Design and ANLY ALGORITHMS

Hands-on-4

K. Abhishek
1002281717

Problem - 0

Debug the code & step into function for fib(5). Here we want to step into all recursive calls & list out the function call stack ex. fib(5) -> fib(4) -> fib(3)?

fib(5) -> fib(4) -> fib(3) -> fib(2) -> fib(1)
-> fib(0)
-> fib(1)
-> fib(3) -> fib(2) -> fib(1) -> fib(0)
-> fib(1)

And here is the steps breakdown for recursive calls:

↳ fib(5) calls fib(4) and fib(3)
↳ fib(4) calls fib(3) and fib(2)
↳ fib(3) calls fib(2) and fib(1)
↳ fib(2) calls fib(1) and fib(0)
↳ fib(1) returns 1
↳ fib(0) returns 0.

This process continues until all recursive calls are resolved.

* Prove the time complexity of algorithms

The time complexity of this recursive Fibonacci implementation. is.

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

$$T(n) \leq \Theta(2^n)$$

* Commenting on my ways that could improve your implementation.

↳ [redacted] store results of previously computed Fibonacci numbers to avoid redundant calculations. This will reduce time complexity to $\Theta(n)$.

↳ Use an approach to compute Fibonacci numbers, will also have time complexity of $\Theta(n)$.

↳ By using matrix exponentiation to compute Fibonacci numbers in $\Theta(\log n)$.