Function Parameters: Summary

This lesson summarizes the how parameters can be passed to functions.

WE'LL COVER THE FOLLOWING

Summary

- A parameter is what the function takes from its calling function to accomplish its task.
- An argument is an expression (e.g. a variable) that is passed to a function as a parameter from the calling function.
- Every argument is passed by copy.
    - However, for reference types, it is the reference that is copied, not the original variable.
- `in` specifies that the parameter is used only for data input.
- `out` specifies that the parameter is used only for data output.
- `ref` specifies that the parameter is used for data input and data output.
- `auto ref` is used in templates only. It specifies that if the argument is an `lvalue`, then a reference to it is passed; if the argument is an `rvalue`, then it is passed by copy.
- `const` guarantees that the parameter is not modified inside the function.
    - Remember that `const` is transitive: any data reached through a `const` variable is `const` as well.
- `immutable` requires the argument to be immutable.

- `inout` appears both at the parameter and the return type and transfers the mutability of the parameter to the return type.
- `lazy` is used to make a parameter be evaluated when (and every time) it is actually used.
- `scope` guarantees that no reference to the parameter will be leaked from the function.
- `shared` requires the parameter to be shared.
- `return` on a parameter requires the parameter to live longer than the returned reference.

In the next lesson, you will find a coding challenge related to function parameters.