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

Recursion by definition is “when a thing is defined in terms of itself.” In this case we are referring to mathematical or programmatic functions. With respect to a programming function, recursion happens when a function calls itself within its own definition. It calls itself over and over again until a base condition is met that breaks the loop.

There are 2 main parts of a recursive function; the base case and the recursive call. The base case is important because without it, the function would theoretically repeat forever (in application there would be what is referred to as a “stack overflow” to stop the repetition which we will touch on a little later)

**The following are the reasons why it is a bad idea to use recursion method to find the Fibonacci of a number:**

**Recursion uses more memory.** Because the function has to add to the stack with each recursive call and keep the values there until the call is finished, the memory allocation is greater than that of an iterative function.

**Recursion can be slow.**If not implemented correctly (as stated above with memorization) it can be much slower than iteration. It is actually pretty difficult to write a recursive function where the speed and memory will be less than that of an iterative function completing the same task. The reason that recursion is slow is that it requires the allocation of a new stack frame.