Michael Dandrea

C10 Writeup

This project combined C8 and C9 to make a new driver program called mul\_matrix\_2d\_faulty. The goal of this assignment was to simulate a bit flip like in C9, while the matrix is being multiplied, rather than after everything was done.

To do this, I created the following functions:

A screen shot of a computer program

Description automatically generated

The mul\_matrixFaulty function does the same as the original mul\_matrix function, but if we are at the row index, column index, and term to be corrupted specified by the arguments, it then calls the mul\_ints function.

The mul ints function takes takes the two terms to be multiplied and the bit to corrupt. It multiplies the A term with the B term after the B term has had the specified bit flipped. While this function isn’t specifically necessary, it adds better readability to the program.

Here is a screenshot of the program working with the workflow given in the assignment description:  
A screen shot of a computer

Description automatically generated

If we use the detect\_data\_2d program on the matrix, it gives us this:

A screen shot of a computer

Description automatically generated

This still correctly identifies the row, column and bit that the error occurs at.

Here, I correctly multiplied the Acs and Bcs matrices into matrix D so I can compare it with the corrected matrix:

A computer screen with white text

Description automatically generated

Now, the corrected matrix looks like this:

A computer screen with white text

Description automatically generated

Comparing it the matrix that was multiplied correctly, they are the same matrix.

The hardest part of this assignment was figuring out what it was actually supposed to do. Once I figured it out, the code showed above was my original idea, and it worked out. I just had to add proper error checking to the program.

Here is the program run with the leaks command on mac (same function as valgrind on the ci):

A screenshot of a computer

Description automatically generated

As shown, there are no memory leaks.