Given a task to rewrite and improve dose.c in terms of compactness, robustness (with the amount of duplicated code as a key aspect for both), readability and (your personally preferred) style, as well as implement relevant unit testing, please include the following in your submission:

1. A description of the characteristics and aspects of dose.c that you consider problematic in terms of code maintainability, error-proneness (regardless of how short or simple the considered code technically is), testing, readability, and lint.

Firstly, to facilitate the unit testing paradigm, especially using the diy_compare.c file, I had to keep to a consistent static declaration for all of the functions to avoid conflicting/link errors.

Secondly, to improve the maintainability of the the calcIED and calcMGD functions, new functions were declared to handle the arithmetic operations for e.g `calcMgdAnode and calcIedAnode.`

Subsequently, the ternary operator was strategically used in specific areas of the code to add more simplicity to some if statement declaration in some parts of the code.

With regards to error proneness, I will handle if I where the developer the rare occasions where the thickness is out of range. Currently there is not such implementation.

The code does not handle the inclusion of header files properly.

It's also important for the sake of maintenance, to highlight the difference between the arithmetic operation for both the `mgd and ied.` To ensure subsequent refactoring do not mistake the two as a repetition.

2. The rewritten file dose.c that compiles and replaces the original file as such (dependencies unchanged).

You can find in the github repository

3. A description of relevant unit tests for ensuring the correct function of calcIED and calcMGD. Writing the tests isn't required, but a boilerplate implementation/draft of the tests (using a DIY or an established testing framework such as googletest) is considered a bonus. In this case please also consider its generalizability assuming that the file dose.c represents a typical example in terms of variable and function scopes etc.

I implemented both a google test boilplate and DIY test that works for reference.

4. Any comments/observations about any peculiarities in the dose calculation protocol.

As I mentioned it easy to mistake the two calculation protocols both the `mgd and ied.` As they closely match the other except that in the case of mgd calculation there is a subtraction to watch out for and it also substracts the current value from the next. Which is different in the ied calculation which adds and rather substract the next value from current value.

For reference I am talking about this functions.

```
mgd = MGD_W_Ag[p][kv-20] - ((interpol * (MGD_W_Ag[p][kv-20] -
MGD_W_Ag[p+1][kv-20])+500)/1000);

ied = IED_mag16_Mo[p][kv-20] + ((interpol * (IED_mag16_Mo[p+1][kv-20] -
IED_mag16_Mo[p][kv-20])+500)/1000);
```