# Exception Handling in C++

Exceptions provide a way to react to exceptional circumstances (like runtime errors) in our program by transferring control to special functions called *handlers*.To catch exceptions we must place a portion of code under exception inspection. This is done by enclosing that portion of code in a *try block*. When an exceptional circumstance arises within that block, an exception is thrown that transfers the control to the exception handler. If no exception is thrown, the code continues normally and all handlers are ignored.An exception is thrown by using the throw keyword from inside the try block. Exception handlers are declared with the keyword catch, which must be placed immediately after the try block:

One of the advantages of C++ over C is Exception Handling. Exceptions are run-time anomalies or abnormal conditions that a program encounters during its execution. C++ provides following specialized keywords for this purpose.  
***try***: represents a block of code that can throw an exception.  
***catch***: represents a block of code that is executed when a particular exception is thrown.  
***throw***: Used to throw an exception. Also used to list the exceptions that a function throws, but doesn’t handle itself.

**Why Exception Handling?**   
Following are main advantages of exception handling over traditional error handling.

***1)******Separation of Error Handling code from Normal Code****:*In traditional error handling codes, there are always if else conditions to handle errors. These conditions and the code to handle errors get mixed up with the normal flow. This makes the code less readable and maintainable. With try catch blocks, the code for error handling becomes separate from the normal flow.

***2)******Functions/Methods can handle any exceptions they choose:*** A function can throw many exceptions, but may choose to handle some of them.   
In C++, a function can specify the exceptions that it throws using the throw keyword.

***3)******Grouping of Error Types****:* In C++, both basic types and objects can be thrown as exception. We can create a hierarchy of exception objects, group exceptions in namespaces or classes, categorize them according to types.

**C++ Exceptions:**

When executing C++ code, different errors can occur: coding errors made by the programmer, errors due to wrong input, or other unforeseeable things.

When an error occurs, C++ will normally stop and generate an error message. The technical term for this is: C++ will throw an exception (throw an error).

**C++ try and catch:**

Exception handling in C++ consists of three keywords: try, throw and catch:

The try statement allows you to define a block of code to be tested for errors while it is being executed.

The throw keyword throws an exception when a problem is detected, which lets us create a custom error.

The catch statement allows you to define a block of code to be executed, if an error occurs in the try block.

|  |  |  |
| --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 | // exceptions  #include <iostream>  using namespace std;  int main () {  try  {  throw 20;  }  catch (int e)  {  cout << "An exception occurred. Exception Nr. " << e << endl;  }  return 0;  } | An exception occurred. Exception Nr. 20 |

