Repetated Substitution Methods for Binary Search

We know from recurrence Relation.

$$T(n) = T\left(\frac{n}{2}\right) + O(1)$$

Solving with repeated substitution method by the expanding form $T\left(\frac{n}{2}\right)$

Subtitute →

$$T(n) = \left[T\left(\frac{n}{4}\right) + O(1)\right] + O(1)$$

$$T(n) = T\left(\frac{n}{4}\right) + O(1) + O(1)$$

Continue substitution for k steps:

$$T(n) = T\left(\frac{n}{2^k}\right) + k \cdot O(1)$$

Where the process ends with $\frac{n}{2^k} = 1$ [Get single element]

$$\Rightarrow n = 2^k$$

$$\Rightarrow \log_2 n = \log_2 2^k$$

$$\Rightarrow k = \log_2 n$$

At this point

$$T(n) = T\left(\frac{n^1}{n}\right) + \log_2 n \cdot O(1)$$

$$T(n) = O(\log_2 n)$$

Time complexity of binary search.