* Problem statement :-

→ A tribonacci sequence $T_n = T_{n-1} + T_{n-2} + T_{n-3}$

→ $T_0 = 0$, $T_1 = 1$, $T_2 = 1$

→ for a given n, return T_n.

* Example :-

$n = 4$

$T_n = T_3 + T_2 + T_1$

$T_3 = T_2 + T_1 + T_0$ (We know the $T_0 = 0$ & $T_1, T_2 = 1$)

$T_3 = 1 + 1 + 0 = 2$

$\therefore T_4 = 2 + 1 + 1 = 4$

* Solution :-

1) Approach 1 :- Simple iterative approach

→ Since we know the values of T_0, T_1 & T_2, so whenever we encounter $n = 0, 1$ or 2, we will return their respective values.

→ For any value of n after 2, we start iterating from 3 to n & compute the next value by $T_{n-1} + T_{n-2} + T_{n-3}$.

→ We will keep on shuffling & adjust the values accordingly after each iteration.

TC :- $O(n)$

SC :- $O(1)$

2) Approach 2 :- Simple recursive approach

→ Base case :- $n = 0 \rightarrow 0$

$n = 1$ || $n = 2 \rightarrow 1$

return $foo(n-1) + foo(n-2) + foo(n-3)$

→ The problem with this approach is that we are re-calculating certain values multiple times.

→ Unnecessary call stacks will be created which increase the overall time complexity.

→ So, on leetcode, we will get TLE with this approach.

Note :- This problem can be solved in a much efficient way using DP. Come back after learning DP.