The folder “DualNumbers” contains the same program as the “FiniteDifferences” exercise, but now uses dual numbers instead of finite differences to calculate the derivative.

Your mission is to implement the body of the CDualNumber operator overload functions to make the program work.

If that sounds challenging, don’t overthink it 😊

Dual numbers are just like complex numbers, except instead of .

Below are two examples to get you started.

**Example: Adding Dual Numbers**

Adding two dual numbers together is as easy as adding their real parts and their dual parts together.

If you replaced in the above with a variable , it would be the same thing. There’s nothing special about here.

**Example: Multiplying Dual Numbers**

Multiplying dual numbers together isn’t much harder if you remember F.O.I.L.:

Remembering that , you can simplify the above to:

# More Things To Try:

1. There is no longer an epsilon to tune, like there was with finite differences, but there is still a gradient step size. Does modifying that have the same behavior with dual numbers as it did with finite differences?
2. Modify the function F to make multiple parameters. How would you change the dual number implementation and the rest of the program to support that?
3. CHALLENGE: What other dual number operations can you implement? Can you implement square(), sqrt(), pow()? How about division? Check the solutions document for a walk through of division and sqrt(), if you get stuck!