Week 8: Error Handling & Debugging

`W8_Practical.cpp`

Objective: Use `try-catch` blocks to handle runtime errors. **Task:** Write a division program that asks the user for two integers. Use a `try-catch` block to handle the case where the user attempts to divide by zero. If division by zero occurs, catch the error and print a user-friendly message instead of letting the program crash.

Solution:

C++

```
#include <iostream>
#include <stdexcept> // Required for standard exception types like std::runtime_error

int main() {
   int numerator, denominator;

   std::cout << "Enter the numerator: ";
   std::cin >> numerator;
   std::cout << "Enter the denominator: ";
   std::cin >> denominator;

   try {
      // The 'try' block contains code that might throw an exception.
      if (denominator == 0) {
         // Throw a runtime_error exception if the denominator is zero.
         throw std::runtime_error("Error: Division by zero is not allowed.");
      }

      // This code only runs if no exception was thrown.
      double result = static_cast<double>(numerator) / denominator;
      std::cout << "Result: " << result << std::endl;
```

```
    } catch (const std::runtime_error& e) {
        // The 'catch' block "catches" the exception.
        // The code here runs only if a std::runtime_error was thrown.
        std::cerr << e.what() << std::endl; // Print the error message.
    }

    return 0;
}
```