365 DataScience Recursion in Python - the Fibonacci sequence

Step 1 Create a recursive function

```
# Create a function that would use recursion to calculate the n-th Fibo
nacci number.
# Recursion is not very efficient since each call of the function asks
the computer for some memory.
# Therefore, calculating big Fibonacci numbers would fill up the memory
 very fast.
# The iterative method (for-loops, for example) is often significantly
faster.

# In this problem, the Fibonacci sequence starts at 0 and the indexing
also starts at 0.
def fib(n):

    # All recursive functions need a base case. In this problem, we hav
e two of them - checking whether n is 0 or 1.
    # If n equals either of these numbers, the function will stop calli
ng itself and start computing the
    # n-th Fibonacci number.
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        # When n = 0 (fib(0) = 0) or n = 1 (fib(1) = 1), we cannot take
 the sum of the two previous numbers because
        # fib(0) and fib(1) *are* the first two numbers. However, we ca
n calculate the numbers that sit at positions
        # n >= 2. In the case of n = 2, we will take the sum of the num
bers at n = 0 and n = 1,
        # which are 0 and 1, as we have defined in the base cases above.
        return fib(n-1) + fib(n-2)
```

Step 2 Call the function

```
# Return the Fibonacci number with index 20.
fib(20)
```

Start your 365 Journey!