James Soto

7-13-2023

CS-405 Secure Coding

4-2 Milestone: Unit Testing

The process for using the Google Test framework to test the C++ code on Clion included a series of steps that involved understanding the code that needed to be tested, defining the testing environment, creating test cases, and executing those tests. While we were given instructions on how to set up the Google testing environment, the instructions were for windows and Visual Studio which are not useable on a Macbook. I used CLion for this assignment and had to follow the instructions on how to setup the Google tests environment and get everything to work as expected for this assignment. Once I had the program running as expected, I ran the provided code from test.cpp file and matched the output of the provided google test guide, where 4 tests passed and 1 failed. The first step I took was to analyze the given code. It consisted of a `CollectionTest` fixture, which sets up a unique pointer to a vector collection of integers. The fixture provided methods for setup and teardown before and after each test, as well as a helper function `add\_entries()` to add elements to the collection. The testing environment was provided as a class `Environment` derived from `::testing::Environment`. It was overriding the `SetUp()` and `TearDown()` methods, which get called before and after the entire test suite runs. In this case, `SetUp()` was initializing a random seed. The next step was to create test cases for each scenario specified in the TODO list, where I also added comments to each line of code to specify what the code was intended to do. I then created 1 positive unit test and 2 negative unit tests that tested if the collection behaved as expected when certain operations were performed. I then compiled and ran the tests where Google Tests would execute all the tests and output the results showing if a test passed or failed. If any of the tests failed, I did debugging, where I identified the issue and made the proper adjustments to the code before compiling and executing again until I had all passing tests.

