**Exception Handling:**

**Overview:**

> It allows us to keep the program frm crashing back to the operating system.

> (or) it allows us to neatly handle the error in our program, so the program dont crash.

For example:

#*include<iostream>*

*int main()*

*{*

*int numerator = 10;*

*int denominator = 0;*

*std::cout << numerator / denominator;*

*return 0;*

*}*

In the above program we have a denominator zero, which on dividing gives a excetion (i.e, crashes the program). So, in order to avoid it we change the program as below,

*#include<iostream>*

*const int DivideByZero = 1;*

*int main()*

*{*

*try{*

*int numer = 12;*

*int deno = 0;*

*if(deno == 0){*

*throw DivideByZero;*

*}else{*

*std::cout << "The divison is " << numer/deno << "\n";*

*}*

*}*

*catch(int e){*

*if(e == DivideByZero)*

*//Normally for error msg we use cerr*

*std::cout << "Can't divide by zero\n";*

*}*

*return 0;*

*}*

The **try** will check the error and will **throw** the error and **catch** will receive it.

i.e, The ***exception is caught by the try*** statement and ***handled by the catch*** statement

**Exception Throwing**

What to learn?

How to create a exception object.

How to throw the exception, once exception object is created.

Ans:

use stdexcept header.

Syntax:

try {

//check the condition if exception is possible

if(condt)

throw fun;

else

return val;

}

**Cathcing Exception**

Catch block is used to receive the data thrown as exception and it will access based on the value thrown.

When we have no idea about the error we can use,

catch(...){

cout << “Exception thrown” << endl;

}

So this is the catch exception error.

Book to refer:

C++ primer Lippman,

Data sructure with c++ , Ford and topp