CS 405 Module Four: Exceptions Activity Summary

# Process Summary

During the development of the exception handling activity, the following was implemented:  
  
- Wrapped the main() function with a try-catch block to handle:  
 - A custom exception derived from std::exception  
 - Standard exceptions (std::exception)  
 - A catch-all for unknown exceptions  
- Created a custom exception class CustomException  
- Used std::runtime\_error in do\_even\_more\_custom\_application\_logic()  
- Used std::domain\_error in divide() to handle divide-by-zero errors  
- Added comments and followed C++ best practices for readability

# Catch-All Exception Handler: Good or Bad?

The use of a catch-all exception (catch (...)) can be useful as a safety net to prevent a crash,   
especially in production software. However, it is generally better to catch specific exceptions   
for better debugging and clarity. Catch-all should be a last resort or used for logging unknown failures.

# Issues Encountered

- Initially forgot to throw the exception in do\_even\_more\_custom\_application\_logic,   
 causing the handler not to trigger.  
- Also encountered an early return in logic which was unreachable due to the thrown exception—resolved by reordering.

# Fixes

- Carefully reviewed the flow and verified exception types.  
- Ensured all outputs give clear messages including the what() method for clarity.

# Console Output (Screenshot):

A screenshot of a computer

AI-generated content may be incorrect.