Oracles In the Code:

1. Address Sanitizer
    a. By enabling the address sanitizer for the C++ build I've allowed checks of various memory errors that could potentially occur in the program
2. Scaling
    a. I took the quadrilateral classes, found the smallest possible variant of each, and then scaled those points up to the limit of our program (100) to check and make sure the program would classify correctly within the extremes of the limits it imposes
3. Assertions
    a. This one was a bit tricky as we already have input checking. At first I considered asserting the inputs were valid in each of my functions that determined whether something was a quadrilateral of that class. However, wanting to avoid the unnecessary overhead, I decided to do some more minor assertions for making sure the input wasn't empty and a second assertion check for invalid inputs even after checking them in a function. This serves as a sanity check and ensuring nothing weird is going on with the program.
4. Nop
    a. I decided to try out placing a nop into my code as well. It's just a semi-complicated math problem put there to make sure that the output would still remain consistent. I placed in my function that checks for valid inputs-- ground zero for nasty bugs.