Module 4 Exceptions Screenshot and Summary

CS-405 23EW6

Jared Smith

7/23/23

**Screenshot:**

A computer screen with white text

Description automatically generated

**Summary:**

For this assignment I chose to try to keep my exception programming relatively simple, so potential errors are thrown using std::exception derived exceptions, and caught at the function call. To start, I wrote a custom exception struct derived from runtime\_error (which is derived from std::exception) to gain easy access to the what() function and constructor. I then surrounded the main from the suspected code with a try block that catches my custom exception, any standard exception and then looks for any other kind of exception and rethrows it as a standard exception.

In do\_division(), I surrounded our divide function with an exception block that catches overflow errors (since in c++ divide by zero results in inf, technically overflowing the variable), and threw that error in the divide function if the denominator equals zero.

In the custom application logic function, I surrounded the even more custom application logic function call with a try block that catches invalid argument exceptions, and prints the constructed what information. I threw a custom exception after that block which is caught by the exception block in the main function.