Week 9: Introduction to Software Testing

`W9_Practical.cpp`

Objective: Write a simple unit test to verify a function's correctness. **Task:**

1. Create a simple function `int add(int a, int b)`.
2. In `main()`, write a "test" for this function using an `if` statement or `assert`. Check if `add(2, 3)` correctly returns `5`. Print a "Test Passed" or "Test Failed" message.
3. Use the `<cassert>` library for a more formal test.

Solution:

C++

```
#include <iostream>

#include <cassert> // Required for the assert() macro



// 1. The function to be tested.

int add(int a, int b) {

  return a + b;

}



int main() {

  // ----- Test 1: Using a simple if-else statement -----

  std::cout << "Running manual test..." << std::endl;

  int expected = 5;

  int actual = add(2, 3);

  if (actual == expected) {

    std::cout << "Test Passed: add(2, 3) correctly returned 5." << std::endl;

  } else {

    std::cout << "Test FAILED: add(2, 3) returned " << actual << ", but expected " << expected << "."
<< std::endl;

  }



  std::cout << "\n-----------------------------\n" << std::endl;
```

```
    // ----- Test 2: Using assert() -----
    // assert() checks if a condition is true. If it's false, the program
    // will terminate and print an error message indicating the failed assertion.
    // This is useful during development to catch bugs early.
    std::cout << "Running assert test..." << std::endl;
    assert(add(5, 5) == 10);
    assert(add(-1, -1) == -2);
    assert(add(10, -5) == 5);
    std::cout << "All assert tests passed successfully!" << std::endl;

    return 0;
}
```