|  |  |  |
| --- | --- | --- |
|  | In the header <exception> is std::exception and derived classses |  |

* [logic\_error](http://en.cppreference.com/w/cpp/error/logic_error)
* [invalid\_argument](http://en.cppreference.com/w/cpp/error/invalid_argument)
* [domain\_error](http://en.cppreference.com/w/cpp/error/domain_error)
* [length\_error](http://en.cppreference.com/w/cpp/error/length_error)
* [out\_of\_range](http://en.cppreference.com/w/cpp/error/out_of_range)
* [future\_error](http://en.cppreference.com/w/cpp/thread/future_error)(C++11)
* [runtime\_error](http://en.cppreference.com/w/cpp/error/runtime_error)
* [range\_error](http://en.cppreference.com/w/cpp/error/range_error)
* [overflow\_error](http://en.cppreference.com/w/cpp/error/overflow_error)
* [underflow\_error](http://en.cppreference.com/w/cpp/error/underflow_error)
* [regex\_error](http://en.cppreference.com/w/cpp/regex/regex_error)(C++11)
* [system\_error](http://en.cppreference.com/w/cpp/error/system_error)(C++11)
* [ios\_base::failure](http://en.cppreference.com/w/cpp/io/ios_base/failure)(C++11)
* [bad\_typeid](http://en.cppreference.com/w/cpp/types/bad_typeid)
* [bad\_cast](http://en.cppreference.com/w/cpp/types/bad_cast)
* [bad\_weak\_ptr](http://en.cppreference.com/w/cpp/memory/bad_weak_ptr)(C++11)
* [bad\_function\_call](http://en.cppreference.com/w/cpp/utility/functional/bad_function_call)(C++11)
* [bad\_alloc](http://en.cppreference.com/w/cpp/memory/new/bad_alloc)
* [bad\_array\_new\_length](http://en.cppreference.com/w/cpp/memory/new/bad_array_new_length)(C++11)

### Member functions

|  |  |
| --- | --- |
| [(constructor)](http://en.cppreference.com/w/cpp/error/exception/exception) | constructs the exception object  (public member function) |
| [(destructor)](http://en.cppreference.com/w/cpp/error/exception/~exception)  [virtual] | destructs the exception object  (virtual public member function) |
| [**operator=**](http://en.cppreference.com/w/cpp/error/exception/operator%3D) | copies exception object  (public member function) |
| [**what**](http://en.cppreference.com/w/cpp/error/exception/what)  [virtual] | returns an explanatory string  (virtual public member function) |

**Lets use the std::exception to define our own exception class and use it.**

**#include <iostream>**

**#include <exception>**

**using namespace std;**

**class MyException : public exception**

**{**

**public:**

**const char \* what () const throw ()**

**{**

**return "My Exception";**

**}**

**};**

**int main()**

**{**

**try**

**{**

**cout << "Do some Computing until something goes wrong.\n";**

**throw MyException();**

**}**

**catch(MyException& e)**

**{**

**cout << "MyException caught" << endl;**

**cout << e.what() << endl;**

**}**

**}**

**Exception classes can be developed in conjunction with user defined class to provide error handling. So if you define a class MyObject -- you might also define a class MyObjectException.**