The folder “FiniteDifferences” contains a main.cpp that does function optimization using finite differences – except there is a TODO to calculate the finite differences! Your mission is to implement that function.

The function F() is the function, and the function FDerivative is what you are supposed to fill out.

If you implement it correctly, the code should find an x value near -1, with a y value of -2.

Here are some more experiments to try:

1. If you implemented forward differences, try also implementing central differences. Does one reach the correct answer sooner than the other?
2. Finite differences requires an epsilon. What happens if you make a larger or smaller epsilon? Is there an optimal value?
3. The code that optimizes the x value uses a step value c\_gradientStepSize. What happens if you make that larger or smaller? Is there an optimal value?
4. The function doesn’t have a minimum but it does have a maximum. How would you find that maximum?
5. If you change the function F to another function, does it still work? Do you have to adjust epsilon or the gradient step size? What happens if you put an for loop in the F() function? What if you put an if statement in the F() function, to make a “split domain” function?
6. Try making the function F take more than one parameter (like x and y, or x,y,z). Are you able to modify the rest of the program, including the derivative (now gradient) calculation to find the minimum value?