```
/* The following code example is taken from the book
 * "The C++ Standard Library - A Tutorial and Reference, 2nd Edition"
 * by Nicolai M. Josuttis, Addison-Wesley, 2012
 *
 * (C) Copyright Nicolai M. Josuttis 2012.
 * Permission to copy, use, modify, sell and distribute this software
 * is granted provided this copyright notice appears in all copies.
 * This software is provided "as is" without express or implied
 * warranty, and with no claim as to its suitability for any purpose.
 */
#include <exception>
#include <system_error>
#include <future>
#include <iostream>

template <typename T>
void processCodeException (const T& e)
{
    using namespace std;
    auto c = e.code();
    cerr << "- category:     " << c.category().name() << endl;
    cerr << "- value:        " << c.value() << endl;
    cerr << "- msg:          " << c.message() << endl;
    cerr << "- def category: "
         << c.default_error_condition().category().name() << endl;
    cerr << "- def value:    "
         << c.default_error_condition().value() << endl;
    cerr << "- def msg:      "
         << c.default_error_condition().message() << endl;
}

void processException()
{
    using namespace std;
    try {
        throw;  // rethrow exception to deal with it here
    }
    catch (const ios_base::failure& e) {
        cerr << "I/O EXCEPTION: " << e.what() << endl;
        processCodeException(e);
    }
    catch (const system_error& e) {
        cerr << "SYSTEM EXCEPTION: " << e.what() << endl;
        processCodeException(e);
    }
    catch (const future_error& e) {
        cerr << "FUTURE EXCEPTION: " << e.what() << endl;
        processCodeException(e);
    }
    catch (const bad_alloc& e) {
        cerr << "BAD ALLOC EXCEPTION: " << e.what() << endl;
    }
    catch (const exception& e) {
        cerr << "EXCEPTION: " << e.what() << endl;
    }
    catch (...) {
        cerr << "EXCEPTION (unknown)" << endl;
    }
}
```