Exceptions

Terms

Exception
Catch an exception
Throw an exception
Rethrow an exception
Call stack

Summary

- We use *exceptions* to report errors that occur while our program is running. An exception is an object that contains information about an error.
- The C++ STL offers a hierarchy of predefined exceptions. All these exceptions derive from the **exception** class.
- We *throw* an exception using the **throw** statement.
- We *catch* exceptions using a **try** statement. A try statement has a try block, followed by one or more catch blocks, each responsible to handle a certain type of exception.
- When using multiple catch blocks, we should order them from the most specific to more generic ones.
- Sometimes we need to catch an exception and re-throw it so it can be handled in a different part of the program. To do that, we use the **throw** keyword without any arguments.
- To create a custom exception, we create a class the inherits the **exception** class in the STL.
- The *call stack* lists the functions that have been called in the reverse order. When an exception is thrown, the runtime looks for a catch block through the call stack. If no catch block is found, the program crashes.

```cpp
try {
    // Code that may throw an exception
    doWork();
}
// Catch blocks ordered from the most specific
// to more generic ones
catch (const invalid_argument& ex) {
    cout << ex.what();
}
catch (const logic_error& ex) {
    cout << ex.what();
}
catch (const exception& ex) {
    cout << ex.what();
}
```