A screenshot of a computer program

Description automatically generated

First, we set up a global test environment using a custom Environment class, where we initialize a random seed in the SetUp() method to ensure tests get consistent results. Then, we create a test class called CollectionTest that holds a smart pointer to a collection (a std::vector<int>) which will be used across different tests. This test class has SetUp() and TearDown() methods that initialize and clean up the collection before and after each test.

A helper function called add\_entries() is included to populate the collection with random numbers, making it easier to test different scenarios. The tests themselves include simple checks like verifying that the collection starts empty and is correctly created. There are also tests to confirm that values can be added, the collection can be resized, and operations like clear() and erase() behave as expected. Additionally, we test for exceptions such as std::out\_of\_range when accessing invalid indices with at(), and handle edge cases like ensuring negative values aren't added or checking if the capacity behaves correctly when resized.

Lastly, the tests are run using the Google Test framework. We use assertions like ASSERT\_TRUE and ASSERT\_EQ to verify that everything works as expected. There's also a test (AlwaysFail) intentionally designed to fail, which helps us see how a failed test appears in the results. These tests help ensure that the collection behaves correctly under different conditions and that any edge cases are properly handled.