RESEARCH ASSISTANT - CRYPTO

Solution by **Favour Ohanekwu** (fohanekwu@gmail.com). September 30, 2020.

Computer Science Section

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

Solution

Mathematically, the Fibonacci of a number n expressed as F_n is given by

$$F_n = F_{n-1} + F_{n-2}$$

with seed values $F_0 = 0$ and $F_1 = 1$

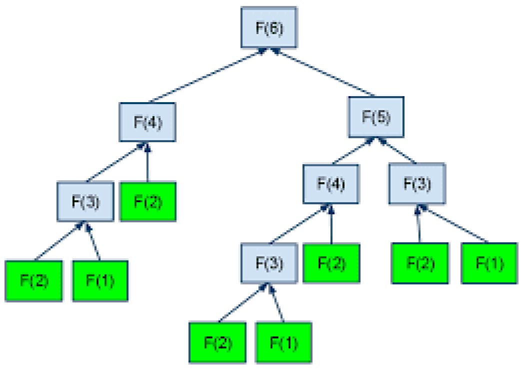
The recursive algorithm for fibonacci numbers is given below

```
For a non-negative number n,
Procedure Fibonacci(n)
If n = 0, return 0
else if n = 1, then return 1
else return  Fibonacci(n-1) + Fibonacci(n-2)
```

Limitations of the Recursion Method

- This implementation does a lot of repeated work thereby causing heavy push-pop of the stack memory in each recursive call.

Take for example the Fibonacci of 6, F_6 shown below



In this implementation, $\mathbf{F_1}$ appears to be computed three (3) times, $\mathbf{F_2}$ computed five (5) times, $\mathbf{F_3}$ computed three (3) times, $\mathbf{F_4}$ twice and $\mathbf{F_5}$ once**.**

This redundancy and repetition leads to very long running time and makes this implementation computationally expensive.

- The amount of memory used for parameter and local variables on the system stack can be very substantial.

- The time complexity $\mathbf{T_n}$ is given by:

$$T(n) = T(n-1) + T(n-2) + C$$
$$T(n) = O(2^{n-1}) + O(2^{n-2}) + O(1)$$

On solving the above recursive equation we get the upper bound of Fibonacci as $\mathbf{O(2^n)}$, however the tight upper bound is $\mathbf{O(1.6180)^n}$. As the value of n increases this grows exponentially.

However, the recursion method can be improved by memoization (an optimization technique used primarily to speed up computer programs by storing the results of expensive function calls and returning the cached result when the same inputs occur again).