**Why is it a bad idea to use recursion method to find the Fibonacci of a number?**

To begin, I’ be defining what Fibonacci means.

* It’s usually associated with the word “sequence”. Thus, Fibonacci-sequence is a sequence such that each number of the sequence is the sum of the two preceding numbers, and the sequence starts from 0 and 1 for example, 0, 1, 1, 2, 3, 5, 8...

Below is a sample java code that uses recursion method to find Fibonacci number

***//Recursive method***

***static int fibRecursion(int n) {***

***if ((n == 1) || (n == 0)) {***

***return n;***

***}***

***return fibRecursion(n - 1) + fibRecursion(n - 2);***

***}***

***}***

The disadvantage of using this method comes to play when the number of ‘n’ increases as it causes heavy push-pop of the stack memory in each recursive call.

***NB:*** *The ‘iteration’ method is an alternate way to make the process faster as the value of n increase (to say, 100).*

//Iteration method

**static** **int** fibIteration(**int** n) {

**int** x = 0, y = 1, z = 1;

**for** (**int** i = 0; i < n; i++) {

x = y;

y = z;

z = x + y;

}

**return** x;

}