1. What is the syntax for declaring a function in your language?
    a. void *myFunction*() {
    }
    Declare the function name and then add a ()
    https://www.w3schools.com/cpp/cpp_functions.asp

2. Are there any rules about where the function has to be placed in your code file so that it can run?
    a. A function has to be defined before it is called.

3. Does your language support recursive functions?
    a. Yes, C++ does support recursive functions. You can find a simple recursive factorial function here:
    https://www.geeksforgeeks.org/cpp-program-to-find-factorial-using-recursion/#

4. Can functions in your language accept multiple parameters? Can they be of different data types?
    a. Yes, c++ does accept multiple parameters and different data types. Since you can declare each parameter independently, you don't need to use the same data type for all of them.
    https://www.udacity.com/blog/2021/09/developers-guide-to-cpp-function-parameters.html#:~:text=As%20with%20the%20function%20itself,same%20data%20type%20for%20all.

5. Can functions in your language return multiple values at the same time? How is that implemented? If not, are there ways around that problem? What are they?
    a. C++ does not return multiple values like Python does. You can use pointers, structures, or arrays instead to get around the problem.
    https://www.geeksforgeeks.org/how-to-return-multiple-values-from-a-function-in-c-or-cpp/

6. Is your language pass-by reference or value? Check your code against outside sources in case there is anything tricky going on (like in Perl).
    a. C++ supports both pass-by-value and pass-by-reference. When you pass by reference, the variable is not copied, so the original variable can be modified vs when in pass-by-value a copy of the argument is passed on to the function.

7. Where are the arguments, parameters and local variables stored (value-on-stack, ref-to-heap-on stack) during execution?

a. Local variables are typically stored on the stack. Global variables using malloc are stored on the heap.

8. What are the scoping rules in your language (visibility and lifetime of variables before, during and after code blocks)?
    a. Local variables do not exist outside the block in which they are declared, i.e. they can not be accessed or used outside that block. Local variables are anything in {}
    b. Global variables are accessible in any part of the program and they are available throughout the lifetime of a program.
    c. https://www.geeksforgeeks.org/scope-of-variables-in-c/#

9. Are side-effects possible? Are there guard rails against side-effects?
    a. According to this University of London source, a "function that does something else apart from just returning its value, such as taking some input or generating some output or changing the values of variables (other than its own local ones) is said to have a side-effect."
    - Examples: getline(getline(cin,s1), s2);

10. Where are local variable values stored? (on the stack? On the heap?)
    a. Local variables are stored on the stack.

11. Are there any other aspects of functions in your language that aren't specifically asked about here, but that are important to know in order to write one? What are they? (e.g. dynamic vs static scope)
    a. C++ is statically typed, meaning variables are bound to a data type during compilation. Also, meaning the programmer has to declare the type of the variable when it is originally declared.
    b. Static scope