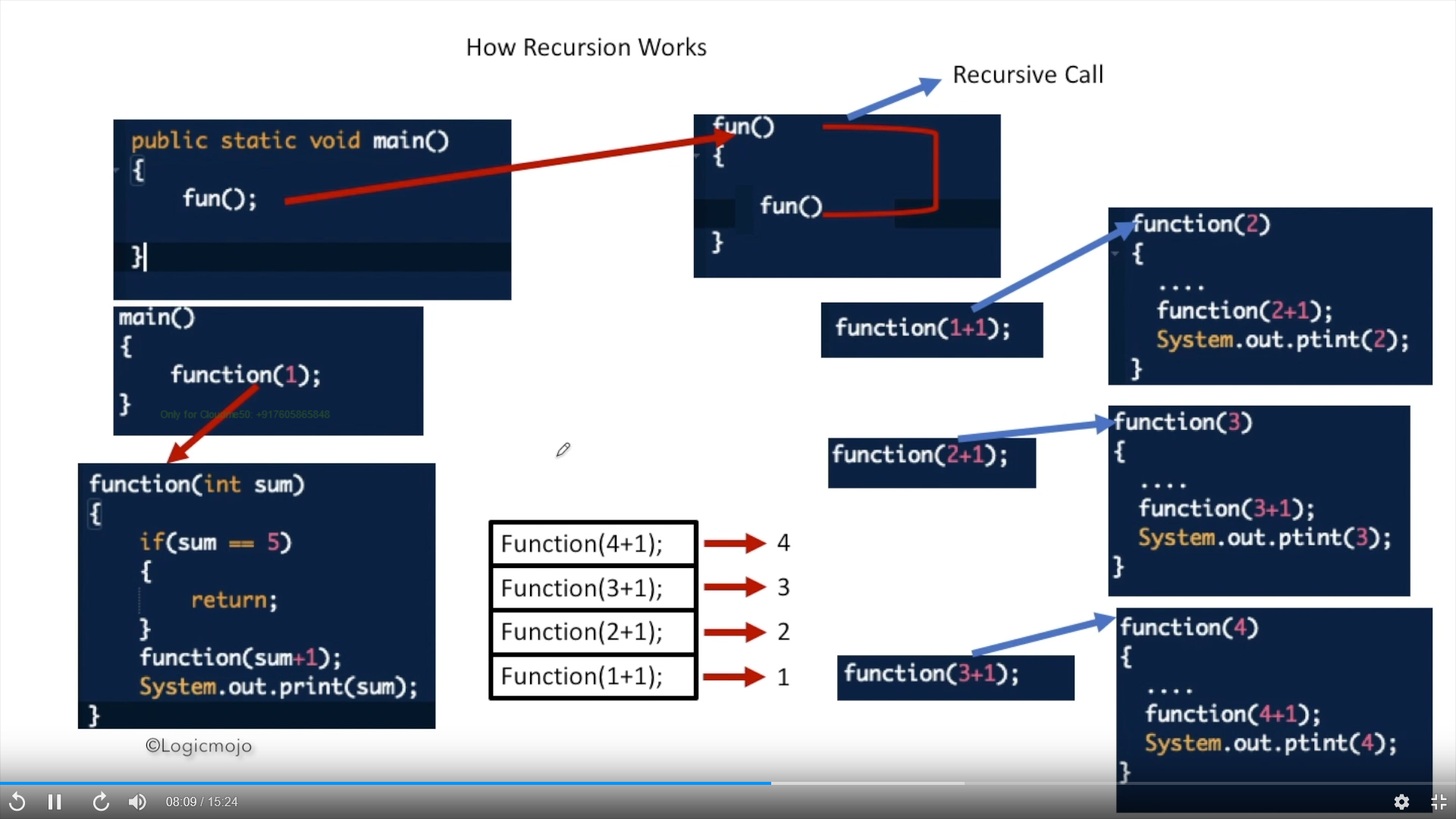
## Merge Sort solution from youtube

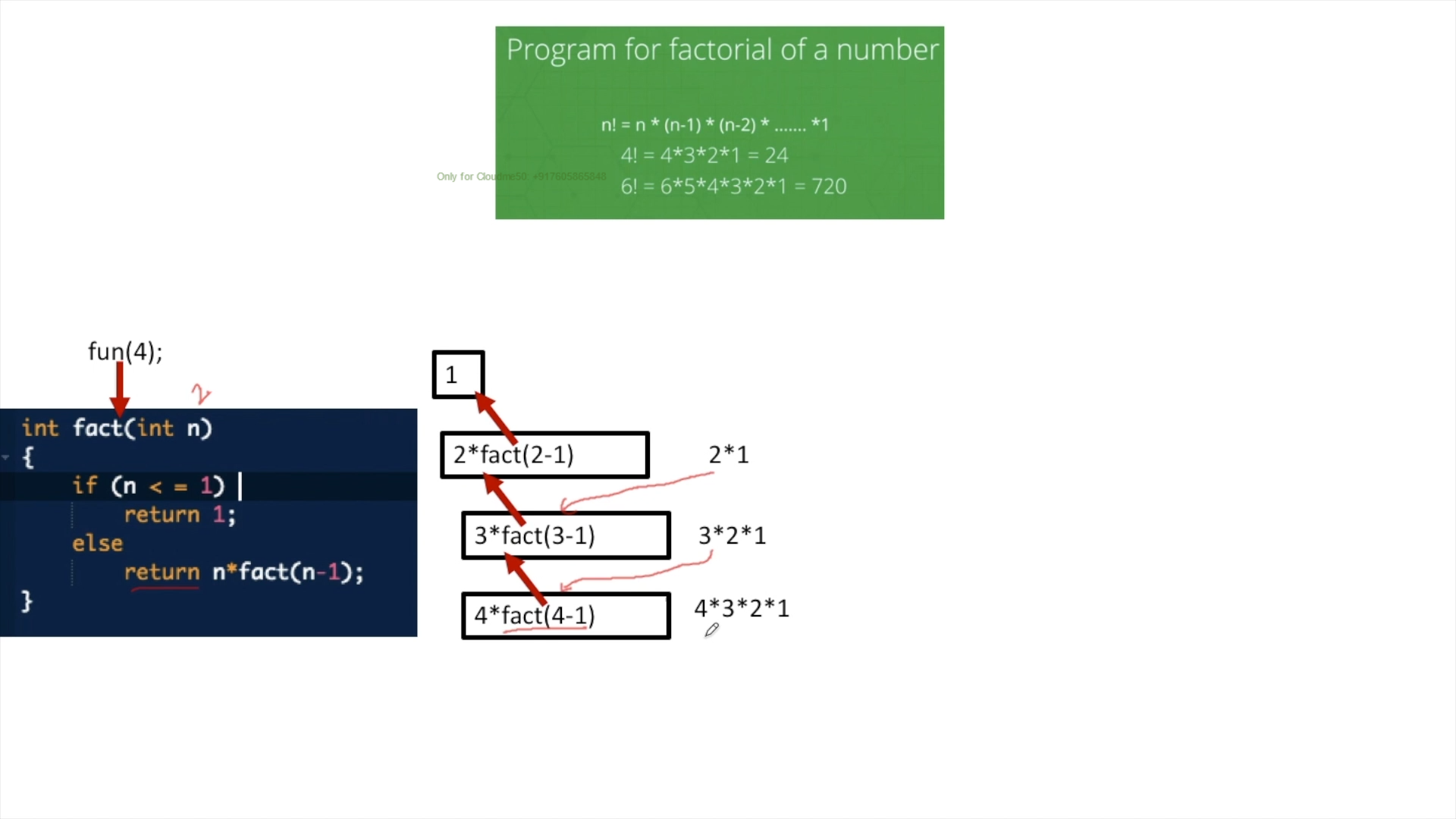


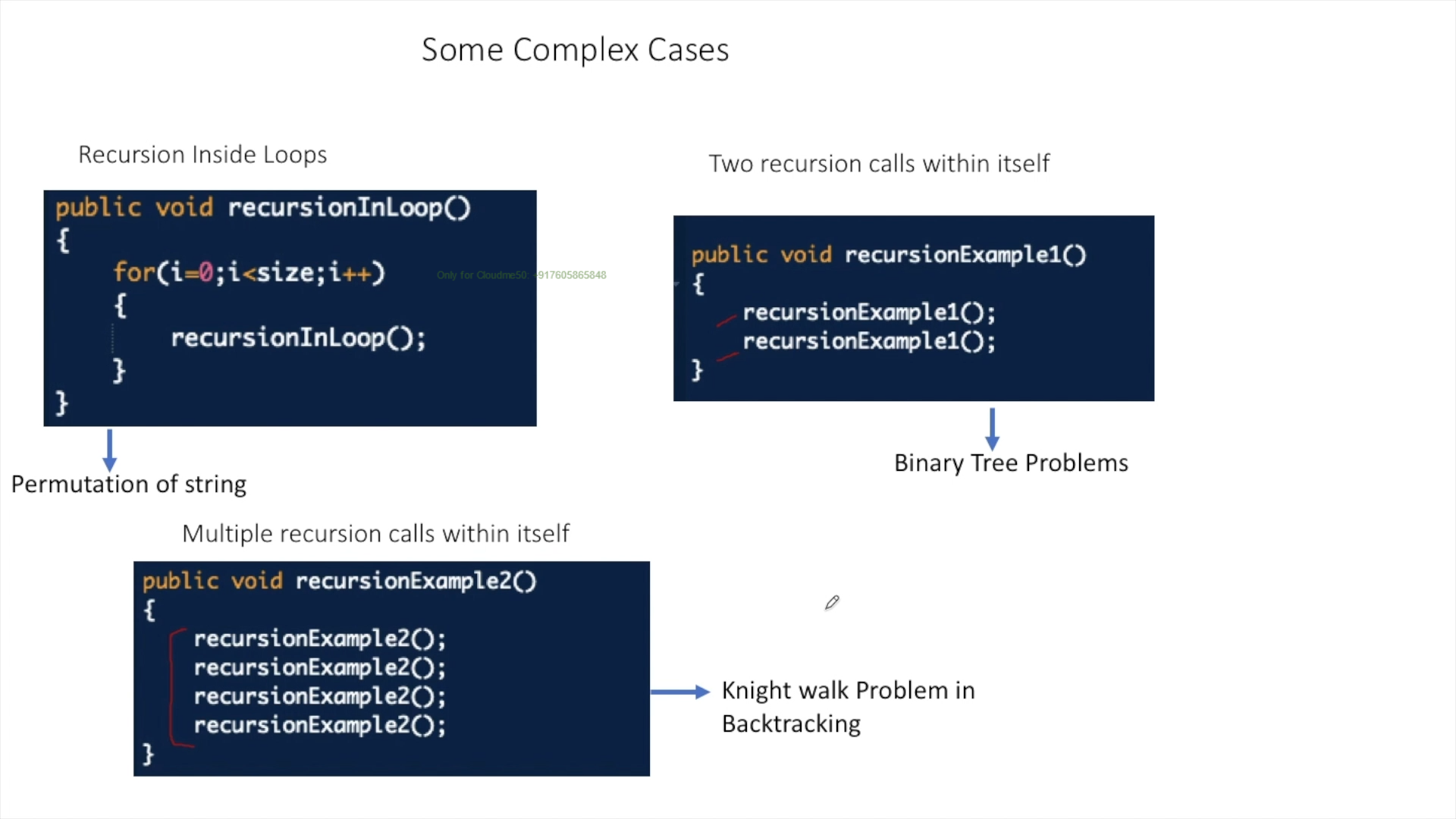


# Recursion

* **Main important** is that call get piled up as stack – LIFO – start executing from first
* Permutation of string
* Knight walk problem in backtracking







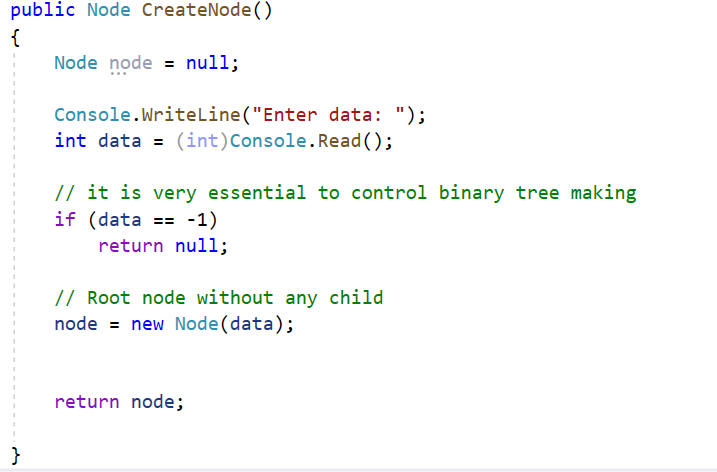
# Binary Tree

## Lab-1

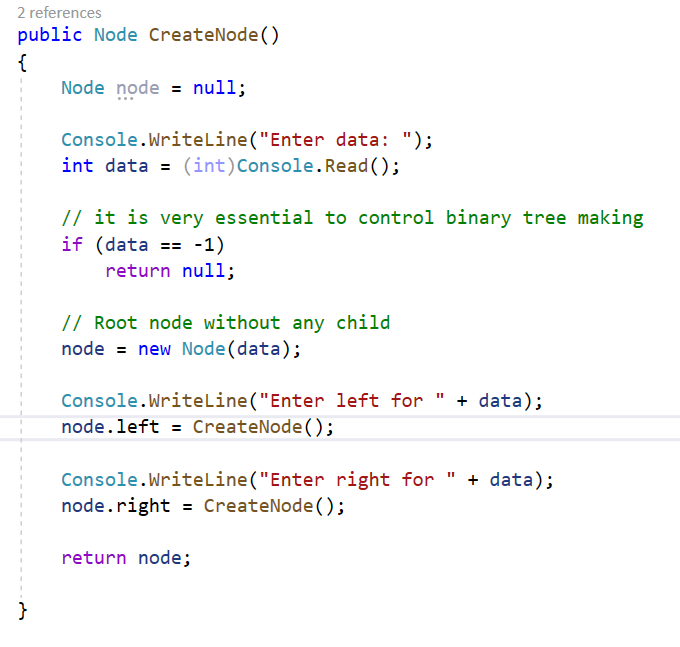
### Create a Node class



### Create a Binary Tree Parent node

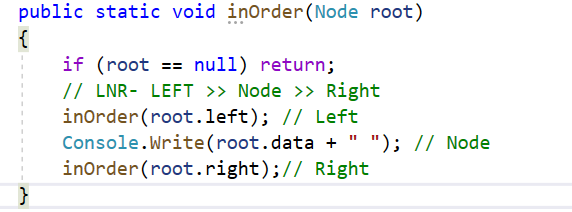


### Create Binary tree left and right node

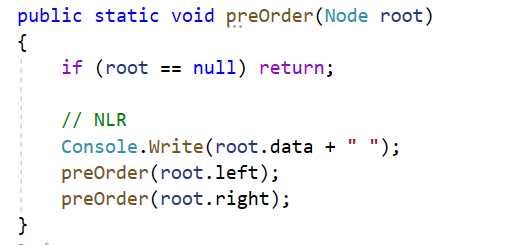


### Traverse the binary tree

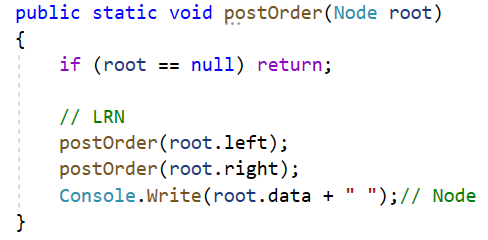
* + In order – **LNR**



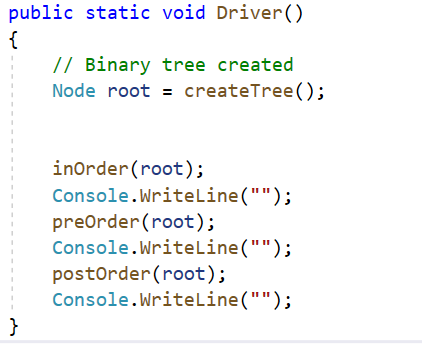
* + Pre- order – **NLR**



* + Post Order LRN



### Call tree and invoke it



* Final

