REVIEW LESSON

MTA Course: 10754 Microsoft .NET Fundamentals
Lesson name: Microsoft .NET Fundamentals 1.3
Topic: Exception handing in the .NET Framework (One 50-minute class period)
File name: 10754_Msft.NET_RL_1.3

**Lesson Objective**

**1.3:** Understand structured exception handling in the .NET Framework. *This objective may include but is not limited to*: understanding error handling concepts, exceptions, and exception types.

**Prerequisite student experiences and knowledge:**

This MTA Certification Exam Review lesson is written for students who have learned about application programming with the Microsoft® .NET Framework. Students who do not have the prerequisite knowledge and experiences cited in the objective will find additional learning opportunities using resources such as those listed in the "Resources" section at the end of this review lesson.

Students should have experience creating simple applications with event handlers such as those reviewed in 10754_Msft.NET_RL_1.2.

**Instructor preparation activities:**

- Make copies of the Student Activity 10754_Msft.NET_SA_1.3.
- If desired, change the code examples in 10754_Msft.NET_PPT_1.3 to Microsoft Visual Basic® (Visual Basic code can be found in the presentation notes).

**Resources, software, and additional files needed for this lesson:**

- 10754_Msft.NET_PPT_1.3
- 10754_Msft.NET_SA_1.3
- Microsoft Visual Studio® 2010; students may also use Microsoft Visual C#® 2010 Express or Microsoft Visual Basic 2010 Express, available free at *http://www.microsoft.com/express/Windows/*.

## *Teaching Guide*

### Essential Vocabulary

**exception—**an event that is raised when a method encounters a condition that prevents it from executing further.

**logic error—**an error that occurs when code executes but does not behave in the intended manner.

**syntax error—**an error that occurs when code does not meet the rules (or "syntax") of the programming language in use.

**try—**a keyword used to indicate that a block of code should be monitored for exceptions.

**catch—**a keyword used to designate a block of code used to handle an exception if one is thrown.

**finally—**a keyword used to designate a block of code that executes last, just before the error-handling block loses scope, regardless of whether the code in the catch blocks has executed.

## *Lesson Sequence*

**Activating prior knowledge/lesson staging (10 minutes):**

1. As indicated in the Microsoft PowerPoint® presentation, engage the class in a brief discussion about the code that attempts to divide a value by zero.

**Lesson activity (15 minutes):**

1. Use the presentation to review the errors, exceptions, and exception handling.
2. Finish the review by discussing the review questions as a class (or in small groups, if time allows).

**Assessment/lesson reflection (25 minutes):**

1. Direct the students to complete the student activity, in which they will create a simple calculator that handles common input problems with try-catch blocks.

    - Students should catch *DivideByZeroException* in the "division" button click event handler and *FormatException* in all of the event handlers.

**Resources:**

- **MSDN®: Structured Exception Handling Overview for Visual Basic**
  *http://msdn.microsoft.com/en-us/library/8a9f2ew0.aspx*

- **MSDN: Exceptions and Exception Handling (C# Programming Guide)**
  *http://msdn.microsoft.com/en-us/library/ms173160.aspx*

- **MSDN Beginner Developer Learning Center Lesson 11 (Visual C#)**
  *http://msdn.microsoft.com/en-us/beginner/bb308817.aspx*
  *http://msdn.microsoft.com/en-us/library/25zf0ze8.aspx*

- **MSDN Beginner Developer Learning Center Lesson 11 (Visual Basic)**
  *http://msdn.microsoft.com/en-us/beginner/bb308820.aspx*

**Suggested best practices:**

- Consider having students add exception handling to a project that they created earlier in the course, rather than starting a new calculator project in the student activity.