Workshop - Exceptions

Part 1

Create a class **Triangle** with 3 auto-properties: *Side1*, *Side2*, *Side3* all in double. It has one constructor that accepts 3 double values and set them to its 3 sides. The class also has methods *Perimeter()* and Area() that compute the perimeter and area of the triangle.

Define a custom Exception class **BadTriangleException**. Inside the above *Triangle* constructor, throw a bad triangle exception with message *"Invalid sides"* if the 3 sides do not represent a valid triangle.

Write method *Main()* to test

1. Create a valid triangle and then display its perimeter and area.
2. Create an invalid triangle and then display its perimeter and area. What happens to the program?
3. Put the creating triangles part into a try-catch block, then display the error message if there is any exception. In the 2nd scenario, can the triangle's perimeter and area be displayed?

Part 2

For each of the following sections,

1. Manually "execute" the code with your understanding about Exceptions to write out the output.
2. Then, type the code to Visual Studio and use debugger to trace through every single line of execution. Make sure that your understanding is correct.

Why do the outputs of the two sections differ?

Section 1

```
public class ExcPropagation
{
   public void M1()
   {
      try
```

```
    {
        Console.WriteLine("Enter try block of M1.");
        M2();
        Console.WriteLine("Exit try block of M1.");
    }
    catch (IndexOutOfRangeException e)
    {
        Console.WriteLine("Enter catch block of M1.");
        Console.WriteLine("Exception from: {0}", e.TargetSite);
        Console.WriteLine("Exit catch block of M1.");
    }
    Console.WriteLine("Exit M1.");
  }
  public void M2()
  {
    Console.WriteLine("Enter M2.");
    int y = 0;
    int x = 10 / y;
    Console.WriteLine("Exit M2.");
  }
}
```

```
class TestPropagation {
  public static void Main()
  {
    Console.WriteLine("Enter Main.");
    ExcPropagation mc = new ExcPropagation();
    try
    {
      mc.M1();
```

```
    }
    catch (Exception e)
    {
      Console.WriteLine("Enter catch block of Main.");
      Console.WriteLine("Exception from: {0}",
                        e.TargetSite);
      Console.WriteLine("Exit catch block of Main.");
    }
    Console.WriteLine("Exit Main.");
  }
}
```

Section 2

```
`public class ExcPropagation
{
  public void M1()
  {
    try
    {
      Console.WriteLine("Enter try block of M1.");
      M2();
      Console.WriteLine("Exit try block of M1.");
    }
    catch (IndexOutOfRangeException e)
    {
      Console.WriteLine("Enter catch block of M1.");
      Console.WriteLine("Exception from: {0}", e.TargetSite);
      Console.WriteLine("Exit catch block of M1.");
    }
    finally {
      Console.WriteLine("Finally M1.");
    }
```

```
    Console.WriteLine("Exit M1.");
  }
  public void M2()
  {
    Console.WriteLine("Enter M2.");
    int y = 0;
    int x = 10 / y;
    Console.WriteLine("Exit M2.");
  }
}
```

```
class TestPropagation {
  public static void Main()
  {
    Console.WriteLine("Enter Main.");
    ExcPropagation mc = new ExcPropagation();
    try
    {
      mc.M1();
    }
    catch (Exception e)
    {
      Console.WriteLine("Enter catch block of Main.");
      Console.WriteLine("Exception from: {0}",
                        e.TargetSite);
      Console.WriteLine("Exit catch block of Main.");
    }
    Console.WriteLine("Exit Main.");
  }
}
```