Module One Numeric Overflow

Jordan Whitney

CS-405 Secure Coding

Professor: Alan Spencer

7/4/2025

This program allows us to test how different numeric types (int, unsigned int, float, double) behave when too much is added, or too much is subtracted, and to detect and prevent those problems. Adding the “success = true;” at the start, it assumes success initially and changes to false if something goes wrong. To report to the calling function whether the operation failed due to overflow or underflow I used the “bool& success”. A true/false value passed by reference to the name of the variable that we’re updating. The integer limit checks prevent the numbers that would exceed the numeric limits (“std::numeric\_limits<T>::max()”) or go below zero. Also I have a floating-point check (“std::isinf()”) to catch the overflow and underflow in float, double, and long double since they don’t wrap like integers. This helps safeguard and ensure that the program handles numeric edge cases safely and reliably across a wide range of data types.

A screenshot of a computer

AI-generated content may be incorrect.