# Week 11: Verifying correctness of recursive programs

1)

Proof by induction

Our pre-condition ensures that n $\geq 1$, which makes sense because we can't take the factorial of negative numbers, that includes zero.

First, we test the base case. If n = 1 then the function returns 1, indeed we can confirm that the factorial of 1 is 1. So far, the program is correct for this base case.

Now we need to test the inductive step. In this case n > 1. For that, suppose n = 2, which is greater than 1, that's why the function will return 2 * fact (2 - 1) => 2 * fact (1). We have already proved that fact (1) = 1, so 2 * fact (1) => 2 * 1 = 2, which is correct.

Now suppose fact ( ) has been given some integer t > 1 as input. In this case, the function will return t * fact (t - 1) => fact (t) = t * fact (t - 1) , for t > 1. We now know that the correctness of fact (t) depends on the correctness of fact (t - 1). This means that fact (t) is correct if fact (t - 1) is correct. Suppose we now pick some specific integer f.eks. t = 4. We then know that fact (4) is correct if fact (3) is correct, further that fact (3) is correct if fact (2) is correct. We already proved fact (2) to be correct. Therefore fact (3) is correct, and further that fact (4) is correct.

In general, no matter which integer there has been chosen for t, we could create a finite if-then propositions of the form: if fact (t - 1) is correct then fact (t) is correct. We will then eventually arrive at the proposition: if fact (1) is correct then fact (2) is correct.

The base case fact (1) proves the case for fact (2) etc. all the way up to sum (t). This proves that our program is correct for all positive integers.