

# Exceptions

# Exception

- Definición.- es un evento que , durante el programa ejecutándose, rompe el flujo normal de las instrucciones
- Puede ser generado por software, hardware, lo importante es lanzarlo (throw) y manejarlo (Exception Handling)

# Exception Handling

- **Exception handling** is the process of responding to the occurrence, during computation, of *exceptions* – anomalous or exceptional conditions requiring special processing – often changing the normal flow of program execution. It is provided by specialized programming language constructs, computer hardware mechanisms like interrupts or operating system IPC facilities like signals.

# Try... Catch

- Las excepciones y los bloques de Código Try... catch van de la mano
- En el try va el Código que se desea evaluar y en el catch el tipo de excepción que se atrapa

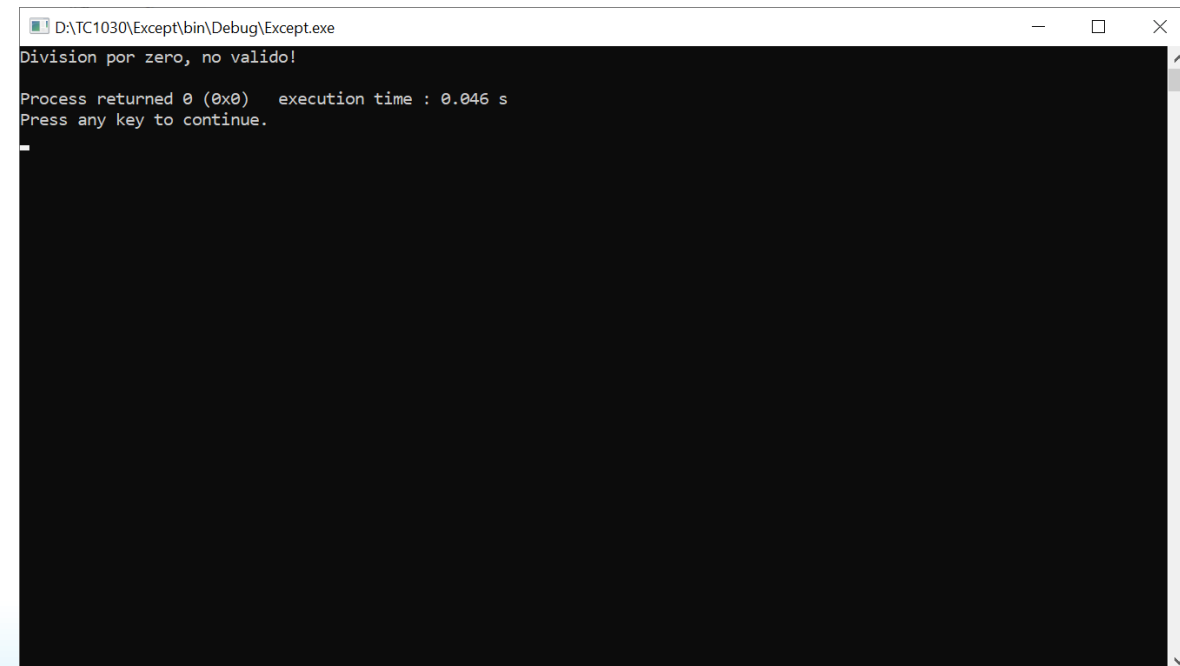
# Consideraciones

- Las excepciones no se deben usar para cambiar el flujo del programa.  
Se usan para reportar errores
- No se deben regresar como un valor de retorno
- No usar excepciones genéricas intencionalmente
- Se pueden anidar bloques try ... catch

# Código

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int x = 1;
8      int y = 0;
9      int z = 0;
10
11     try {
12
13         if( y == 0 )
14             throw "Division por zero, no valido!";
15         z = x / y;
16
17         cout << "el valor de z es: " << z << endl;
18     } catch (const char* msg) {
19         cerr << msg << endl;
20     }
21
22     return 0;
23 }
```

Para este ejemplo, se hace la operación no válida de dividir entre cero, la ejecución de este programa da como resultado



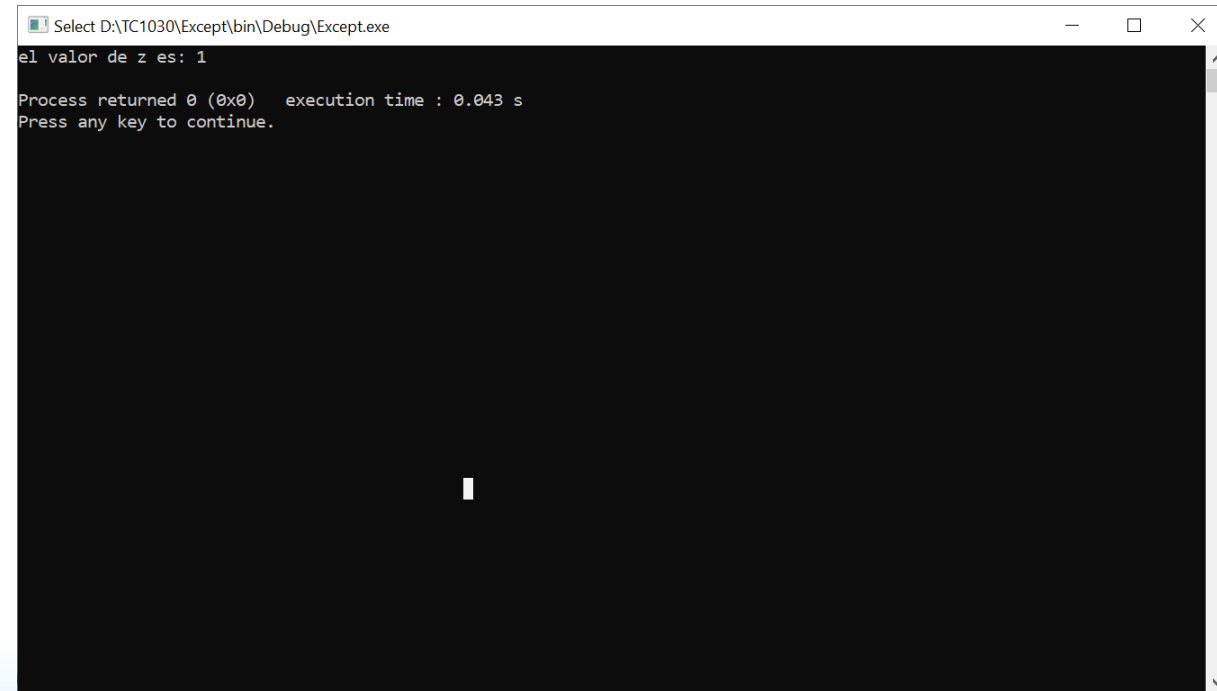
```
D:\TC1030\Except\bin\Debug\Except.exe
Division por zero, no valido!

Process returned 0 (0x0)   execution time : 0.046 s
Press any key to continue.
```

# Código

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int x = 1;
8      int y = 1;
9      int z = 0;
10
11     try {
12
13         if( y == 0 )
14             throw "Division por zero, no valido!";
15         z = x / y;
16
17         cout << "el valor de z es: " << z << endl;
18     } catch (const char* msg) {
19         cerr << msg << endl;
20     }
21
22     return 0;
23 }
24
```

Si se cambia el valor de “y” por 1, la operación es válida y da como resultado :



```
Select D:\TC1030\Except\bin\Debug\Except.exe
el valor de z es: 1
Process returned 0 (0x0)   execution time : 0.043 s
Press any key to continue.
```

# NOTA

De tener experiencia en otros lenguajes, notar que por ejemplo en java o c#, el programa no tiene el throw en la sentencia, el lenguaje se encarga en muchos casos de manejar la excepcion

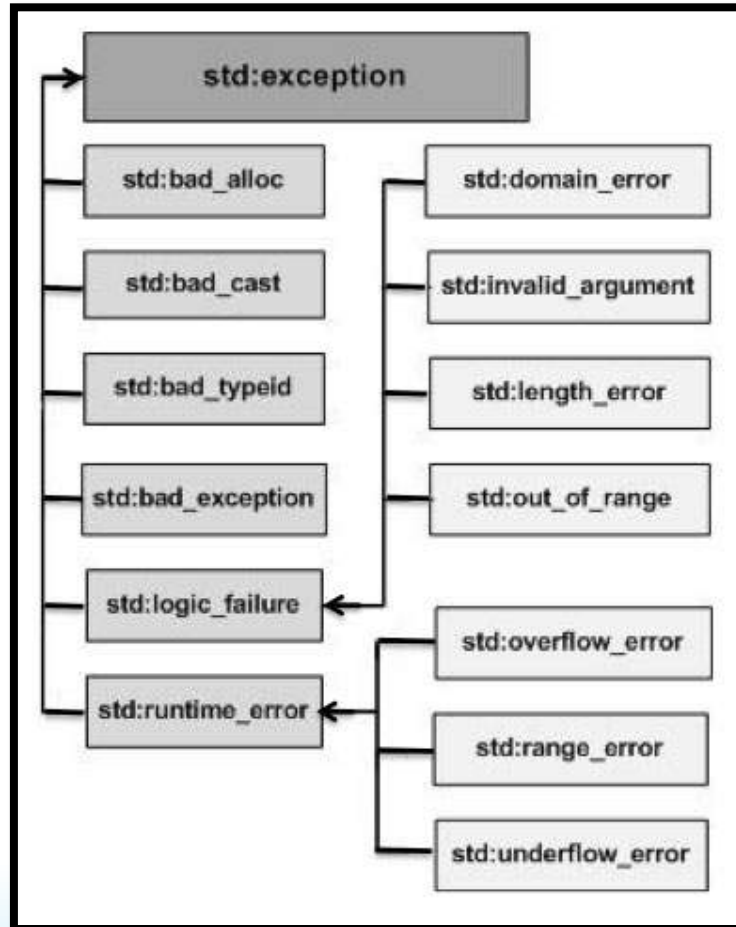
**En C++ el programador es el responsable del manejo de las excepciones**

```
C#  
  
using System;  
  
public class Example  
{  
    public static void Main()  
    {  
        int number1 = 3000;  
        int number2 = 0;  
        try {  
            Console.WriteLine(number1 / number2);  
        }  
        catch (DivideByZeroException) {  
            Console.WriteLine("Division of {0} by zero.", number1);  
        }  
    }  
}
```

No hay throw!!!!



# Excepciones definidas



# Práctica

- recorrer un arreglo de 10 elementos, hasta el elemento 11 y si esto sucede, lanzar una excepción

# Creado excepciones definidas por el usuario

```
// using standard exceptions
#include <iostream>
#include <exception>
using namespace std;

class myexception: public exception
{
    virtual const char* what() const throw()
    {
        return "My exception happened";
    }
} myex;

int main () {
    try
    {
        throw myex;
    }
    catch (exception& e)
    {
        cout << e.what() << '\n';
    }
    return 0;
}
```

## Pasos

- 1.- Crear una clase que derive de exception
- 2.- Sobreescribir el método what()
- 3.- Usar

Quieres saber más?

<http://www.cplusplus.com/doc/tutorial/exceptions/>