EXCEPTION HANDLING

- An exception is any error condition or unexpected behavior in an executing program. Exceptions occur because of errors in program logic or because of insufficient system resources. The programs you write can generate many types of potential exceptions, including when:
    - Your program asks for user input, but the user enters invalid data.
    - The program attempts to divide an integer by zero.
    - You attempt to access an array with a subscript that is too large or too small.
    - You calculate a value that is too large for the answer's variable type.
- These errors are called exceptions because presumably they are not usual occurrences; they are "exceptional." The object-oriented techniques used to manage such errors make up the group of methods known as exception handling. If you do not handle an exception, the running program terminates abruptly. In programming, error and exception handling is very important.

TRY and CATCH

C# has a built-in way to handle exception errors.

- The keyword **try** is used to enclose any statements that might possibly cause an exception error. This block of code is followed by a **catch** block. The **catch** block is a series of statements that will be executed if an exception is thrown from within the try block.
- Here is an example of a try and catch block of code:

```
try
 {
 //Any Exception that happens within this try block will cause
 program flow to immediately transfer to the beginning of the
 catch block
     int bottom = 0;
     int top = 1;
     //the next line throws a divide-by-zero exception
     int result = top / bottom;
 }
 catch (Exception ex)
 {
     //Display nicer error to the user
     MessageBox.Show(ex.Message, "Oops you goofed!");

 }
```

- You can give the **System.Exception** error variable any name you would like in the catch statement. Short names like "e" or "ex" are conventionally used. The **Exception** variable has a property called **Message**. This property will give you a description of the error that caused the exception. If you want to give the user some information about what caused the error, you can easily use this information in a **MessageBox** as shown above.

- The **try** and **catch** blocks are very useful whenever you have a section of code that is complicated, risky, or subject to bad user input. (like a calculator program where a user could attempt to divide by zero.)
- The example above would display.