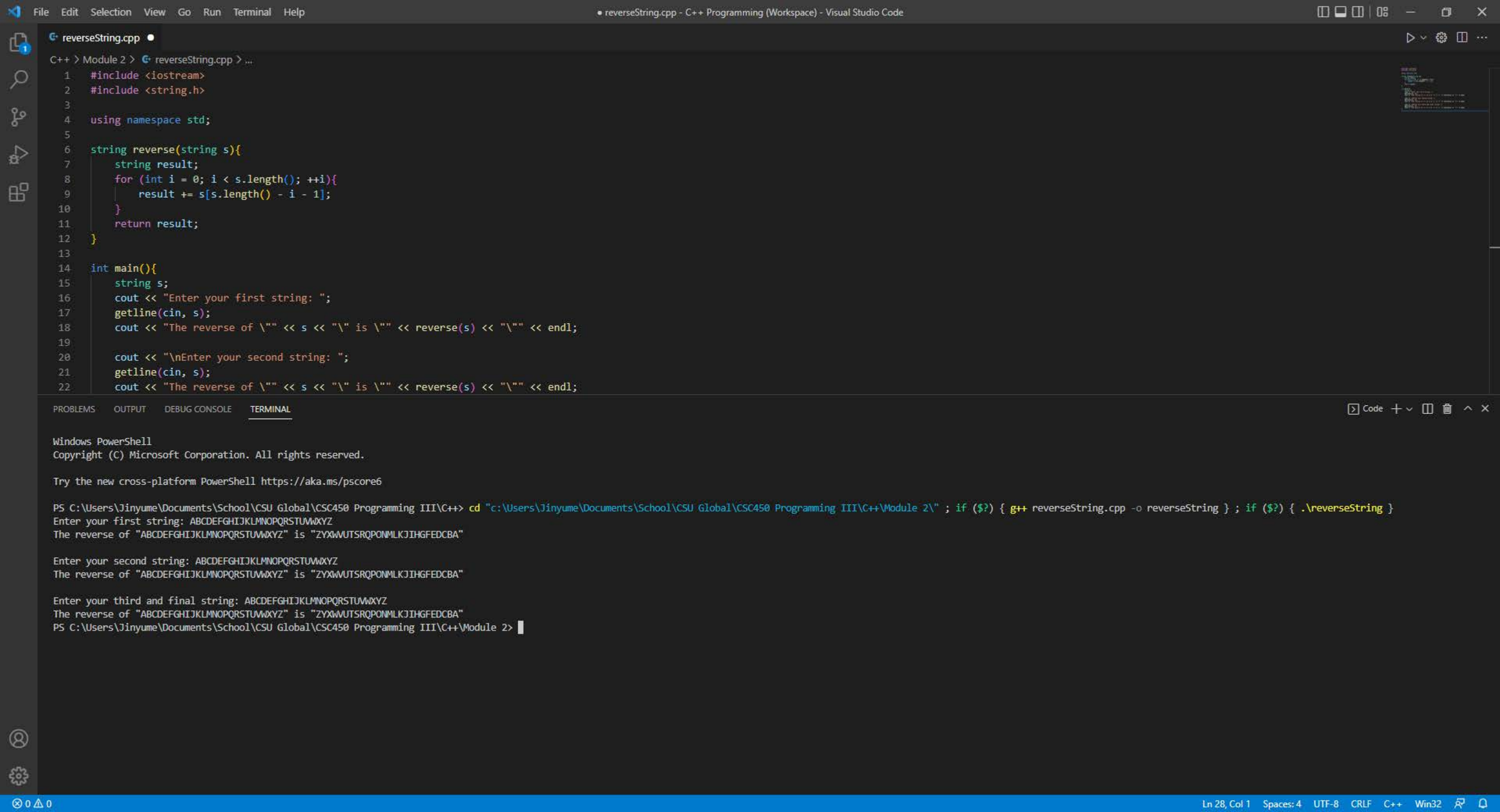
reverseString.cpp •

C++ > Module 2 > reverseString.cpp > ...

```cpp
#include <iostream>
#include <string.h>

using namespace std;

string reverse(string s){
    string result;
    for (int i = 0; i < s.length(); ++i){
        result += s[s.length() - i - 1];
    }
    return result;
}

int main(){
    string s;
    cout << "Enter your first string: ";
    getline(cin, s);
    cout << "The reverse of \"" << s << "\" is \"" << reverse(s) << "\"" << endl;

    cout << "\nEnter your second string: ";
    getline(cin, s);
    cout << "The reverse of \"" << s << "\" is \"" << reverse(s) << "\"" << endl;

    cout << "\nEnter your third and final string: ";
    getline(cin, s);
    cout << "The reverse of \"" << s << "\" is \"" << reverse(s) << "\"" << endl;
}
```


File Edit Selection View Go Run Terminal Help
• reverseString.cpp - C++ Programming (Workspace) - Visual Studio Code
reverseString.cpp
C++ > Module 2 > reverseString.cpp > ...
#include <iostream>
#include <string.h>
using namespace std;
string reverse(string s){
string result;
for (int i = 0; i < s.length(); ++i){
result += s[s.length() - i - 1];
}
return result;
}
int main(){
string s;
cout << "Enter your first string: ";
getline(cin, s);
cout << "The reverse of \"" << s << "\" is \"" << reverse(s) << "\"" << endl;
cout << "\nEnter your second string: ";
getline(cin, s);
cout << "The reverse of \"" << s << "\" is \"" << reverse(s) << "\"" << endl;
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Code
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\Jinyume\Documents\School\CSU Global\CSC450 Programming III\C++> cd "c:\Users\Jinyume\Documents\School\CSU Global\CSC450 Programming III\C++\Module 2\" ; if ($?) { g++ reverseString.cpp -o reverseString } ; if ($?) { .\reverseString }
Enter your first string: ABCDEFGHIJKLMNOPQRSTUVWXYZ
The reverse of "ABCDEFGHIJKLMNOPQRSTUVWXYZ" is "ZYXWVUTSRQPONMLKJIHGFEDCBA"
Enter your second string: ABCDEFGHIJKLMNOPQRSTUVWXYZ
The reverse of "ABCDEFGHIJKLMNOPQRSTUVWXYZ" is "ZYXWVUTSRQPONMLKJIHGFEDCBA"
Enter your third and final string: ABCDEFGHIJKLMNOPQRSTUVWXYZ
The reverse of "ABCDEFGHIJKLMNOPQRSTUVWXYZ" is "ZYXWVUTSRQPONMLKJIHGFEDCBA"
PS C:\Users\Jinyume\Documents\School\CSU Global\CSC450 Programming III\C++\Module 2>
0 0
Ln 28, Col 1 Spaces: 4 UTF-8 CRLF C++ Win32

reverseString.cpp •

C++ > Module 2 > reverseString.cpp > main()

```cpp
#include <iostream>
#include <string.h>

using namespace std;

string reverse(string s){
    string result;
    for (int i = 0; i < s.length(); ++i){
        result += s[s.length() - i - 1];
    }
    return result;
}

int main(){
    string s;
    cout << "Enter your first string: ";
    getline(cin, s);
    cout << "The reverse of \"" << s << "\" is \"" << reverse(s) << "\"" << endl;

    cout << "\nEnter your second string: ";
    getline(cin, s);
    cout << "The reverse of \"" << s << "\" is \"" << reverse(s) << "\"" << endl;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Jinyume\Documents\School\CSU Global\CSC450 Programming III\C++> cd "c:\Users\Jinyume\Documents\School\CSU Global\CSC450 Programming III\C++\Module 2\" ; if ($?) { g++ reverseString.cpp -o reverseString } ; if ($?) { .\reverseString }
Enter your first string: This is a string.
The reverse of "This is a string." is ".gnirts a si sihT"

Enter your second string: And this is also a string, but longer.
The reverse of "And this is also a string, but longer." is ".regnol tub ,gnirts a osla si siht dnA"

Enter your third and final string: This is too.
The reverse of "This is too." is ".oot si sihT"
PS C:\Users\Jinyume\Documents\School\CSU Global\CSC450 Programming III\C++\Module 2>
```

Clarissa Kuter

CSC450 Programming III

Reginald Haseltine

26 June 2022

String Input Application Program Analysis

For the critical thinking assignment due this week, I was asked to create a simple C++ console application that allowed for string input to be received from a user. The program was meant to allow for the input of three strings of varying length, and afterwards, it was required to reverse the string and print its output to screen. In the .cpp file submitted with this analysis, I have written a short and simple application that first initializes a Reverse method that is called and used later in the program to enact the reversing of the characters in the strings that are sent to it. In the main body of the application, the program prompts the user to enter a string on three different occasions. Currently, I believe that the biggest flaw in my program is the fact that the strings do not have a designated maximum size. While this allows for the user to input three separate strings of any lengths, it also does not prevent the user from inputting three strings of the exact same length. I chose to write my program this way because limiting the strings to a specific number of characters may lead my program to becoming prone to an onslaught of string manipulation errors. A buffer overload is one of the most common errors caused by string manipulation, and it is often caused when an unbounded string is input into a fixed length array that is too small to hold the data. If I were to use three fixed arrays to store the user input, the program would be more susceptible to errors, as the user may accidentally input strings that were

too big for the array. A potential fix to this problem would be to utilize `ios_base::width` to limit the amount of characters sent to the array. While a decent quick fix, this fix can also lead to truncation and overflow errors of its own. Thus, I decided that the best action was to use an unbounded string instead.

While users may not encounter them in my program, a few other common string manipulation errors outside of unbounded string copies they may run into with other applications are null-termination errors, string truncation errors, and off-by-one errors. Being able to identify possible causes for these errors is important. If they are not fixed, many of these errors can result in a buffer overload.

Out of the bunch, null-termination errors can occur if a c-style string is unable to properly null terminate. String truncation errors often occur when functions are used to restrict the amount of bytes allowed. If the string surpasses that limit, it is truncated; this can result in data loss and software vulnerabilities. Similar to an unbounded string error, off-by-one errors occur when the user writes outside of an array's bounds. The above are all common reasons that can lead to one or more of those errors occurring. By knowing some of the more common causes behind these errors, programmers will have an easier time identifying where in their code their program may be malfunctioning.