# **Exception Handling**

Our program sometimes have runtime problems that prevent from running normally. Such situation is handled by the exception handler. It provides a mean to detect and report an exception occurrence situation so that some action can be taken. The method or process of taking some action after detecting the exception is known as **Exception Handling**.

**Description:**

In C++ exception handling is managed by three keywords -try, catch, throw

1. **try:** It consists of one or more statements that might throw an exception.
2. **catch:** It specifies the type of exception it can handle. It’s immediately after **try** block. There can be one or more **catch** block for one **try** block.
3. **throw:** It accepts one parameter as an argument to the exception handler.

***SYNTAX 1:***

1. Simple try catch block:

try

{  
 //Code that we want to protect from generating

Exception

}

catch (ExceptionHandlerName object)

{

//catch block code statement

}

***SYNTAX 2:***

1. One try may contain multiple catch blocks:

try

{  
 //Code that we want to protect from generating

Exception

}

catch (ExceptionHandlerName object)

{

//catch block code statements

}

catch (ExceptionHandlerName1 object)

{

//catch block code statements

}

catch (ExceptionHandlerName2 object)

{  
 //catch block code statements

}

**Example:**

#include<iostream>

using namespace std;

double divisionby(int num1,int num2)

{  
 if(num2==0)

{  
 throw “Division by zero condition!”;

}

cout<<”answer=”;

return(num1/num2);

}

int main()

{  
 int n1,n2;

cout<<”Enter two numbers for division operation :”;

cin>>n1>>n2;

double n3=0;

//try catch to handle run time error

{  
 n3=divisionby(n1,n2);

cout<<n3<<endl;

}

catch( const char \*msg)

{  
 //it will display error msg

cout<<n3<<endl;

}

return 0;

}

**Output:**

Enter two numbers for division operation:

20

0

Division by zero condition!

Enter two numbers for division operation:

50

25

answer =2