**Module One – Numeric Overflow**

**Justin Hancock**

**CS-405 Secure Coding**

**09.02.2024**

I improved the safety of arithmetic operations by adding specific checks within the functions to prevent overflows and underflows. In both the ‘add\_numbers’ and ‘subtract\_numbers’ functions, I introduced a preventative measure to check if the next arithmetic operation would result in a value exceeding the allowable limits of the data type. I checked to see if the current total plus the increment or decrement would exceed the maximum or minimum value that the data type could hold. If that is the case, the formula stops any further additions to prevent an overflow. Both functions return a pair of values, where the first is the result and the second is a Boolean indicating whether the operation was successful or not.

When implementing the overflow and underflow checks, a significant challenge was making sure that characters like ‘char’ data type would display as numeric values rather than ASCII characters that would confuse the interpretation of the results. To resolve this, we used the unary ‘+’ operator in the output statements. This addition forces C++ to interpret and display the character types as their numeric value. For example, without this change, a ‘char’ value of ‘65’ would display as ‘A’, but with the unary ‘+’, it displays as ‘65’. This ensures clarity and accuracy in the output which aids in debugging and validation of the function’s behavior.

A screen shot of a computer program

Description automatically generated