Debugging Exercises

This exercises is intended to improve your debugging and refactoring skills. On Canvas you will find the Debugging Exercise Project that contains a Visual Studio solution that currently doesn't compile and may have a series of issues in it.

The intention of the program is to implement a skirmish between Marines and alien Zerglings, using the following game mechanics:

- The Marines attack first, each one shooting once at the first alive Zergling from a list.
- The Zerglings attack next, each one attacking the first alive Marine once.
- If either group is reduced to zero members, the program should stop and announce the winner as the side that still has living members, or a draw if no one lives.
- Marines and Zerglings are both sub-classes of a class called Entity. The main game loop is implemented in the main() function of the Main.cpp.

Your job is to address the 4 points below:

1. Syntax Errors

You will need to first solve syntax errors so that the program can successfully compile and link to form an executable application. You are allowed to edit any part of the code which contains syntax errors, however you are not to delete overall systems within the code.

Hint: A Linker Error is a mismatch between two compiled object files (cpp) when it attempts to link them to form an executable. These can occur for a few reasons, either something exists within multiple files and must be a duplicate, or it does not exist at all. Remember that a function or class declaration is not the actual definition, it is simply a placeholder stating "hey, this thing will exist somewhere".

2. Logical Errors

You may find that even if you can compile and run the program, the result might not run as expected or even as intended. Look for issues where the code does not match the intention of the program's game mechanics and adjust them so that the game functions as specified.

After running the program, are you sure that it works in all possible situations? If a designer you're working with needed to change the statistics of units, would the fight mechanics still work?

3. Efficiency

There may be sections of the code that are unnecessary for its successful implementation of the game mechanics. There may also be inefficient parts of this code.

See if you can find a way to make the program run in the most efficient and clean way possible.

4. Clean Code and Future Use

Look at the overall structure of the code and address the following:

- Is there another way to structure the code that would improve performance and readability?
- Is there another way that would make it easier to extend, like adding another squad or unit type?
- Is it easy for a designer to modify unit stats / variables, or is there a way to expose this extendability?