Exception Handling in C++

Exception Handling in C++

- The process of converting system error messages into user friendly error message is known as **Exception handling**. This is one of the powerful feature of C++ to handle run time error and maintain normal flow of C++ application.

- **Exception**
- An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's Instructions.

Exception Handling in C++

- Exceptions are runtime anomalies that a program encounters during execution. It is a situation where a program has an unusual condition and the section of code containing it can't handle the problem. Exception includes condition such as division by zero, accessing an array outside its bound, running out of memory, etc.
  In order to handle these exceptions, exception handling mechanism is used which identifies and deal with such condition. Exception handling mechanism consists of following parts:
- Find the problem (Hit the exception)
- Inform about its occurrence (Throw the exception)
- Receive error information (Catch the exception)
- Take proper action (Handle the exception)

Exception Handling in C++

Handling the Exception

- Handling the exception is nothing but converting system error message into user friendly error message. Use Three keywords for Handling the Exception in C++ Language, they are;
- try
- catch
- throw

Exception Handling in C++

- **try:** Try block consists of the code that may generate exception. Exception are thrown from inside the try block.
- **try:** represents a block of code that can throw an exception.
- "**try**" block groups one or more program statements with one or more catch clauses.

Exception Handling in C++

- **throw:** Throw keyword is used to throw an exception encountered inside try block. After the exception is thrown, the control is transferred to catch block.

- Raising of an exception is done by "throw" expression.

Exception Handling in C++

- **catch:** Catch block catches the exception thrown by throw statement from try block. Then, exception are handled inside catch block.

- **catch :**The catch block defines the action to be taken, when an exception occur.

Exception Handling in C++

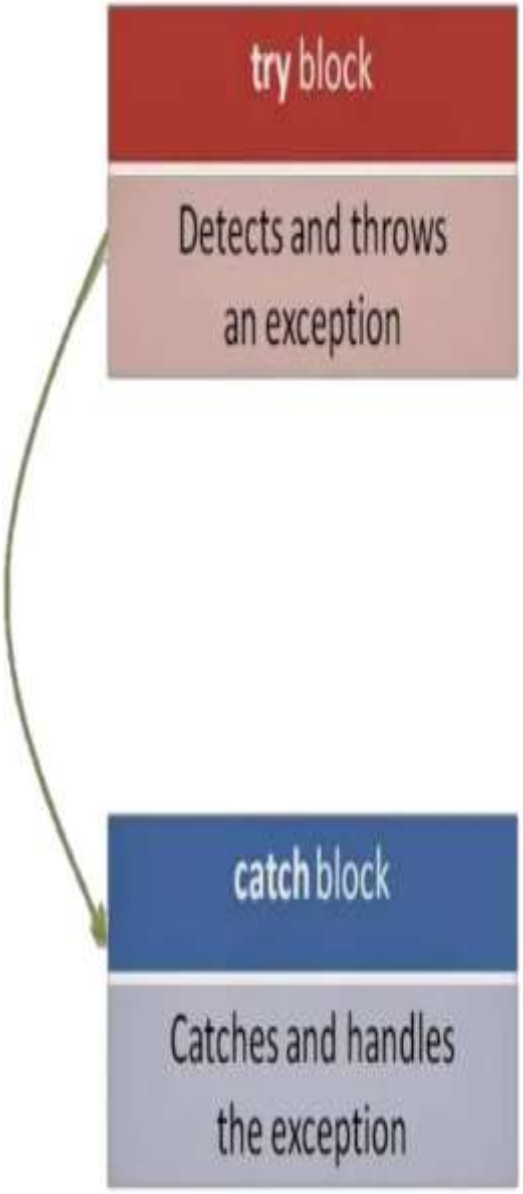
Syntax of Exception Handling

```
try
{
  statements;
  ... ... ...
  throw exception;
}

catch (type argument)
{
  statements;
  ... ... ...
}
```

```
try
{
  statements;
  ... ... ...
  throw exception;
}

catch (type argument)
{
  statements;
  ... ... ...
}
```



Multiple Catch Exception-Syntax

```
try {
   body of try block
}
catch (type1 argument1) {
   statements;
   ... ... ...
}
catch (type2 argument2) {
   statements;
   ... ... ...
}
... ... ...
... ... ...
catch (typeN argumentN) {
   statements;
   ... ... ...
}
```

Example without Exception Handling

```
#include<iostream>
using namespace std;
int main()
{
        int number, ans;
        number=10;
        ans=number/0;
        cout<<"Result: "<<ans;
}
```

Abnormally Terminate Program

Example of Exception Handling

```
#include<iostream>
using namespace std;
int main()
        int number=10, ans=0;
        try
        {
                ans=number/0;
        }
        catch(int i)
        {
                cout<<"Denominator not be zero";
        }
}
```

Denominator not be zero

Example of Exception Handling-1

```
#include <iostream>
using namespace std;
int main()
{
        int n1,n2,result;
        cout<<"Enter 1st number : ";
        cin>>n1;
        cout<<"Enter 2nd number : ";
        cin>>n2;
```

Example of Exception Handling-2

```
try {
    if(n2==0)
        throw n2;    //Statement 1
    else {
        result = n1 / n2;
        cout<<"\nThe result is : "<<result;
    }
}
 catch(int x) {
    cout<<"\nCan't divide by : "<<x;
}

cout<<"\nEnd of program.";

}
```

Output of the Previous Program

```
Enter 1st number : 45
Enter 2nd number : 0

Can't divide by : 0
End of program
```

```
Enter 1st number : 12
Enter 2nd number : 20

The result is : 0
End of program
```

Thank You