COMPUTER SCIENCE

QUESTION 3

If we pass a number that is greater than 0 and 1. Then we make two recursive calls where we add both calls with the nth-Number minus 1 and 2 in both calls. Passing these Integers 0, 1, 2, 3, 4, 5, Will most likely give us 0, 1, 1, 2, 3, 5 in a timely fashion. But if we pass higher numbers like 50, 67, 100. You will begin to notice how much longer it takes for this method gives us our Fibonacci number. The reason for the poor performance is heavy push-pop of the stack memory in each recursive call and the reason for this is when you call a Function against itself (as recursion) the compiler allocates new activation record for that new function. That stack is used to keep your states, variables and addresses. Compiler creates a stack for each function and this creation process continues until the base case reached. So, when the data size becomes larger, the compiler needs large stack segment to calculate the whole process. Thus, causes stack overflow.