The first thing we had to implement for this new iteration of the final assignment was the IF control structure. Since both the **'If'** and **'If/Else'** statements share a very similar structure (in C and in Lisp) to the **'While'**, which we managed to complete in the previous session, we only needed to add two derivations to our **control_sentence** non-terminal.

The next thing we had to do was the FOR control structure. This structure is also a new derivation of the **control_sentence** non-terminal, since it shares a very common usage and structure to the previously mentioned functionalities. The main difficulty lies when determining which kind of non-terminal the different expressions inside the for statement would be. We determined that the 'initialization' (where a value is assigned to the new variable) would be a **axiom_sentence**, the 'expr' (stopping condition for the loop) would be an expression non-terminal (the LISP code would only work if the user creating the C code uses an expression resulting in a boolean, but we assume that will always be the case) and the 'inc/dec' (which is responsible for how the variable changes throughout the loop) would be a **IDENTIF '=' expression**, since we actually need it to be an assignment operation. For translating the definitions will be done before the actual loop, the stopping criterion will be used as the while condition and the increment will go at the end of the loop, recreating the original functioning of the for loop.

The last thing that we did was adding the functionality of recognising the creation of functions (with multiple arguments) and translating it into Lisp. Creating a function was relatively simple, we only needed to add a new derivation of function_rec. But for passing multiple arguments (or none) we had to create 4 new non-terminals, two for defining them and two for when they were already defined and you need to call the function. The other two non-terminals were used to add recursion to the previously mentioned ones.