

③ Given a recurrence relation, solve it using the recursive-tree approach:

a)  $T(n) = 2T(n-1) + 1$

$$\begin{array}{c}
 T(n) = T(n-1) + T(n-1) + 1 \\
 \swarrow \quad \searrow \\
 T(n-1) \quad T(n-1) \\
 \downarrow \quad \downarrow \\
 T(n-2) \quad T(n-2) \\
 \downarrow \quad \downarrow \\
 T(n-3) \quad T(n-3) \\
 \downarrow \quad \downarrow \\
 1 \quad 1 \\
 2^0 \quad 2^1 \quad 2^2
 \end{array}$$

K-term  $\rightarrow n - k = 0$  (Base case condition)

$$n - k = 0 \Rightarrow k = n$$

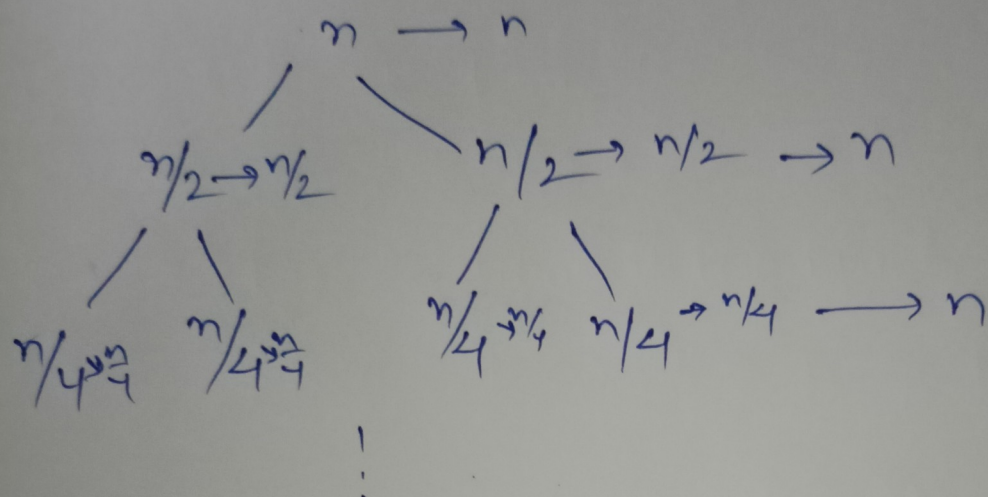
$$\begin{aligned}
 T(n) &= T(0) + 2^1 + 2^2 + \dots + 2^k \\
 &= 2^0 + 2^1 + 2^2 + \dots + 2^n \\
 &= 2^0 \left( \frac{2^{n+1} - 1}{2 - 1} \right) \\
 &= \frac{1}{1} (2^{n+1} - 1)
 \end{aligned}
 \left| \begin{array}{l} \text{A.G.P.} \\ a = 2^0 \\ r = 2 \\ S_n = a \frac{(r^n - 1)}{r - 1} \\ r > 1 \end{array} \right.$$

$\Rightarrow$  Time complexity will be  $O(2^n)$

Ans

$$(b) \quad T(n) = 2T(n/2) + n$$

$$T(n) = T(n/2) + T(n/2) + n$$



K times

$$\frac{n}{2^K} = 1 \Rightarrow n = 2^K \Rightarrow 2^K = n$$

$$K \log_2 2 = \log_2 n$$

$$\Rightarrow K = \log_2 n$$

$$T(n) = T(0) + n + n + n \dots \dots K \text{ times}$$

$$= 1 + nK$$

$\mathcal{O}(n \log_2 n)$  is the time complexity

Ans