Security Injections @ Towson

Module: Exception Handling - CS2 C++

Student: Winston Phillips

Date: 11/09/2017

ID: 551480414

Discussion Questions

With keyboard input, what types of errors may occur?

User errors, "fat finger" errors.

An <i>iostream::failure</i> is an unchecked exception, and indeed must be activated before it can be thrown. Since exception handling is not required, why should the programmer add this checking?

The programmer ought to include this check to specify the type of error that occurred and to use that information to address the issue as needed.
(So a different solution, or possibly output to the screen, ought to occur if a character is put rather than a a floating point number or string).

What types of exceptions may occur when working with a file? Hint, visit the Java API and select the java.io library. Review the list of exceptions included in this library.

Before I answer, I just wanted to state that I looked up the exception class in the c++ exception library. With a file, I would assume that, if one were looking for a specific range of values to be inputted, an out_of_range exception could be thrown. An overflow_error might be thrown as well if the information in the file proves too many bytes.

Describe other types of exceptions (non-IO related) that may occur at runtime. Hint: view the documentation for the <i>exception</i> class, and any of its children.

In runtime, the following exceptions have a possibility of occurring: range_error, overflow_error, underflow_error, regex_error, and system_error.

If a method throws a checked exception, how can the calling method avoid coding a try/catch block to handle the exception? In what cases might the calling method use this option?

The calling would simply bypass the try/catch block method because your are passing that checked exception to the compiler, which handles the exception. There would be no need to code any try/catch blocks

Exception classes are derived from other exception classes. For example, the <i>overflow_error</i> class is derived from the <i>runtime_error</i> class. And, the <i>runtime_error</i> class is derived from the <i>exception</i> class. Rather than check for the exact exception class, the catch clause could specify a super class. What are the advantages and disadvantages of specifying a super class in a catch clause?

When compared to a more specific approach, specifying a super class in a catch clause would provide greater generality, allowing for multiple errors to be caught under the same catch. While this does provide some key advantages, one loses the ability to provide varying solutions for different problems when taking this route. So, for example, if one wanted to have different information displayed on the screen after either a char or float were entered, that might not be possible with the more general approach.