I'm struggling with the space complexity of the function in #37.

So tail recursion takes up constant space, and I believe the recursive calls in this function are iterative processes (correct me if I'm wrong). But my question is, does an iterative process = tail recursion? And does Scam utilize tail recursion?

Tail recursion takes up constant space on the stack. Scam does use tail recursion.

I believe a function can be tail recursive and non-iterative.

If a function is purely tail recursive, it generates an iterative process. If it is both tail and non-tail recursive, then it *may* not be iterative.

Okay thanks! That also leads to another question. A couple of the problems contain both tail and non-tail recursion, so how would one determine whether the function is overall iterative or recursive? I was thinking it had something to do with how the conditionals were set up, or maybe how one type of recursion might be more dominant in the function?

You are correct. If, by examining the conditional, you can bound the number of non-tail recursive calls to a constant, then you get an iterative process.