Chris Bull

1/12/2025

CS-405

Module 1 Assignment

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

I began by refactoring the add\_numbers function. I pulled the maximum value possible for the specific type, T, with std::numeric\_limits<T>::max. Then, I checked to make sure that adding “start” and “increment” would not exceed this value. I split this up into two different checks based on whether increment is positive or negative. If “increment” was positive, then I knew the sum would be greater, so I checked to make sure it didn’t go over the maximum value. If increment was negative, then I knew the sum would be lesser, so I checked to make sure it didn’t go below the minimum value of the specific type , T, with std::numeric\_limits<T>::lowest.

Then, I refactored the subtract\_numbers function with a similar process. If “decrement” was negative, then I would check to make sure that the difference didn’t exceed T’s max value (std::numeric\_limits<T>::max). If “decrement” was positive, I checked to make sure that the difference didn’t go below T’s minimum value (std::numeric\_limits<T>::min).

I also added a static global variable flag that would indicate when overflow/underflow occurred. This way, all functions can access and manipulate the value when needed. This value gets reset after each test. If a test fails with overflow/underflow, then the console will display that error instead of the overflow/underflow value.