2 Exceptions In Functions

- In some cases, an exception generated in a function is not handled in the function
  - It might be that some programs should end, while others might do something else, so within the function you might not know how to handle the exception.
- In this case, the program places the function invocation in a **try** block and catches the exception in a following **catch**-block

```
class DivideByZero{};

double safe_divide(int top, int bottom) throw(DivideByZero);

double safe_divide(int top, int bottom) throw(DivideByZero){
    if (bottom == 0) throw DivideByZero();
    return top / static_cast<double>(bottom);
}

int main(int argc, char **argv)
{
    int numerator;
    int denominator;
    double quotient;
    cout << "Enter numerator:";
    cin >> numerator;
    cout << "Enter denominator:";
    cin >> denominator;
    try
    {
        quotient = safe_divide(numerator, denominator);
    }
    catch(DivideByZero)
    {
        cout << "Error: Division by zero!\n";
        exit(-1);
    }
    cout << numerator << "/" << denominator << "=" << quotient << endl;
    cout << "Bye\n";
    return 0;
}
```

- In the code above, exception class **DivideByZero** was defined as class **DivideByZero { } ;**
  - This class has no member variables or member functions
  - This is a trivial exception class
  - **DivideByZero** is used simply to activate the appropriate **catch**-block
  - There is nothing to do with the **catch**-block parameter so it can be omitted.
- The program above includes a function that throws, but does not catch an exception
- In function **safe_divide**, the denominator is checked to be sure it is not zero. If it is zero, an exception is thrown:
  if (bottom == 0) throw DivideByZero( );
- The call to function **safe_divide** is found in the **try**-block of the program.
- If a function does not catch an exception it should warn programmers that an exception might be thrown by the function.

- An exception specification, also called a throw list, appears in the function declaration and definition:
  double safe_divide(int n, int d) throw (DivideByZero);
- if multiple exceptions are thrown and not caught by a function:
  double safe_divide(int n, int d) throw (DivideByZero, OtherException);

- If an exception is not listed in an exception specification and not caught by the function:
  - The program ends
- If there is no exception specification at all, it is the same as if all possible exceptions are listed
  - These exceptions will be treated "normally"
- An empty exception specification list means that no exceptions should be thrown and not caught

```
int myfunction (int param) throw(); // all exceptions call unexpected
int myfunction (int param);         // normal exception handling
```