# **CS 405 4-2 Milestone: Unit Testing**

# **Austin Donaubauer**

# **Southern New Hampshire University**

# **CS-405 Secure Coding**

# **Wasim Alim**

# **July 26th, 2025**

# **CS 405 4-2 Milestone: Unit Testing**

Unit testing is a very important development step for a programmer. Testing the code completely is challenging because no one can write up all the different scenarios that a user may encounter with a system. But if no testing is done in development, you would lose confidence in your code and users may run into many different errors.

In this milestone we were given a template of a Google Testing program for testing functionality of a vector. This set up a good program to begin writing the unit tests. I began with the first test which was included in the template to test that the collection is created, and size is zero. The next test is to see if the collection is empty when created. Another test was included in the template to show a failed test in the output. This test was uncommented during testing but has since been commented again to remove it from the output for clarity. The first full test I wrote was for testing adding a single value to an empty collection. Using ASSERT\_TRUE and ASSERT\_EQ to assert that initially the collection is empty, and its size is zero. Then I used the helper function to add a single entry to the collection. Then using EXPECT\_TRUE and EXPECT\_EQ to test the collection is not empty and the size is now one after adding the entry. These tests explicitly prove that adding one entry to the collection worked as expected. The next test was similar as it requested to test that adding five entries works as expected. Using the same approach as the previous test, I asserted that the collection is initially empty and its size is zero, then added the five entries to the collection, and then expected that the collection was not empty and that the size is now five, which confirms the functionality of adding five entries. The next test was to test the max size and how it may be affected by adding entries multiple times. Like the previous test, I tested the size and then compared the size to the max size before adding additional entries. I did the same for the next test of capacity. The following three tests were testing the resize functionality. First, the test checked that resize increased the size from the previous size, the second test checked that the size was decreased, and lastly the resize was tested that it reset the collection size to zero. Another test was used to test the clear functionality, and this was done by first adding entries to the collection, then using clear, and then checking that the size of the collection was set to zero after the clear. Similarly, to test the erase functionality when erasing the begin and end of the collection. This was tested by erasing the begin and end and then checking that the size of the collection was zero as expected. The reserve functionality was then tested by adding entries, and then reserving more capacity, and then checking the previous capacity is less than the new capacity after reserving more. The last templated test is to test that an out of index exception is thrown when using the at function to get a entry of the collection. This was done by first adding entries to the collection and then using EXPECT\_THROW to check that using the at function to get an entry out of index of the size of the collection. The two custom tests I wanted to create one positive, and one negative test as required. The first test I tested the swap functionality. I created a local vector and used that to swap the collections. I then checked that the initial collection was now the local vector, and that the local vector became the initial collection after swapping. The last test I wrote was another test of the out of index exception. I tested a second edge case for the at function on an empty collection would throw the out of index exception.

I found unit testing pretty simple and easy to work through as I enjoy figuring out the scenarios and how to test each case. I think testing is one of my stronger coding skills and I think I did a good job of covering the tests that were required in this milestone.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.