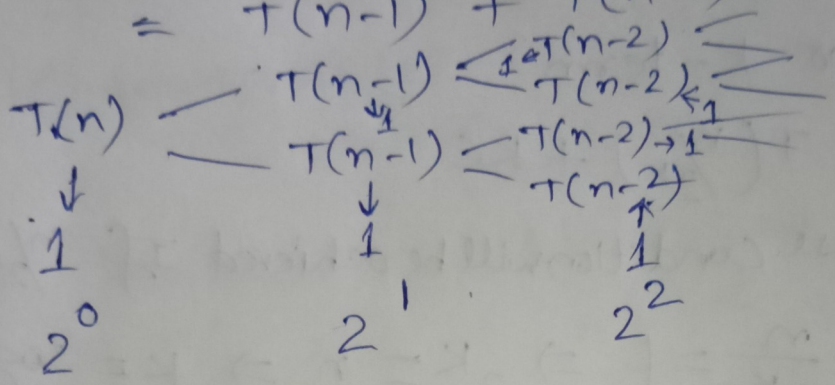
(3) Given a recurrence relation, solve it using the recursive-tree approach:

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

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



K-term → $n - K = 0$ (Base case condition)

$n = K \Rightarrow K = n$

$T(n) = T(0) + 2^1 + 2^2 + \cdots + 2^K$

$= 2^0 + 2^1 + 2^2 + \cdots + 2^n$

$= 2^0 \left(\frac{2^n - 1}{2-1}\right)$

$= \frac{1}{1}(2^n - 1)$

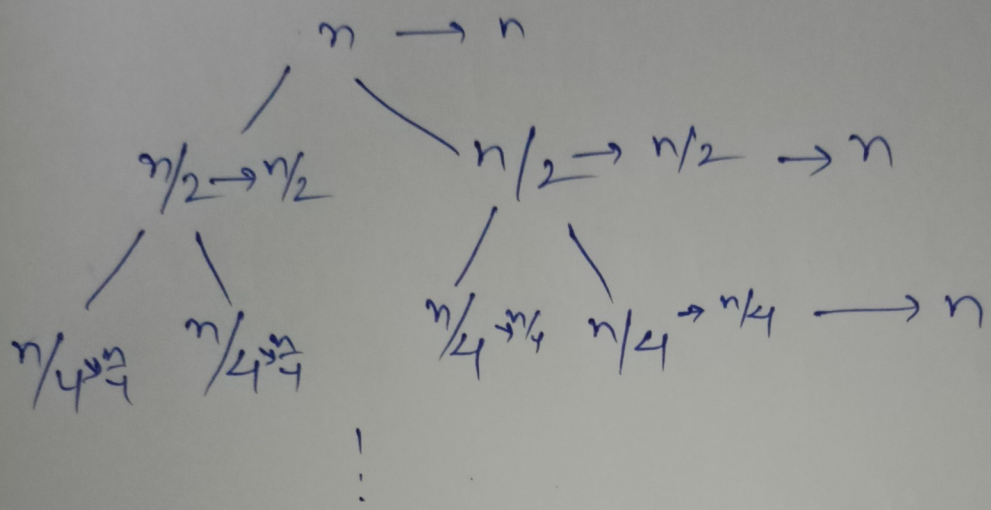
(A G.P.)
$a = 2^0$
$r = 2$
$S_n = \frac{a(r^n - 1)}{r-1}$
$r > 1$

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

Ans

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

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



K times

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

$K \log_2 2 = \log_2 n$

$\Rightarrow K = \log_2 n$

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

$= 1 + nK$

$O(n \log_2 n)$ is the time complexity

Ans