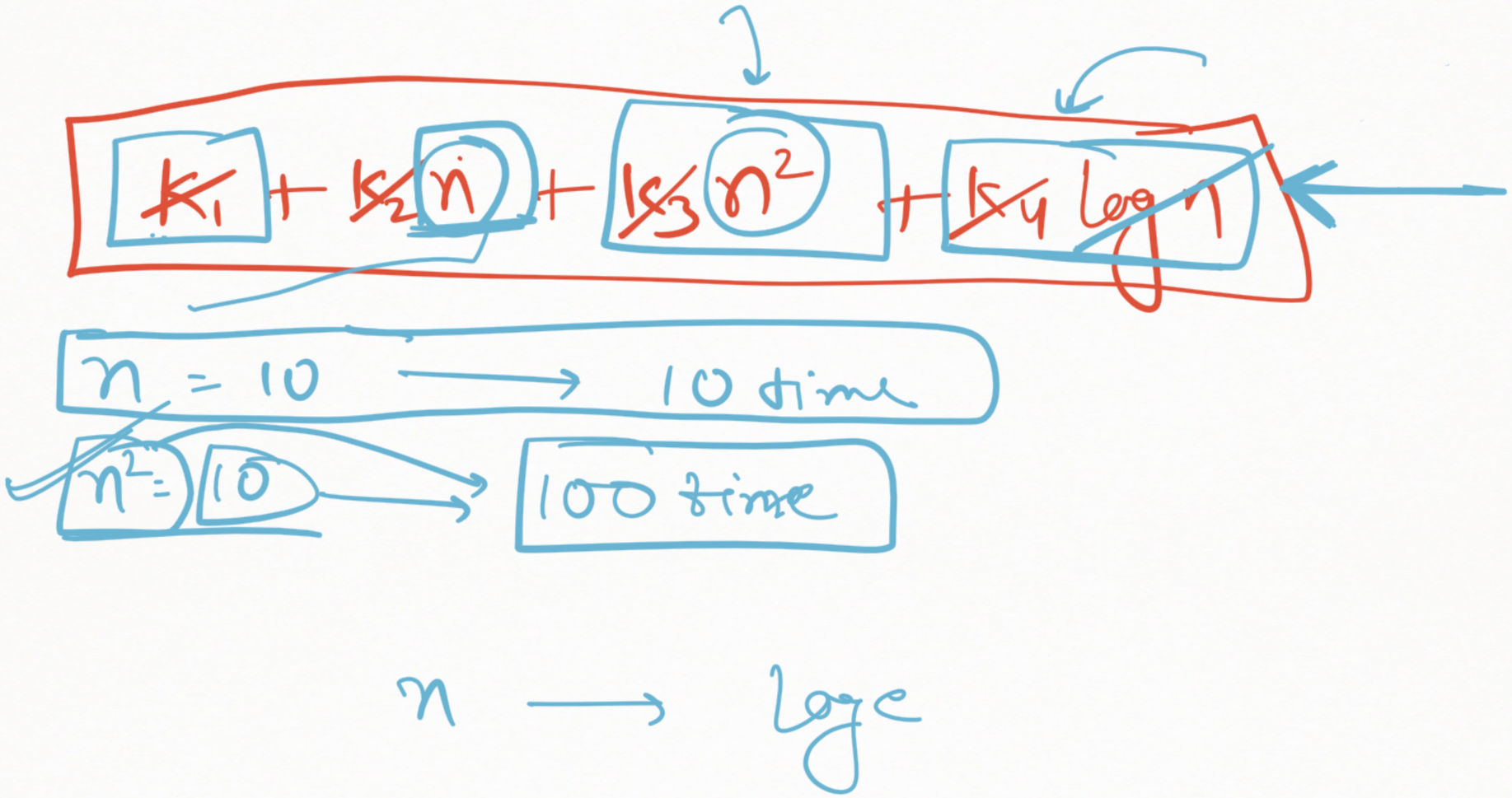
Notation

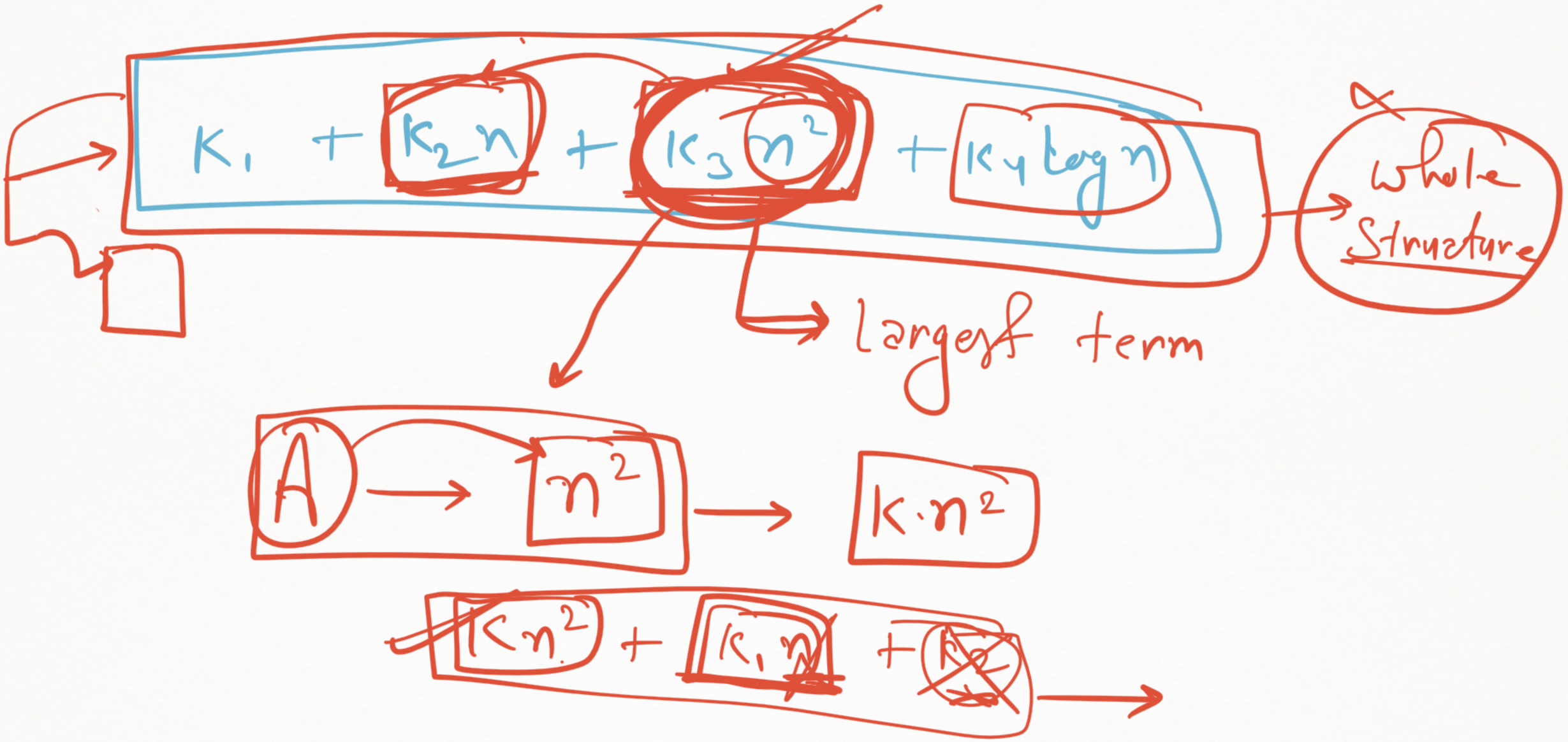
Big O Notation

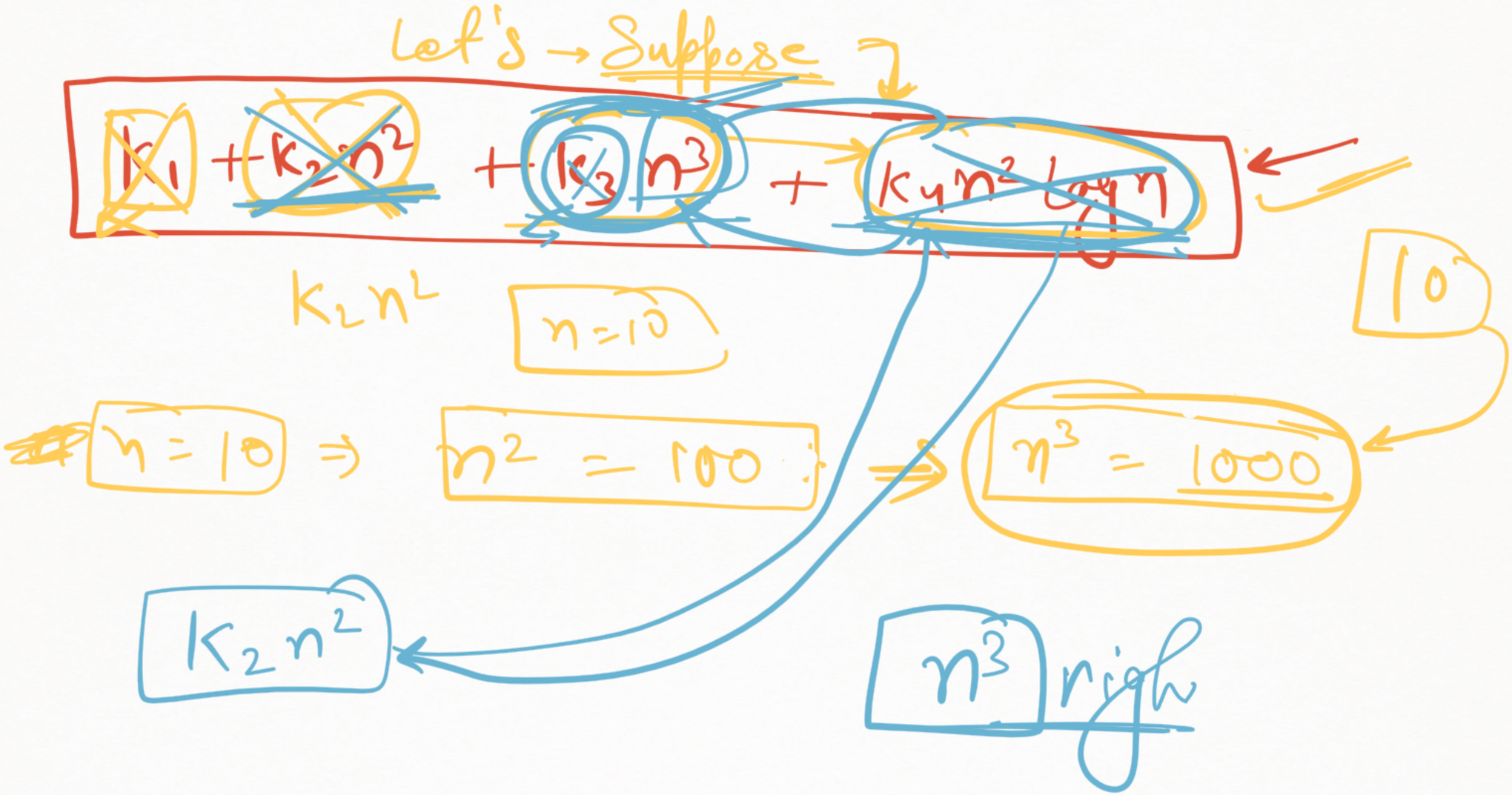
$k_1 + k_2 n$ operation

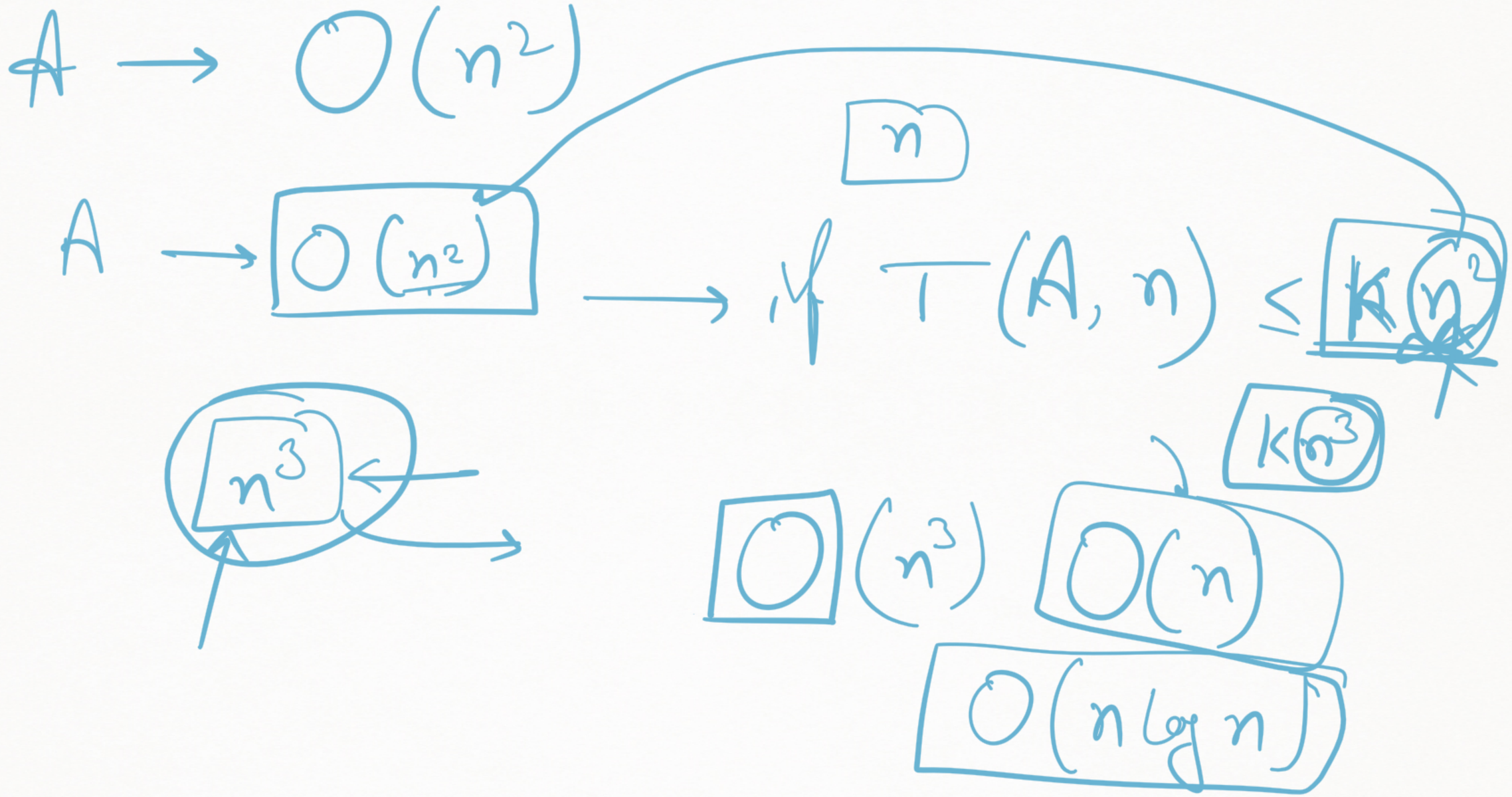
$k_1 + k_2 n + k_2 n^2$ operation

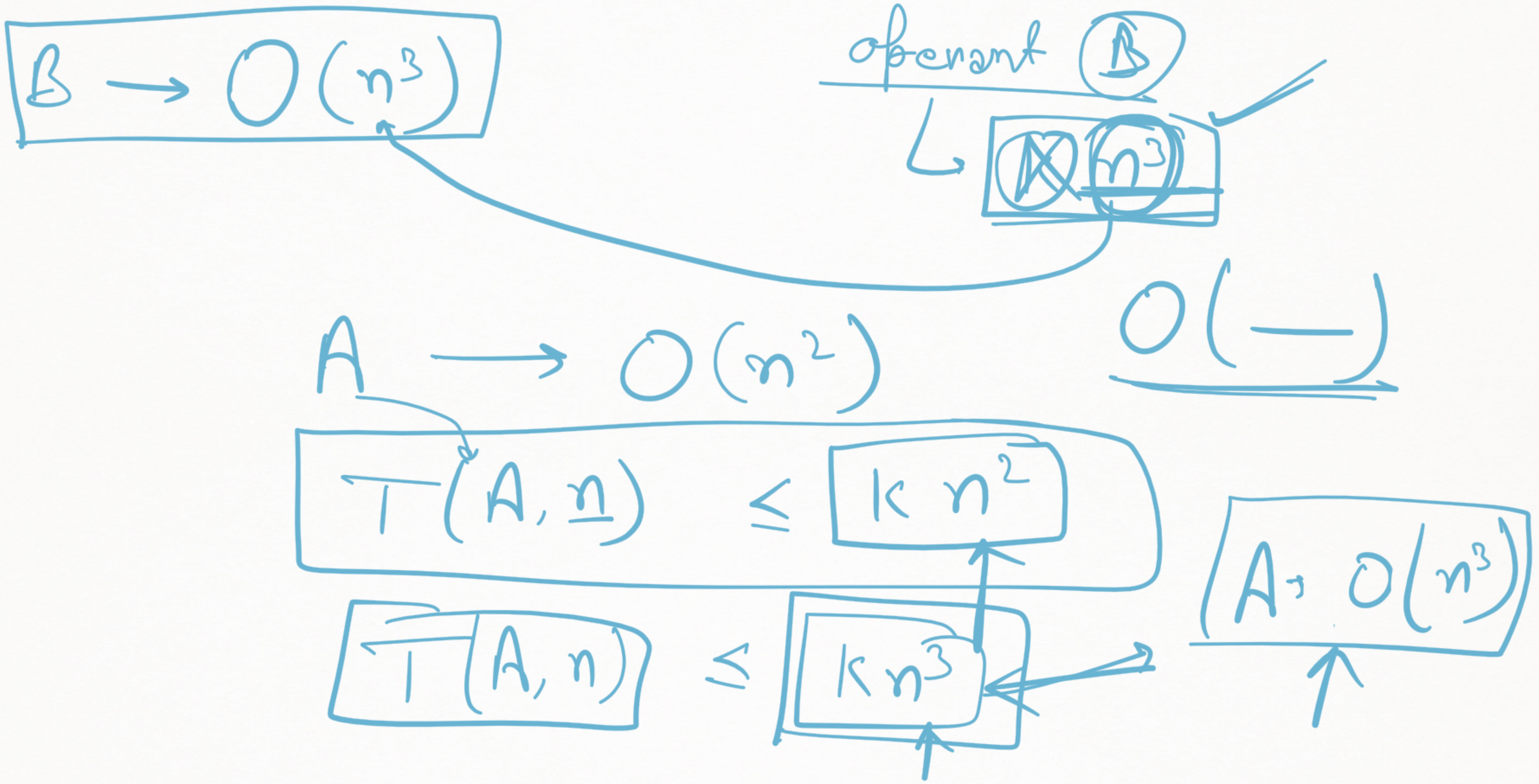
$k_1 + k_2 n + k_3 n^2 + k_4 \log n$ operation

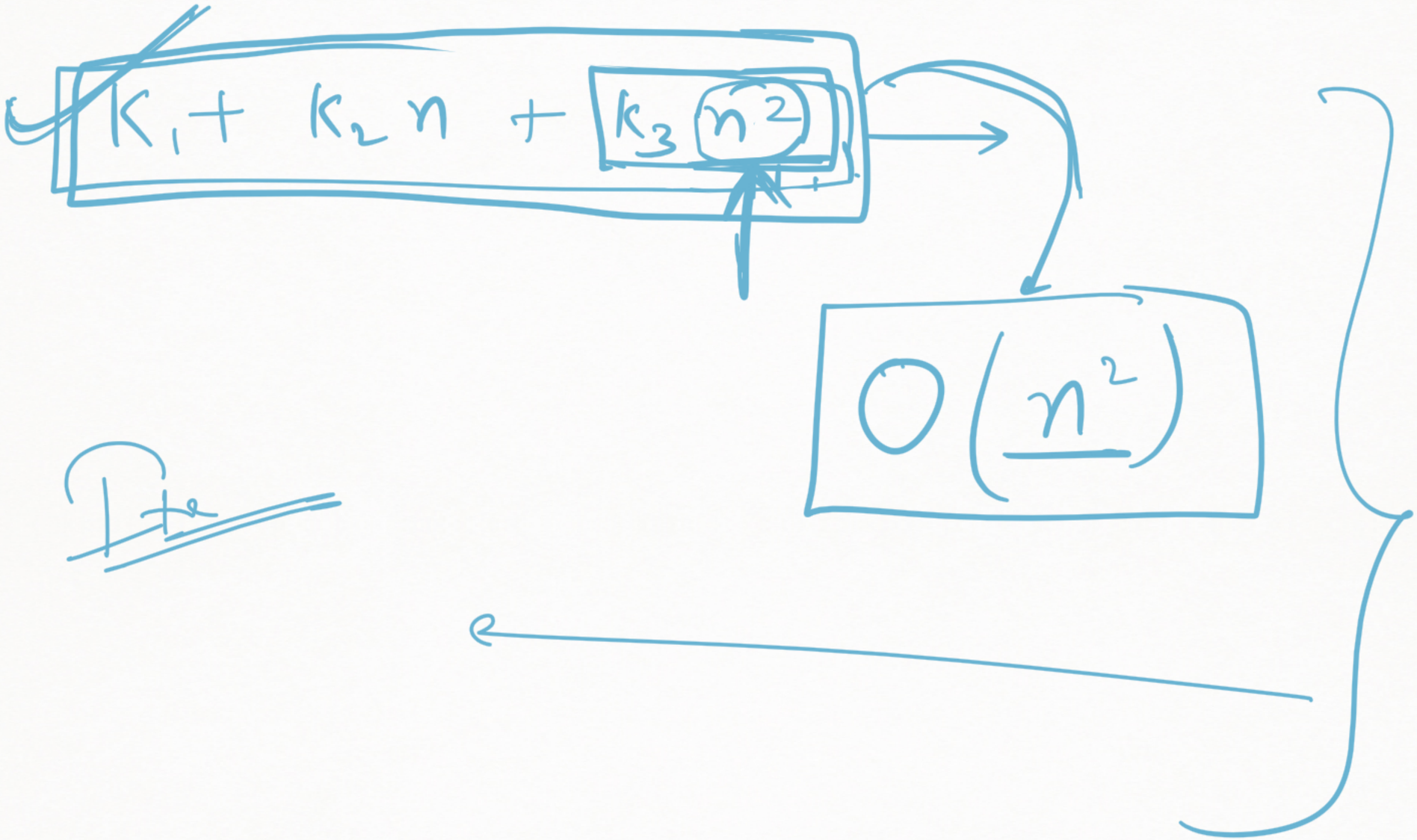

$k_1 + k_2 n + k_3 n^2 + k_4 \log n$
$n = 10$ → 10 time
$n^2 = 10$
100 time
$n \longrightarrow \log e$


$K_1 + K_2 n + K_3 n^2 + K_4 \log n$
whole Structure
largest term
A
n^2
$K \cdot n^2$
$K n^2 + K_1 n +$

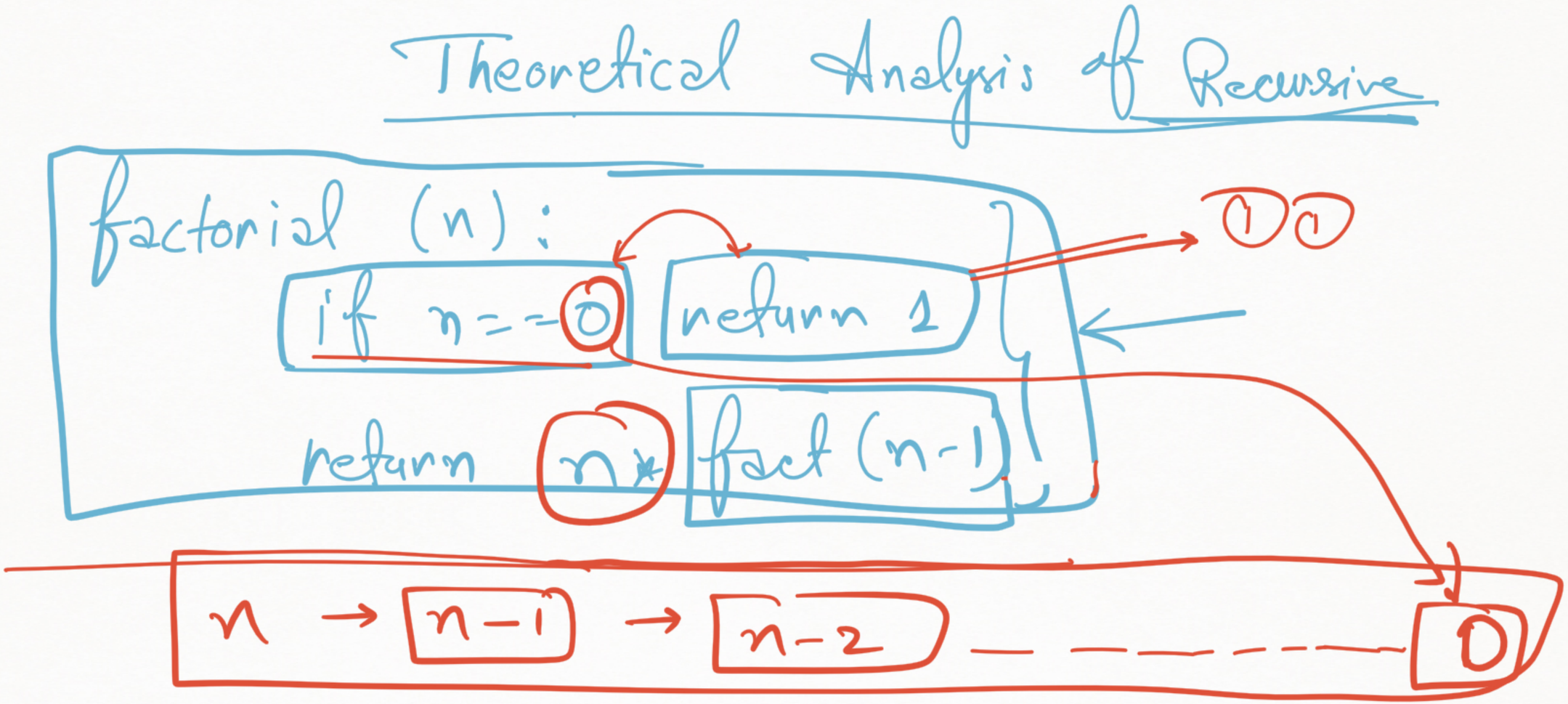

Let's → Suppose
$k_1 + k_2 n^2 + k_3 n^3 + k_4 n^2 \log n$
$k_2 n^2$
$n = 10$
10
$n = 10 \Rightarrow$
$n^2 = 100$
$n^3 = 1000$
$k_2 n^2$
n^3 right

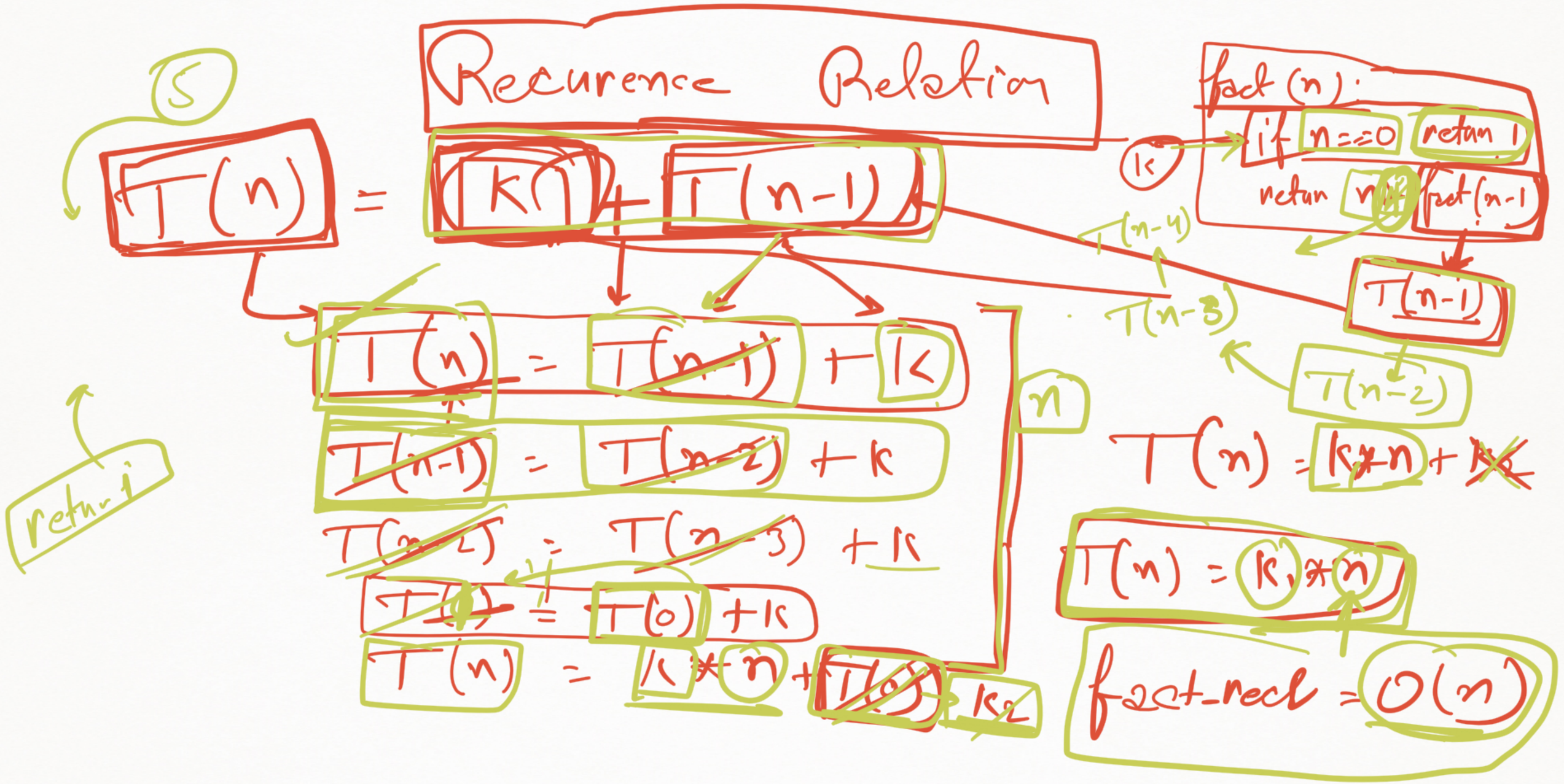

A → O(n²)
A → O(n²)
n
if T(A, n) ≤ kn²
kn³
n³
O(n³)
O(n)
O(n log n)


B → O(n³)
operant B
n³
O(—)
A → O(n²)
T(A, n) ≤ k n²
T(A, n) ≤ k n³
A → O(n³)

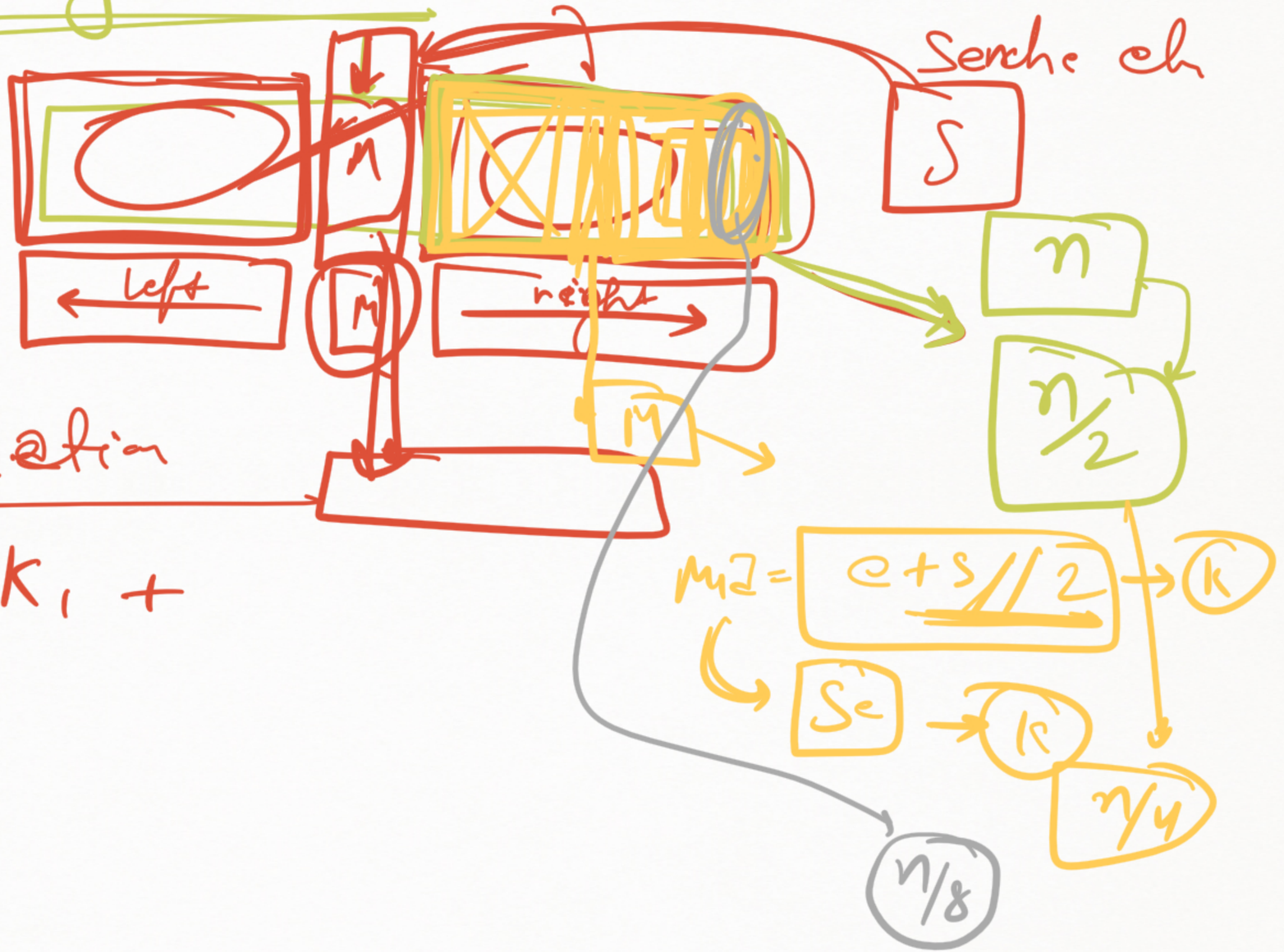

$k_1 + k_2 n + k_3 n^2$
$O(n^2)$

Theoretical Analysis of Recursive



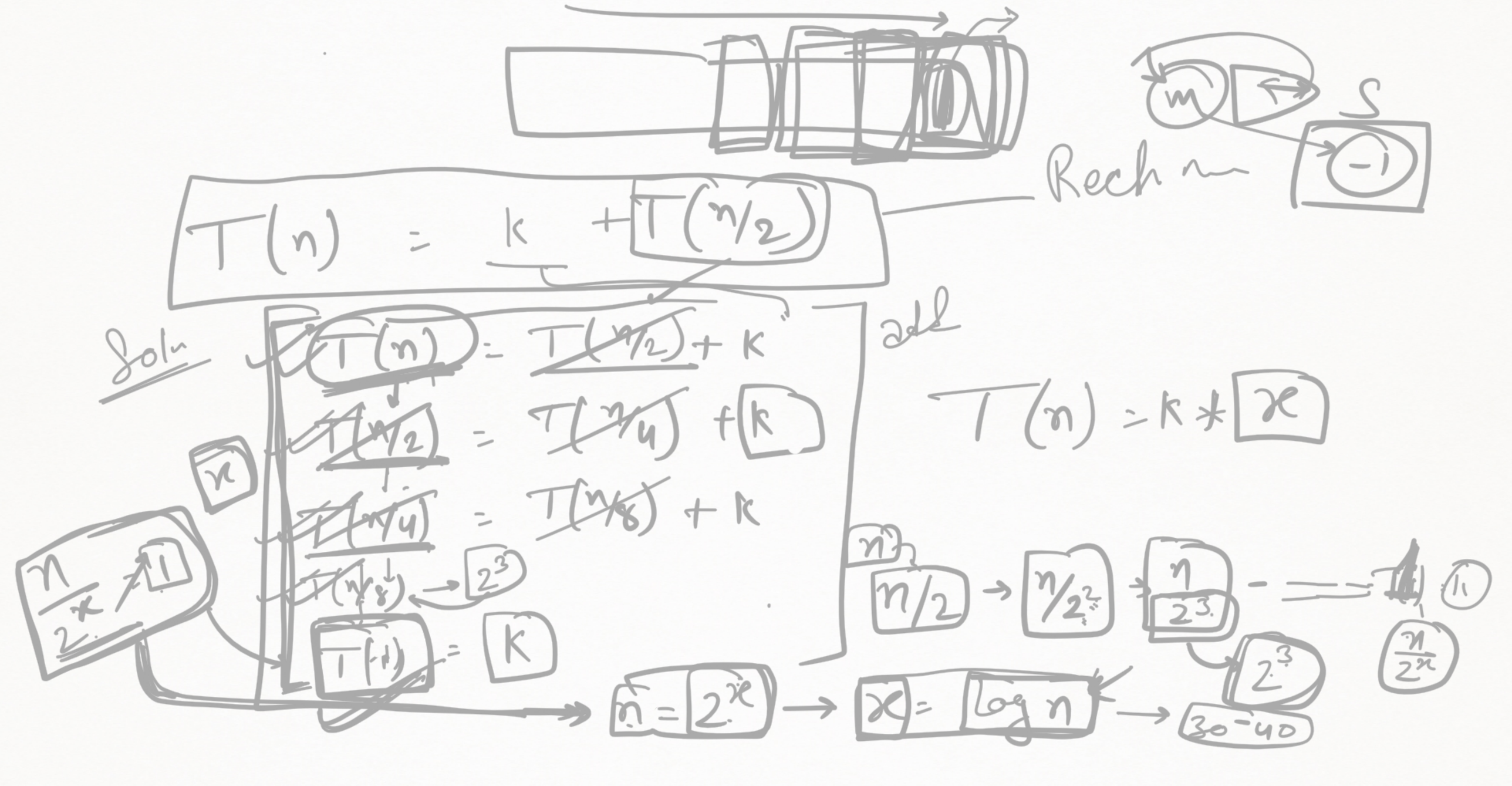

Recurence Relation
T(n) = K + T(n-1)
S
return 1
T(n) = T(n-1) + K
T(n-1) = T(n-2) + K
T(n-2) = T(n-3) + K
T(0) = T(0) + K
T(n) = K*n + T(0)
n
K
fact(n):
if n==0 return 1
return n * fact(n-1)
T(n-1)
T(n-2)
T(n-3)
T(n-4)
T(n) = K*n + K
T(n) = K*n
fact_recl = O(n)

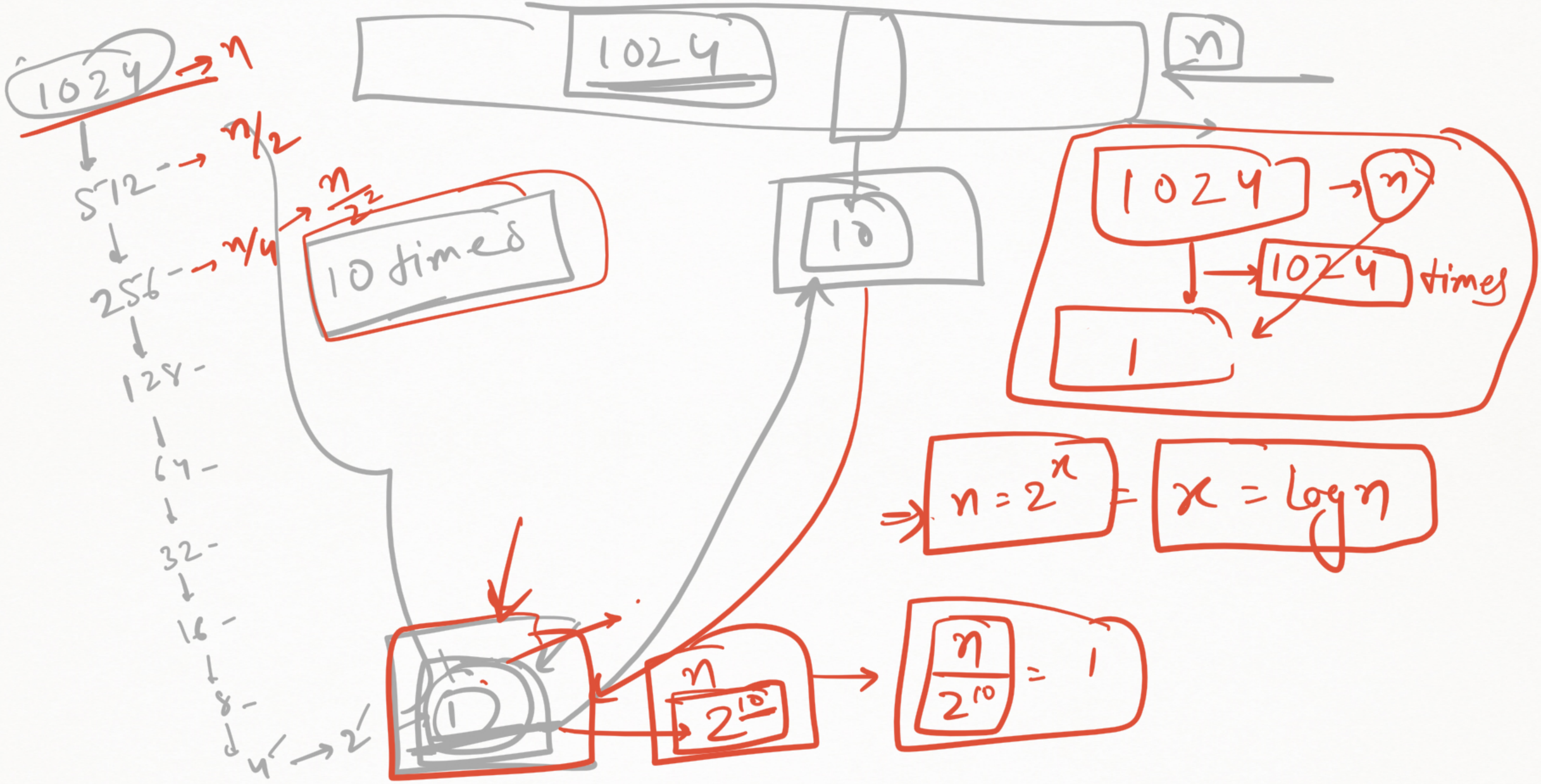
Binary Search



Recrence Relation

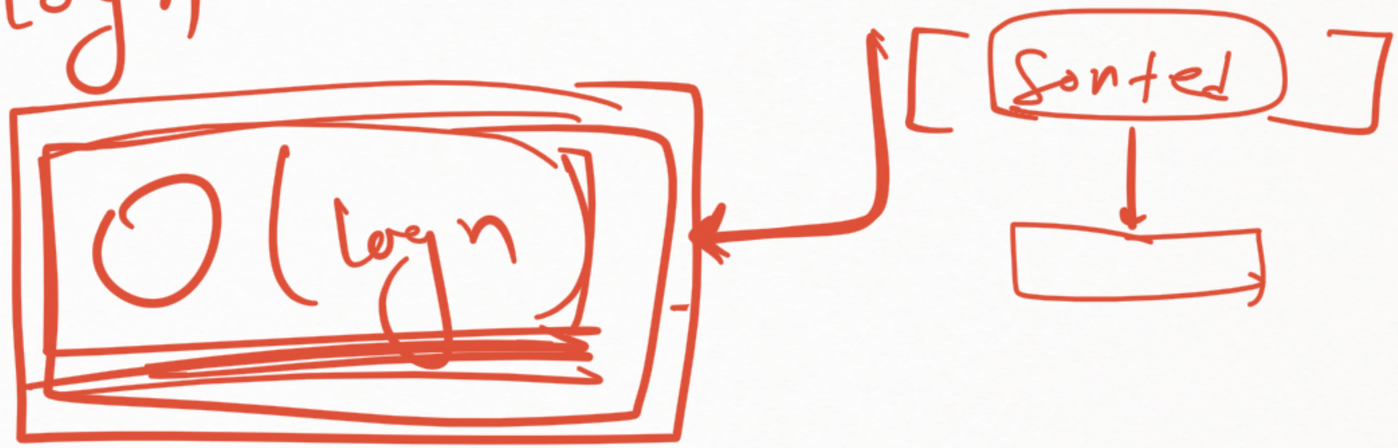
$$T(n) = k_1 +$$

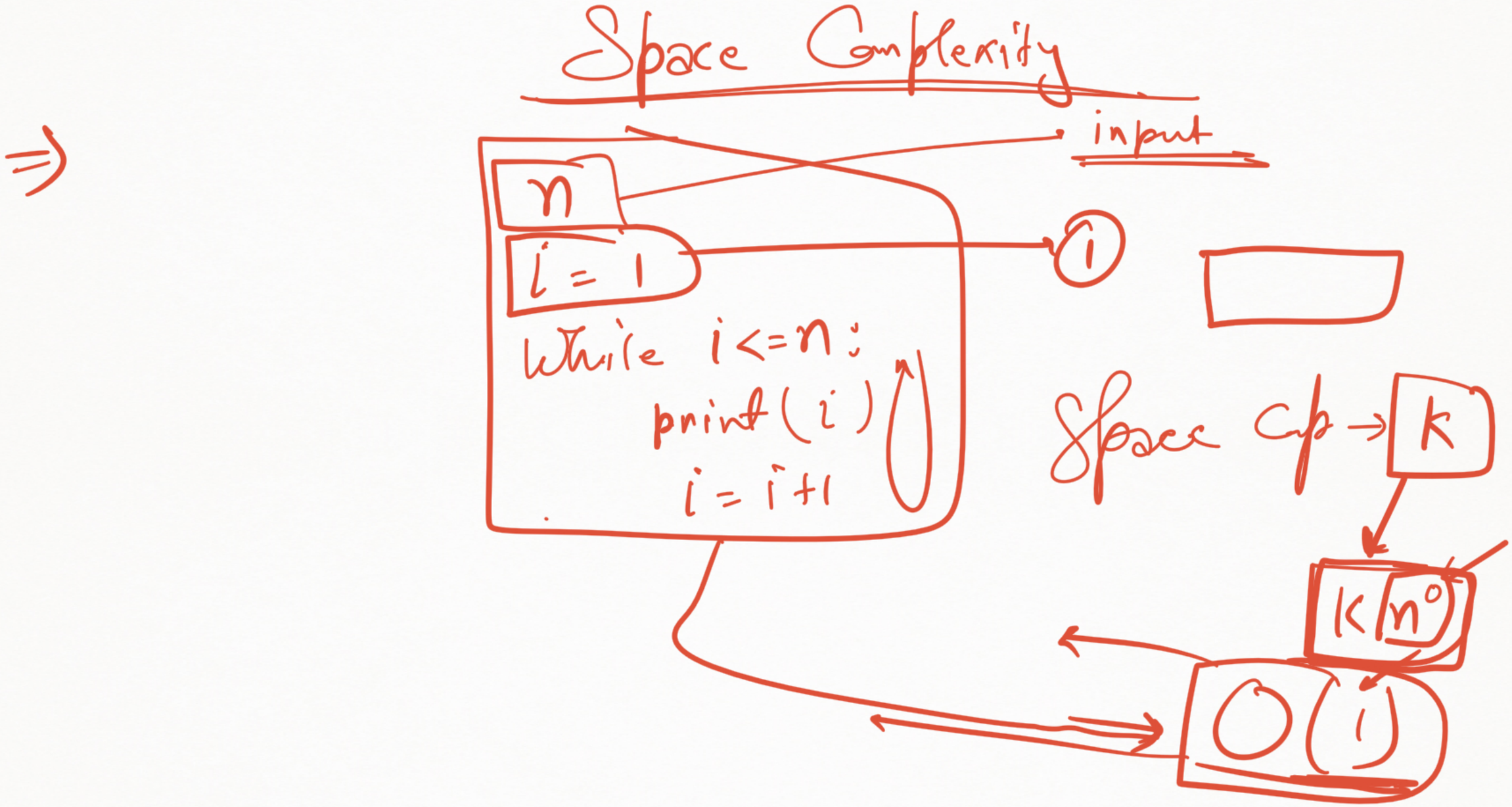

T(n) = k + T(n/2)
Soln
T(n) = T(n/2) + k
T(n/2) = T(n/4) + k
T(n/4) = T(n/8) + k
T(n/8) → 2^3
T(1) = k
x
n/2^x → 1
n
n/2
n/2^2
n/2^3
n/2^x
2^3
30-40
n = 2^x
x = log n
add
T(n) = k * x
Reach
m
s
-1

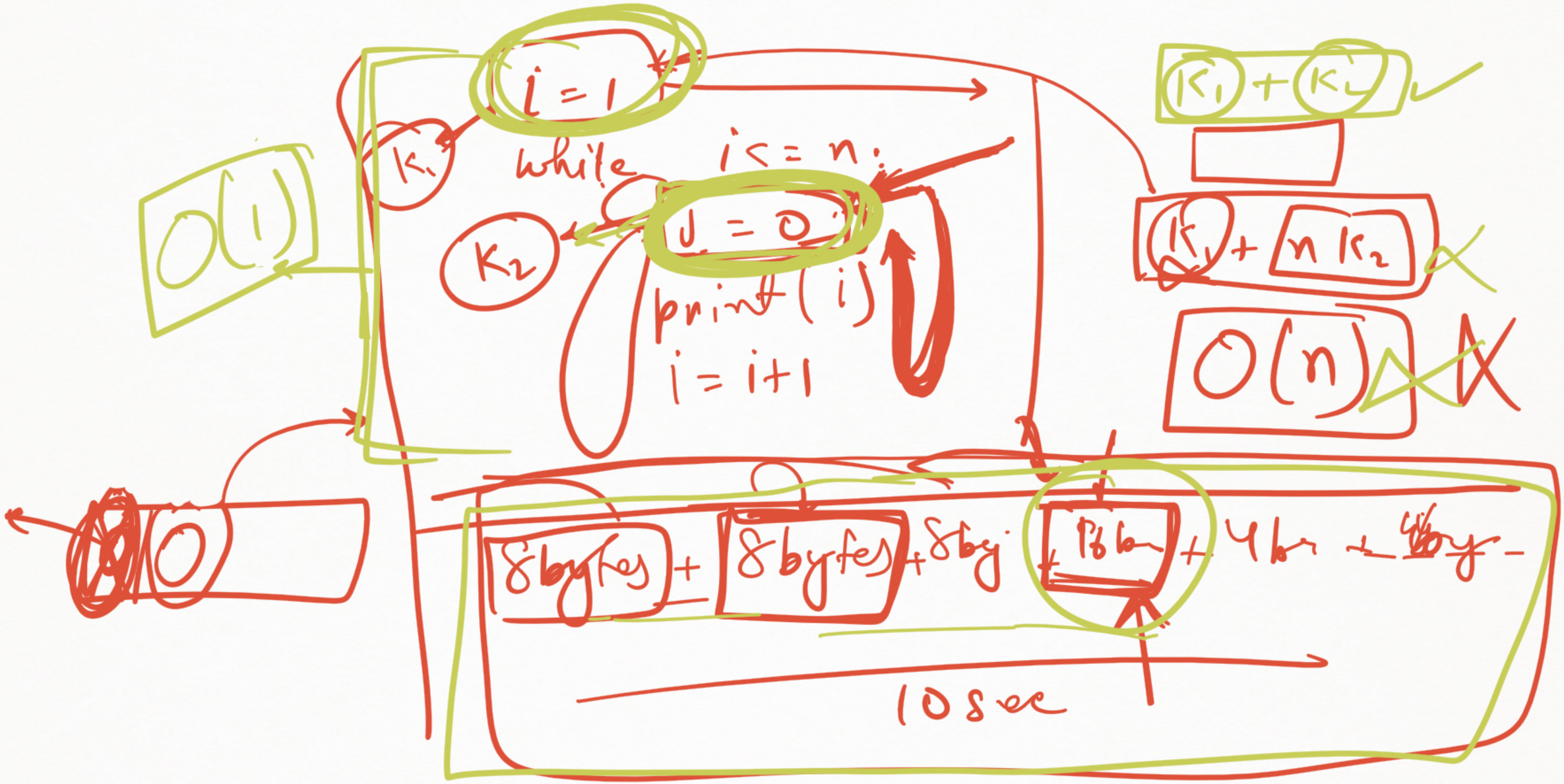

1024
n
512
n/2
256
n/4
128
64
32
16
8
4
2
1
n/2^2
10 times
1024
n
10
1024
n
1024
times
1
n = 2^x
x = log n
1
n/2^10
n/2^10 = 1

$T(n) = k * \log n$

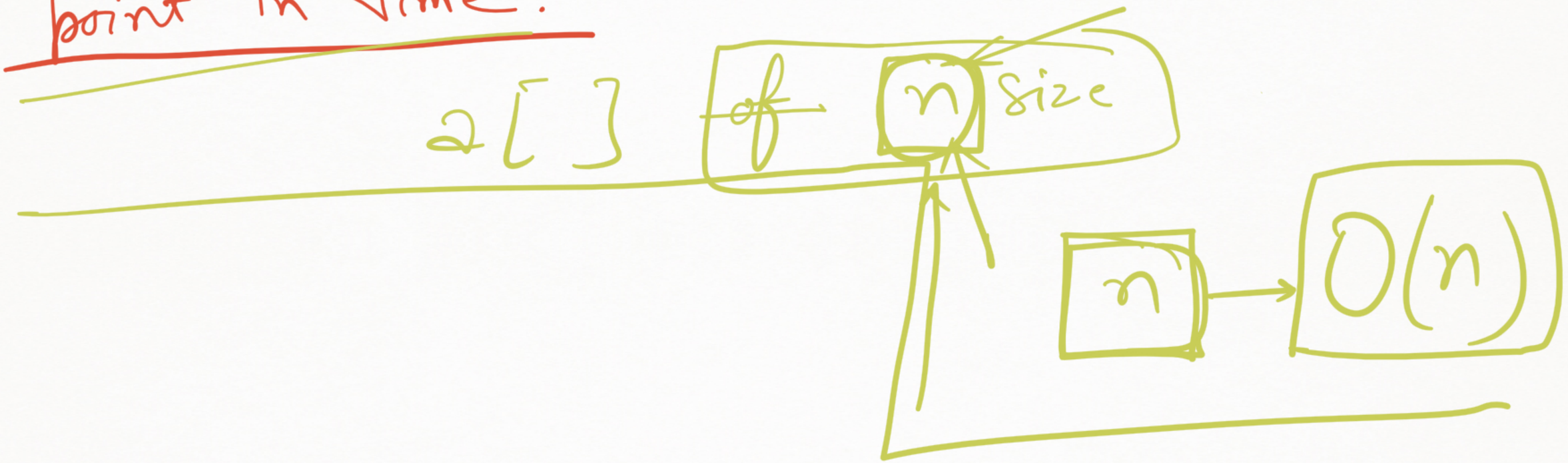
Binary Search → O(log n)




Space Complexity
input
n
i = 1
while i<=n:
print(i)
i = i+1
1
Space cp → k
k n°
1


i = 1
K1
while i <= n:
K2
j = 0
print(i)
i = i+1
O(1)
K1 + K2
K1 + n K2
O(n)
8byte
8byte
8byte
Pblock
4byte
10sec

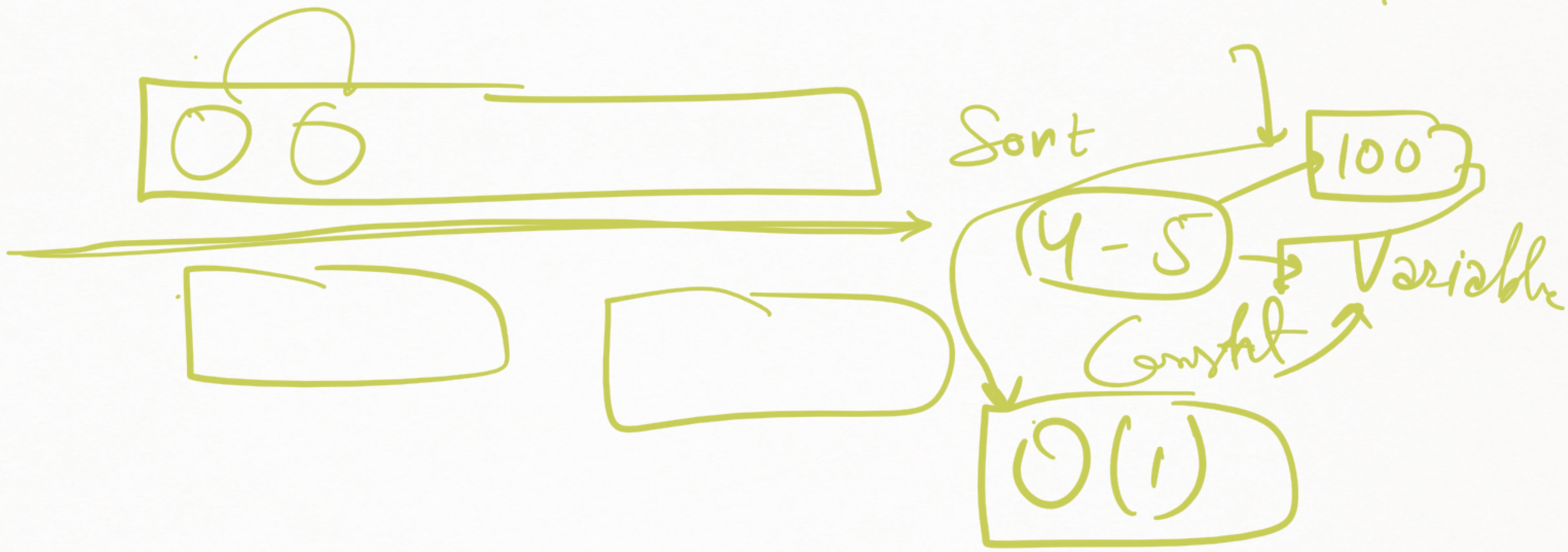
Space Complexity is maximum Space requirement at any point in time.

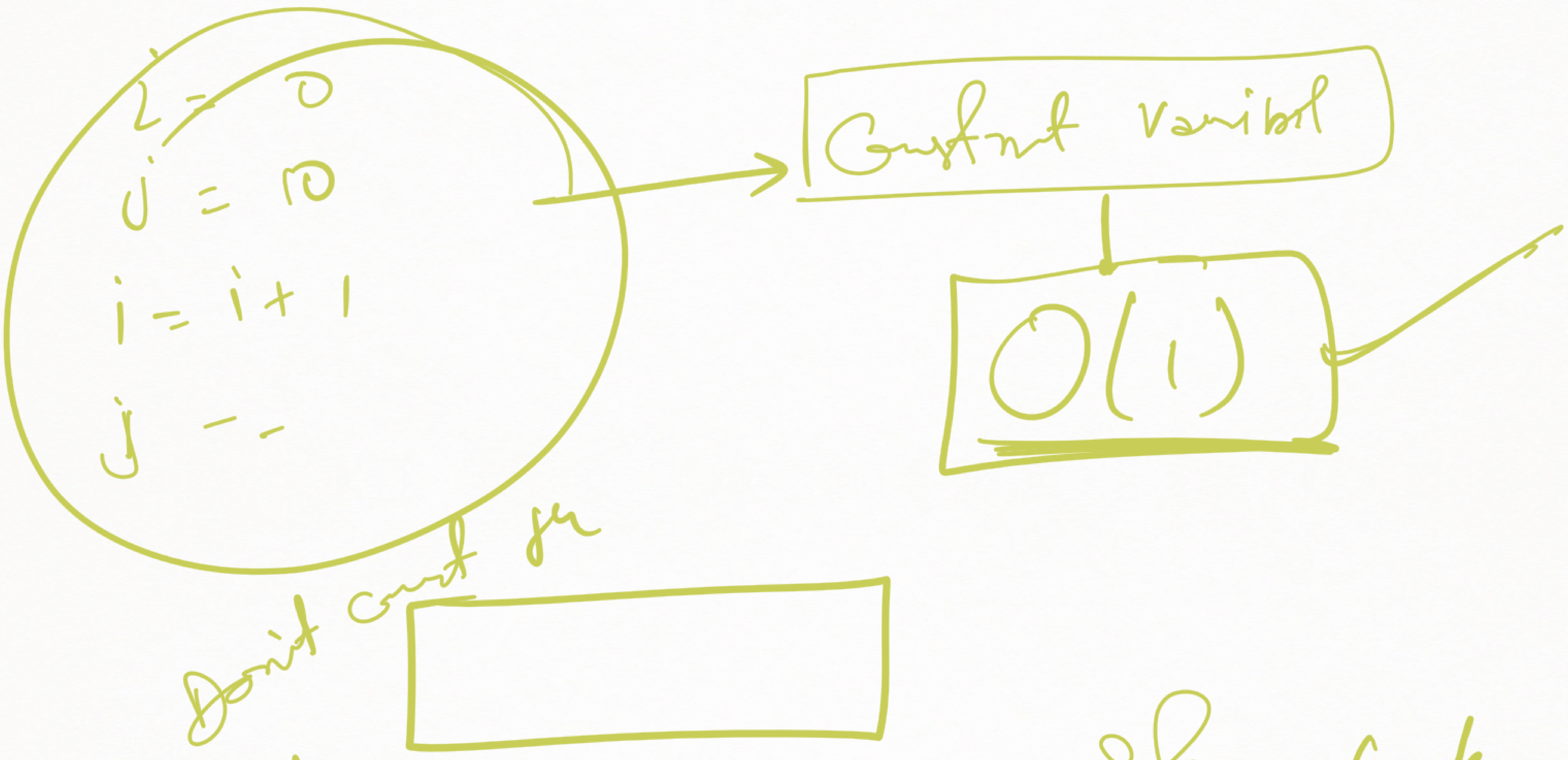


1) at any point in time

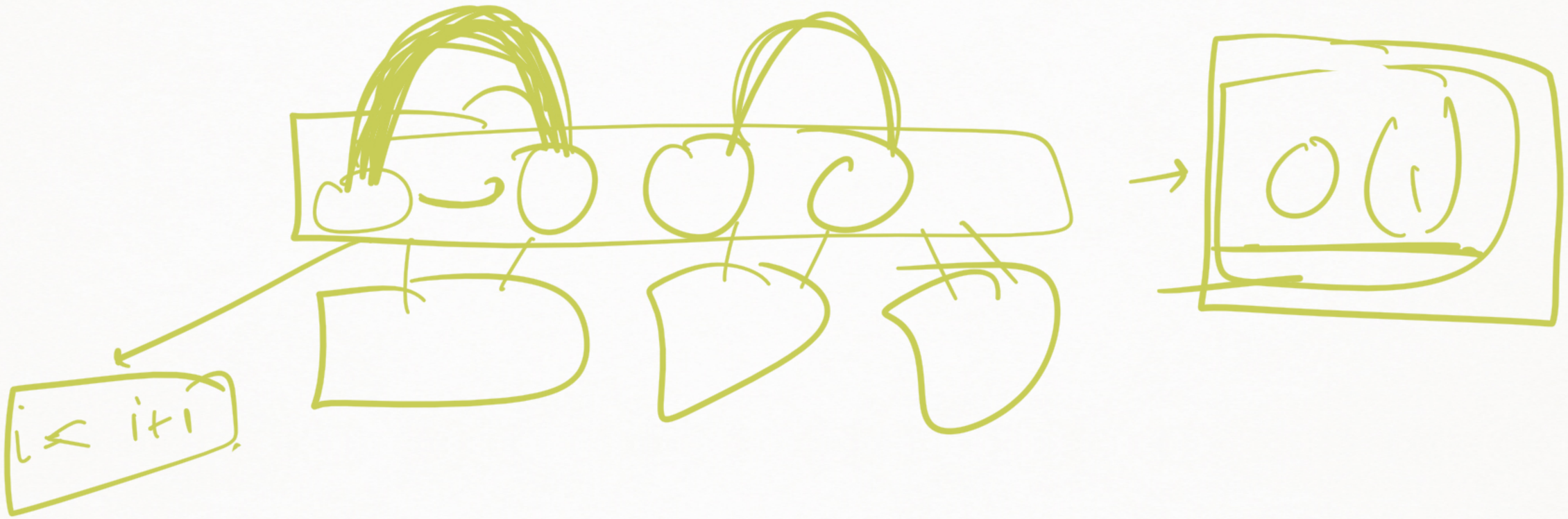
2) Auxiliary Space

Extra Space





input space req in your Space Comp.


l < i+1