**Part 44 - C# Tutorial - Preventing exception handling abuse**

In this session we will learn how to prevent exception handling abuse, by rewriting the program, that we have started in Part 43.

**Rewritten example that doesn't use exception handling to control program's logical flow.**  
using System;  
public class ExceptionHandlingAbuse  
{  
    public static void Main()  
    {  
        try  
        {  
            Console.WriteLine("Please enter Numerator");  
            int Numerator;  
            //int.TryParse() will not throw an exception, instead returns false  
            //if the entered value cannot be converted to integer  
            bool isValidNumerator = int.TryParse(Console.ReadLine(), out Numerator);  
  
  
            if (isValidNumerator)  
            {  
                Console.WriteLine("Please enter Denominator");  
                int Denominator;  
                bool isValidDenominator = int.TryParse(Console.ReadLine(), out Denominator);  
  
  
                if (isValidDenominator && Denominator != 0)  
                {  
                    int Result = Numerator / Denominator;  
                    Console.WriteLine("Result = {0}", Result);  
                }  
                else  
                {  
                    //Check if the denominator is zero and print a friendly error  
                    //message instead of allowing DivideByZeroException exception   
                    //to be thrown and then printing error message to the user.  
                    if (isValidDenominator && Denominator == 0)  
                    {  
                        Console.WriteLine("Denominator cannot be zero");  
                    }  
                    else  
                    {  
                        Console.WriteLine("Only numbers between {0} && {1} are allowed",  
                            Int32.MinValue, Int32.MaxValue);  
                    }  
                }  
            }  
            else  
            {  
                Console.WriteLine("Only numbers between {0} && {1} are allowed",  
                            Int32.MinValue, Int32.MaxValue);  
            }  
        }  
        catch (Exception ex)  
        {  
            Console.WriteLine(ex.Message);  
        }  
    }  
}