

CS-405 Secure Coding – Module 4 Assignment

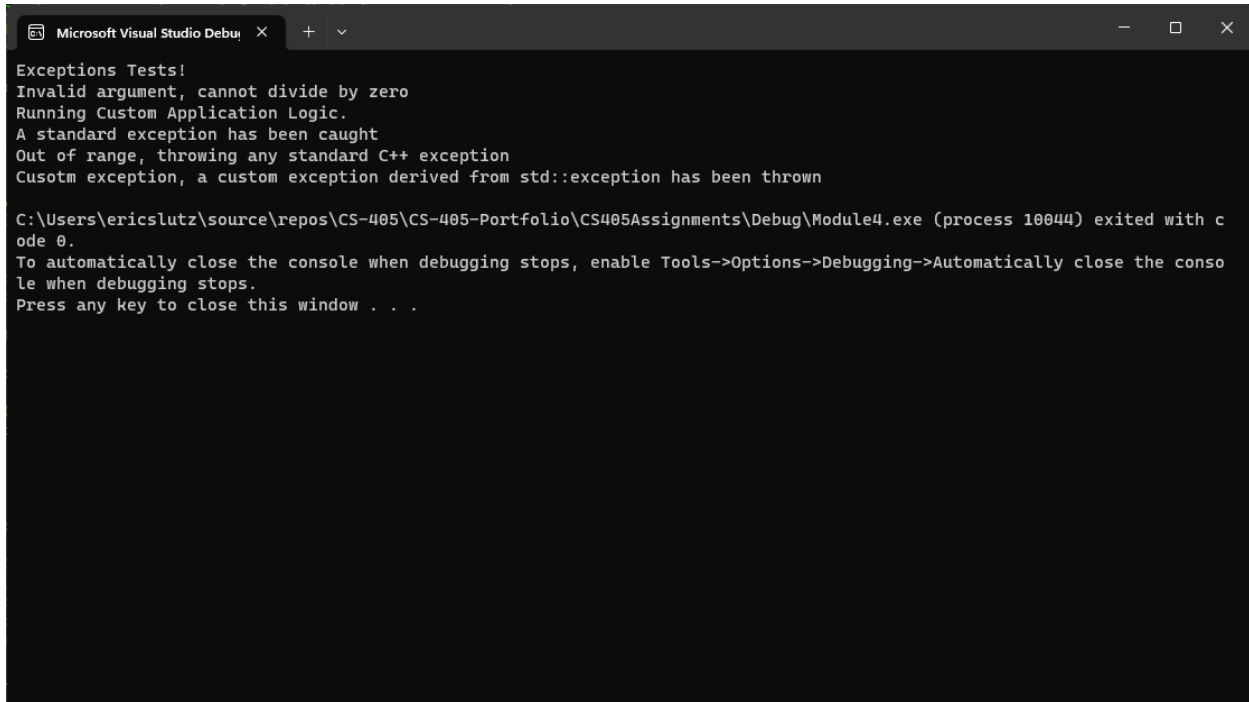
Eric Slutz

Southern New Hampshire University

MODULE FOUR ASSIGNMENT	3
EXCEPTION TESTING SCREENSHOT	3
SUMMARY OF EXCEPTION TESTING PROCESS.....	3

Module Four Assignment

Exception testing screenshot



```
Microsoft Visual Studio Debug Console
Exceptions Tests!
Invalid argument, cannot divide by zero
Running Custom Application Logic.
A standard exception has been caught
Out of range, throwing any standard C++ exception
Custom exception, a custom exception derived from std::exception has been thrown

C:\Users\ericslutz\source\repos\CS-405\CS-405-Portfolio\CS405Assignments\Debug\Module4.exe (process 10044) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Summary of exception testing process

In this assignment, there were many scenarios where an exception could be thrown or caught.

First, for testing purposes, the `do_even_more_custom_application_logic()` method is designed to throw a custom exception that is derived from the standard exceptions. The method

`do_custom_application_logic()` wraps the call to the throwing method in a try/catch block. The try ensures that any exception will be captured and pass it to the catch block. In this case, the catch block is set to only catch standard exceptions and display an error message. If it is not a standard exception, it will not be caught and could cause the program to fail if not caught elsewhere. The `divide(float num, float den)` method could potentially be passed a 0 for the `den` parameter. If that

were to happen, the program would crash because you cannot divide by zero. To protect against that when the method is called the value for den needs to be checked if it is zero. If it is, then an illegal argument exception is thrown. Otherwise, the program carries on as expected. When the `do_division()` method attempts to divide using the other method, that call is wrapped in a try/catch block. If an invalid argument exception is thrown, then the exception is caught, and the exception message is displayed. Lastly, the method calls in the `main()` method are wrapped in a try block with several catch blocks. The catch blocks go from most to least specific. In this case it first tries to catch the custom exception, then any other standard exception, and finally any other type of exception.