=====================================================================

Template for writing proof of correctness for recursive algorithms

=====================================================================

**STEP 1**: State the precondition and post-condition

Pre-condition: ....

Post-condition: ....

**STEP 2**: STATE WHAT YOU MUST PROVE (i.e. precondition -> termination + post-condition being met upon termination)

We must prove that if (precondition) is true for some (input), then this call terminates and upon termination, returns (post-condition).

**STEP 3**: PROVE FOR EACH PROGRAM PATH

For each path, show:

(A) it terminates

(B) post-condition is satisfied upon termination

\* If there is a recursive call on the path:

(1) Argue why preconditions are satisfied for recursive call

(2) The recursive call occurs on a smaller input than the original call

\* STEP 3.5 FIGURE OUT A VARIABLE TO REPRESENT THE INPUT SIZE (this variable must be something that gets smaller upon each recursive call)

I.H.: If (1) and (2) are true for the recursive call then you may assume the recursive call terminates, and that post-condition are met when the recursive call terminates.

Finally, argue from the last recursive call to the end of the function why post-condition of the original function call will hold.

-----------------------------------------------------------------------

PROOF FOR FACT\_REC(N) FROM LECTURE SLIDES

-----------------------------------------------------------------------

**STEP 1**: State the precondition and post-condition

Precondition: n is an integer >= 0

post-condition: returns the factorial of n

**STEP 2**: STATE WHAT YOU MUST PROVE (i.e. precondition -> termination + post-condition being met upon termination)

We must prove that if the precondition (n being an integer >= 0) is true for some n passed into fact\_rec(n), then this call terminates and upon termination, returns the factorial of n.

**STEP 3**: PROVE FOR EACH PROGRAM PATH

Path 1: We go into this path when n = 0, or n = 1.

When n = 0 and n = 1 we must return 1, since 0! = 1! = 1.

And this is what the function returns, so this path works correctly.

**Path 2**: We go into this path when n >= 2.

We have a recursive call fact\_rec(n-1)

**\* STEP 3.5** FIGURE OUT A VARIABLE TO REPRESENT THE INPUT SIZE (this variable must be something that gets smaller upon each recursive call)

Let's take n to be my input size.

Then, since n-1 < n, I know that my input size for each recursive call is decreasing.

Why is precondition satisfied when input = n-1?

1) n-1 is still an integer if n is an integer

2) We know that we only enter this path if n >= 2, therefore n-1 >= 0

Therefore, precondition holds.

So, by I.H. I can assume that fact\_rec(n-1) terminates and returns (n-1)! as the post-condition states.

Now, we can see that the final call returns n \* fact\_rec(n-1) = n \* (n-1)! = n! by definition.

Therefore, this function terminates and returns n! which satisfies the post-condition in this path.

So, the function is correct.