We have given two implementation code of for finding n-th Fibonacci number.

The condition for fibonacci isa: fibonacci_1(n-1) + fibonacci_1(n-2) The way the tree representation works is that

stored or not. As a res fibonacci_1(n) to phelymon anit with

fibonacci_1(n-1) fibonacci_1 (n-2).

50, the implementation 2 is better fibmacci_1(n-3)

fibonacci(n-4).

fibmacei_1 (n-3)

It is visible that, we start from a given number and goes until we reach at the base case. It In this way, many functions gets recalled more than once in implemen--tation 1, which results as in taking a huge amount of space. As a result, the time complexity for implementation! is $O(2^n)$,

On the order other hand, in implementation 2, we store the outputs in fib list, and when we are about to store another value result, we check whether we already have this value stored or not. As a result, the time complexity of implementation - 2 is O(n) We know, time complexity O(n) (10(2n)) and 1 50, the implementation-2 is better fibonacci_1(n-3) time complexity. fibracei_1 (n-3) It is visible that We start from a given number and goes until we reach at the bane case: A. In this may, mony functions gets recalled more than once in implemen. -lation 1, which results as in taking a luge amount of space. As a result the time complexity for implementations [15 (2n).