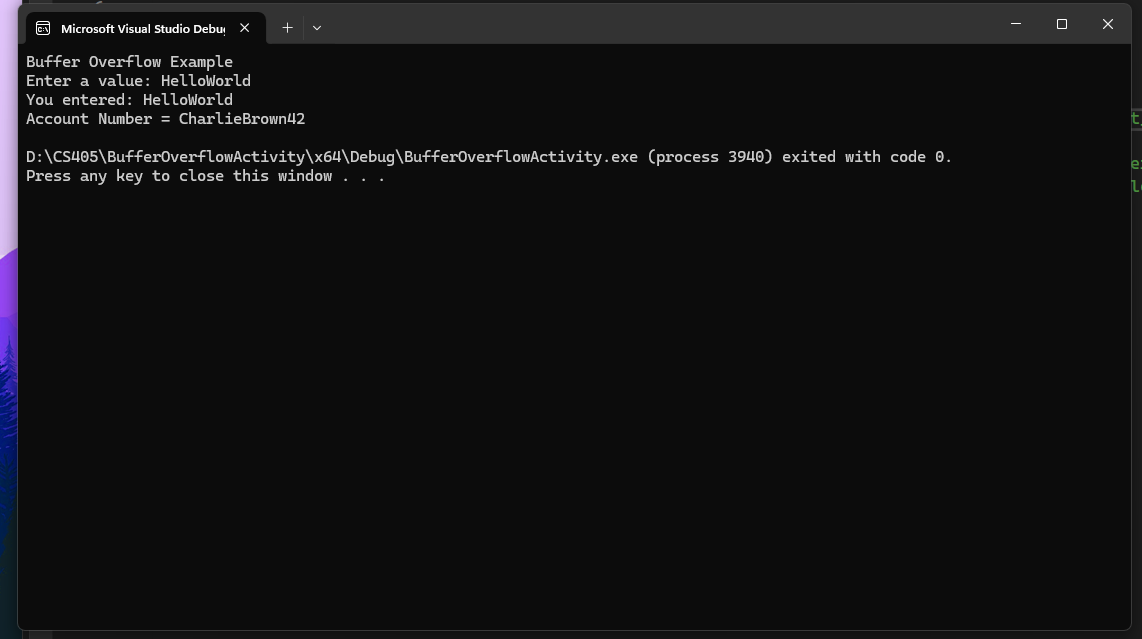
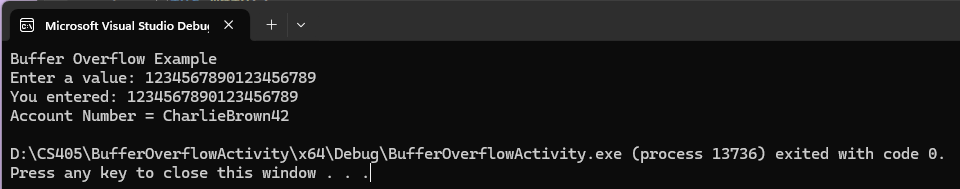
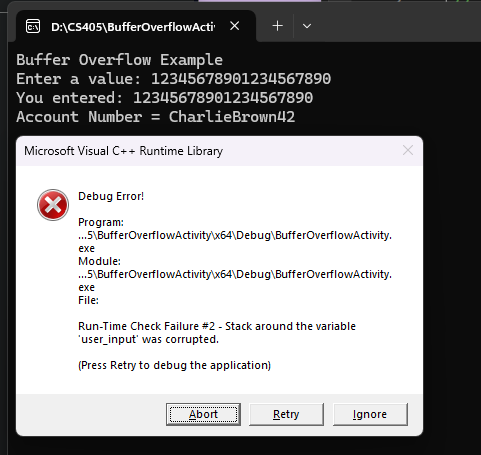
**Test Cases before updating BufferOverflow.cpp**  
Short Input: HelloWorld

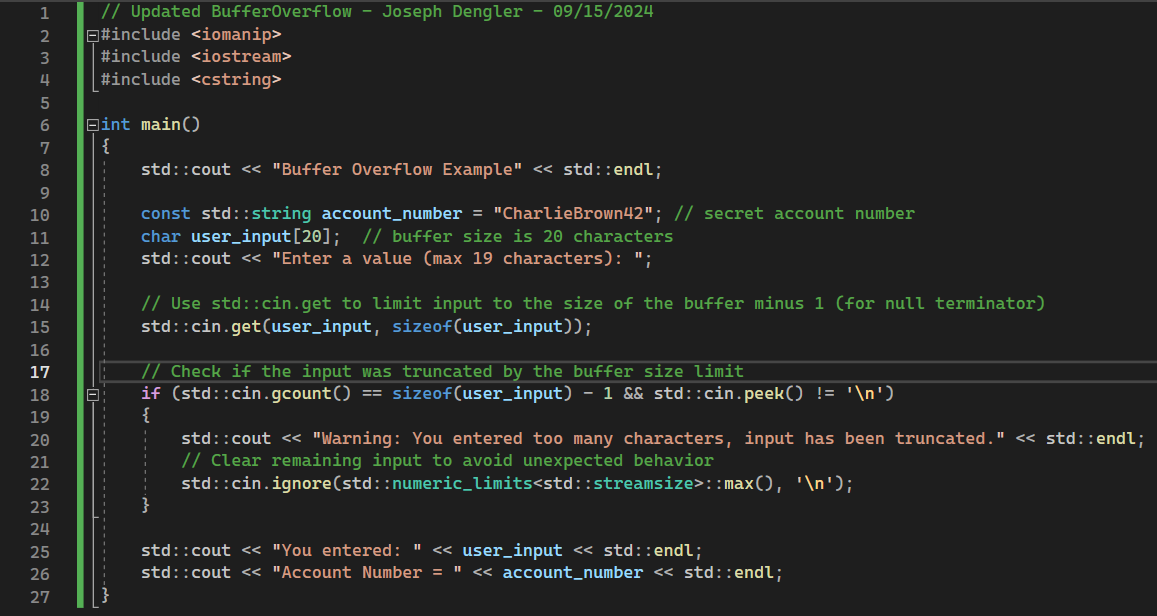


Boundary Test: 1234567890123456789 (19 characters)



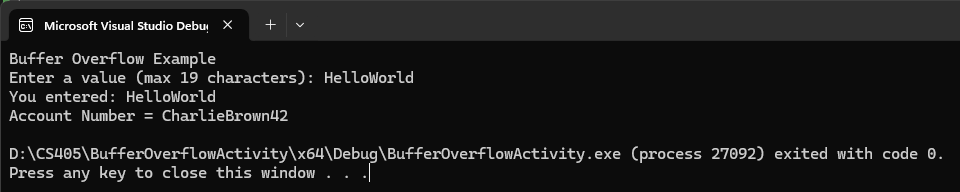
Overflow Test: 12345678901234567890 (20 characters)

Initial findings show the issue of buffer overflow with 20 characters. This showcases the problem of entering more characters than the allocated buffer can handle.

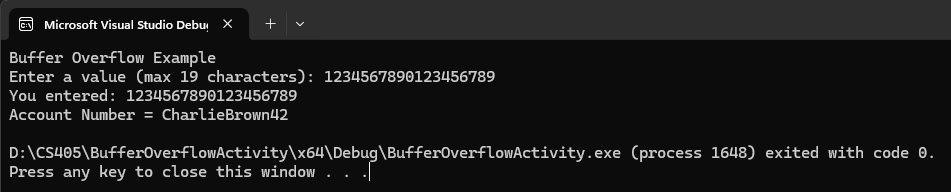
**Updated BufferOverFlow.cpp**

**Test Cases with updated BufferOverflow.cpp**

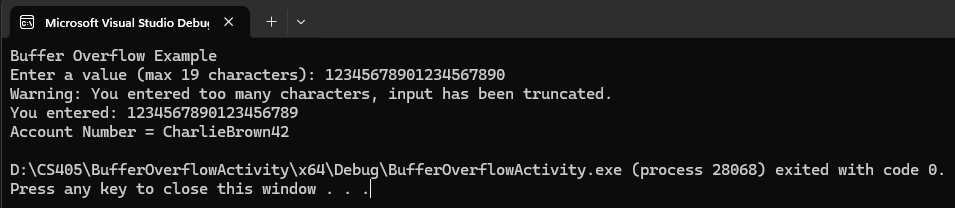
Short Input: HelloWorld



Boundary Test: 1234567890123456789 (19 characters)



Overflow Test: 12345678901234567890 (20 characters)

****

**Summary**

This updated version of the buffer overflow example addresses the issue of preventing overflow while maintaining the original buffer size and account number. The input field user\_input is limited to 19 characters using the std::cin.get() function, which leaves space for the null terminator (\0). The code also checks if the input was truncated by comparing the number of characters read (std::cin.gcount()) with the buffer size minus one. If truncation occurs and the user enters more characters than allowed, a warning message is displayed, and the excess input is cleared using std::cin.ignore() to prevent any further unexpected behavior. Finally, the entered value and the account number are displayed to the user, ensuring the account number remains intact and unaffected by any attempts to overflow the buffer.