Ans to the question no. 2

For the implementation 1, the function is using recursion.

So, let,

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

$$= 4T(n-2) + 3c$$

$$= 8T(n-3) + 7c$$

$$\vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots$$

$$= 2^k T(n-k) + (2^k - 1)c \quad \ldots (i)$$

Now, $n - k = 0$

$\Rightarrow n = k \quad \ldots$ (ii)

$\Rightarrow k = n \quad \ldots$ (ii)

Putting Eq. (ii) k value in Eq. (i).

$$T(n) = 2^n T(0) + (2^n - 1)c$$

$$= 2^n(1 + c) - c \approx 2^n$$

So, For implementation 1 using recursion, the time complexity is ~~O(n~~ $O(2^n)$. So, This function is an exponential function and have an a bad time complexity.

The implementation 2 is using simple loop / for loop and memorization method. Not any recursive method. ~~Therefor~~ Also, implementation 2 does not have any nested loop. Therefore, the time complexity for this function implementation ~~wou~~ would be O(n). So, implementation 2 is an linear time algorithm function.

Here, Implementation 2 has better time complexity than implementation 1. Therefore, For larger inputs implemention 1 would require more time than implementation 1 to solve the problem.