4. Seja a fórmula de recorrência abaixo:
$T(n) = 1$, se $n = 1$
$T(\lceil n\ 2 \rceil) + 1$, se $n > 1$
Pelo método da substituição mostre que $T(n) = O(\lg n)$

hipótese
$T(k) \leq c * \lg(k)$ para todo $k < n$

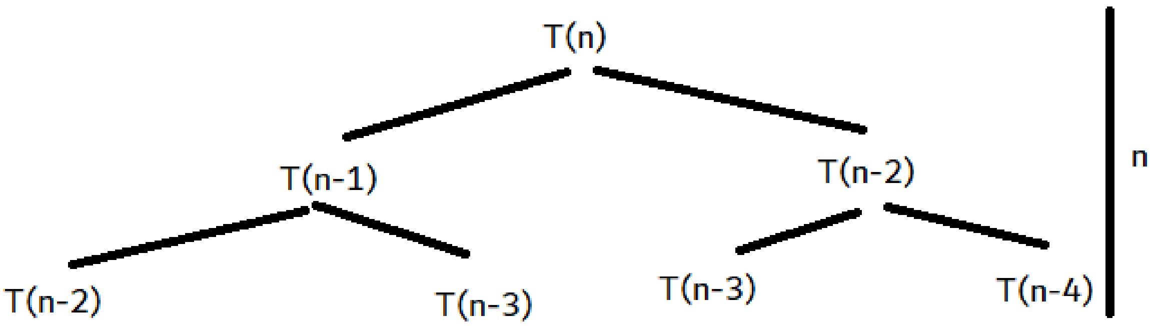
substituir $T(n/2)$ na fórmula de recorrência:
$T(n) = T(\lceil n/2 \rceil) + 1$
Usando a hipótese de indução, temos:
$T(n) \leq c * \lg(\lceil n/2 \rceil) + 1$
$T(n) \leq c * \lg(n) + 1$

escolher $c \geq 1$, de modo que $c * \lg(n) \geq \lg(n)$ para todo $n > 1$

$\therefore T(n) \leq c * \lg(n) + 1 \leq c * \lg(n) + \lg(n) = (c + 1) * \lg(n)$
Se $C = c + 1$ e $n_0 = 1$, então $T(n) = O(\lg n)$
$T(n) \leq (c + 1) * \lg(n) = C * \lg(n)$ para todo $n \geq n_0$.

$\therefore T(n) = O(\lg n)$

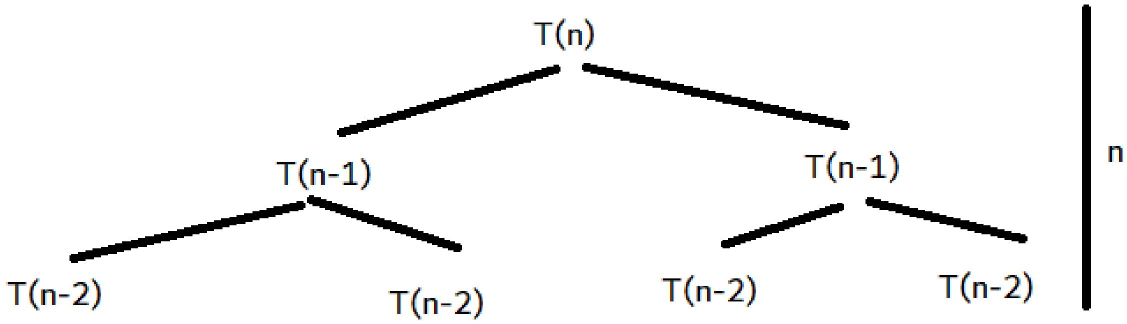
5. Determine a complexidade de tempo do seguinte trecho de código, utilize a árvore de recorrência.
a)
```
int fibonacci(int n)
{
if (n == 0 || n == 1) return n;
else return fibonacci(n-1) + fibonacci(n-2);
}
```



$\therefore T(n) = O(2^n)$

b)
```
void hanoi(int n, char origem, char destino, char auxiliar)
{
        if (n > 0)
        {
        hanoi(n - 1, origem, auxiliar, destino);
```

```
            printf("Movimento: %c -> %c", origem, destino);
            hanoi(n - 1, auxiliar, destino, origem);
            }
}
```



$\therefore T(n) = O(2^n)$